# 시맨택 웹 서비스 환경에서 시맨틱 질의 어댑터의 설계 및 구현

조 명 현[†] · 손 진 현[††]


## 요 약

시맨틱 웹 서비스는 최근에 많은 연구가 수행되고 있는 시맨틱 웹 기술을 기반으로 웹 서비스를 지원하는 웹 기술이다. 지금까지 시맨틱 웹 서비스와 관련된 연구는 시맨틱 웹 문서 저장 기법과 시맨틱 질의 처리를 위한 추론 엔진 개발 등에 많이 집중되어 왔다. 그러나 근본적으로 시맨틱 웹 서비스 환경을 지원하기 위해서는 사용자 혹은 에이전트가 시맨틱 정보를 질의할 수 있는 효과적인 질의 인터페이스의 제공이 필수적이다.

이에 관하여, 본 논문에서는 복잡한 시맨틱 정보에 대해 일반 사용자의 높은 질의 투명성을 제공하기 위한 시맨틱 질의 어댑터(SQA)를 제안한다. 먼저 DAML-S Profile의 요소를 분석하여 그래픽 기반의 절차적 사용자 질의 인터페이스를 설계한다. 그리고 사용자 인터페이스로 입력된 질의는 시맨틱 질의어인 RDQL로 변환하도록 구현하였다. 이때 RDQL 술어의 결합(disjunctive) 질의 문제를 해결하기 위한 시맨틱 질의어 다중 생성 프로세서를 제시한다.


# Design and Implementation of the Semantic Query Adapter(SQA) in the Semantic Web Service Environment

Myung Hyun Jo[†] · Jin Hyun Son[††]


## ABSTRACT

Abstract The Semantic Web Services is a next-generation Web technology that supports Web services, based on the semantic Web technologies. Until now, the researches on semantic Web services may be forced on the semantic Web document management and the inference engine to efficiently process the semantic queries. However, in order to realize the principle semantic Web environment, it is necessary to provide a semantic query interface though which users and/or agents can efficiently request semantic information.

In this regard, we propose the Semantic Query Adapter(SQA) to provide a high query transparency with users, especially when querying about a complex semantic information. We first design the procedural user query interface based on a graphic view, by analyzing DAML-S Profile documents. And then, we builds a module which a user input query transforms its corresponding RDQL. We also propose the multiple semantic query generating procedure as a new method to solve the disjunctive query problem of the RDQL primitive.


**키워드 : 시맨틱 웹(Semantic Web), 시맨틱 질의어(Semantic Query Language), 시맨틱 웹 서비스(Semantic Web Services), 사용자 인터페이스(User Interface), 온톨로지(Ontology)**

## 1. 서 론

HTML(Hyper Text Markup Language)을 이용한 웹은 네트워크의 발전에 따라 상당한 성장을 이루었다. 이것은 HTML 언어가 가공하거나 관리하기 편리해 일반 사용자 누구나 쉽게 이용할 수 있기 때문이다. 하지만 HTML은 데이터들의 연관성보다는 특정 데이터의 디스플레이 목적만을 위해 만들어졌다. 그래서 지금의 웹은 더 이상의 기능적 성장을 기대하기 어려운 상황이다. 이러한 웹의 한계점을 극복하기 위해서, 월드 와이드 웹 컨소시엄은 시맨틱 웹(Semantic Web)을 주장하였다[1].

시맨틱 웹은 데이터의 디스플레이보다는 다양한 응용 프로그램에서 데이터의 자동 생성(automation), 통합(integration), 재사용(reuse)을 가능하도록 하기 위한 기능들을 부각시켰다. 시맨틱 웹은 웹을 사용하는 주체가 사람뿐만 아니라 에이전트도 자동적으로 데이터를 이해하고, 해석할 수 있도록 만들어졌다.

시맨틱 웹을 구성하기 위해 가장 우선시 되는 것은 프레임 워크를 구성하는 것인데, 시맨틱 웹의 프레임 워크는 온

※ 본 연구는 대학 IT연구 센터 육성·지원 사업의 연구 결과로 수행되었음
본 연구는 한국 과학재단 목적 기초 연구(R08-2003-000-10464-0) 지원으로 수행되었음
† 준 회 원 : 한양대학교 컴퓨터공학과 석사과정
†† 종신회원 : 한양대학교 컴퓨터공학과 교수
논문접수 : 2004년 10월 7일, 심사완료 : 2005년 3월 3일

톨로지(Ontology)이다. 온톨로지는 시맨틱 웹에서 가장 중심에 있는 개념으로서, Tob Gruber는 "공유된 개념화에 대한 정형화되고 명시적인 명세"[2]라고 정의한다. 온톨로지는 특정 도메인에 관계된 단어들과 그 단어들 사이의 관계를 정의하는데, 이런 정의들은 한 개인에게만 국한된 것이 아니라 그룹 구성원 모두에게 동의를 얻은 것이다. 좀 더 구체적으로 말하자면, 온톨로지는 분류(Taxonomy)를 통해 개념을 공유하고, 공리(Axiom)를 통해 그 개념을 정형화한다. 분류는 특정 도메인의 의미와 개념을 정의한 사전(Vocabulary)과 구조(Structure)로 이루어지며, 공리는 이들의 관계(Relationship), 규칙(Rule)들로 이루어진다.

위의 온톨로지를 표현하기 위해서 스키마와 구문구조 등을 정의한 온톨로지 언어가 필요한데, DAML+OIL[3], OWL[4], Ontolingua[5]의 온톨로지 언어가 대표적인 예이다. 본 논문은 B2B 환경의 효율적인 시맨틱 웹 서비스 검색을 목적으로 하고 있기 때문에, DARPA에서 개발한 시맨틱 웹 서비스(DAML-S)[6]의 기반이 되는 DAML+OIL를 기본 온톨로지 언어로 선택하였다. DAML+OIL은 머신이 이해할 수 있는 언어로써, 각 에이전트 간의 자원 통합 및 자원 자동 생성 등의 특징을 갖는 언어이다.

지금까지 DAML+OIL로 구성된 시맨틱 웹 서비스의 검색은 키워드 방식이나 시맨틱 질의어를 직접 입력하는 방식이였다. 그러나 키워드 검색은 간단한 데이터 구조를 갖는 기존 웹서비스에서 적합할 수 있지만 복잡한 데이터 구조를 갖는 시맨틱 언어 및 시맨틱 질의어에는 적합한 검색 방법이 아니다. 특히 시맨틱 웹의 구조를 전혀 모르는 일반 사용자에게 시맨틱 질의어를 이용해 검색하는 방법은 적합하지 못하다. 즉, 시맨틱 웹 서비스를 검색하는 일반 사용자의 질의 투명성(Query Transparency)을 보장하는 기술이 요구된다.

그래서 본 논문은 SQA(Semantic Query Adapter)를 제안한다. 먼저, 시맨틱 언어 및 시맨틱 질의어를 모르는 일반 사용자에게 그래픽 기반의 절차적 사용자 질의 인터페이스를 제공한다. 제공된 인터페이스는 시맨틱 웹 서비스의 메타정보를 담고 있는 DAML-S Profile[6]에 대한 정보를 보여준다. 두 번째로, 절차적 사용자 질의 인터페이스를 시맨틱 질의어로 자동 변환하는 컴포넌트를 구성한다. 서비스 속성을 표현한 DAML-S Profile[6]에 대한 질의 패턴을 분석해서 시맨틱 질의어 RDQL[16]로 변환하는 시스템이다. 이때, RDQL에서 결합(disjunctive) 질의를 가능하게 하기 위한 시맨틱 질의어 다중 생성 프로시저를 제시한다.

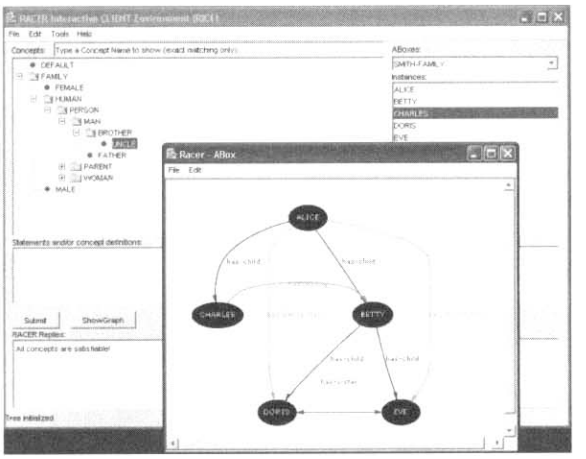
논문의 구조는 다음과 같다. 제 2장에서는 관련 연구로써 기존 온톨로지 기반의 사용자 인터페이스를 살펴보고, 시맨틱 웹 서비스 환경의 기반이 되는 시맨틱 웹 기술 언어 및 시맨틱 질의어에 대해 설명한다. 그리고 시맨틱 웹 서비스를 기존 웹 서비스와 비교하여 기술한다. 제 3장에서는 시맨틱 질의어 자동 생성기(SQA)의 각 컴포넌트 설계 및 특징에 대해 기술한다. 제 4장에서는 SQA의 구현 및 결과를 소개하고, 제 5장에서는 결론을 맺는다.

## 2. 관련 연구

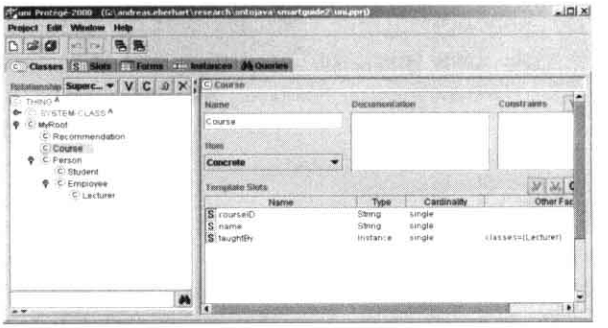
본 장에서는 온톨로지 데이터를 편집하거나, 검색하기 위해 사용되었던 기존 툴들의 인터페이스에 대해 분석한다. 그리고 시맨틱 웹 서비스 환경의 기반 요소인 시맨틱 웹 기술언어 및 시맨틱 질의어에 대해 설명한다. 마지막으로, 기존 웹 서비스와 시맨틱 웹 서비스의 차이에 대해 기술한다.

### 2.1 온토롤지 사용자 인터페이스

지금까지 온톨로지를 위한 사용자 인터페이스는 일반적으로 온톨로지 편집을 위해 많이 이용되어 왔다. 예를 들어, (그림 1)의 Racer Interactive Client Environment(RICE)[22]는 분류(taxonomy)와 A-box[25]구조를 시각화하여 보여준다. [22]는 온톨로지 편집뿐만 아니라, (그림 1)과 같은 그래프를 통해 질의를 수행할 수 있다. (그림 2)의 Protégé[23]와 (그림 3)의 WebODE[24]도 온톨로지를 편집하기 위한 에디터 툴이다.



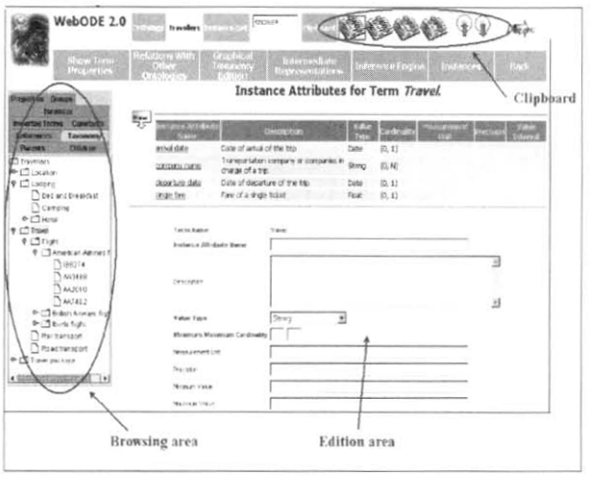
(그림 1) RICE



(그림 2) Protégé

(그림 1, 2, 3)의 사용자 인터페이스는 계층적 구조를 갖는 온톨로지를 생성하기 위해 편집에 관련된 여러 가지 기능을 제공한다. 특히, 현재 까지 작업한 모든 정보를 사용자에게 보여주기 위해, 트리형식의 GUI를 제공한다. 이것은 온톨로지에 대한 모든 정보를 하나의 인터페이스에 보여주

려고 한다는 의미로, 사용자가 온톨로지에 대한 기본적인 이해를 갖는다고 가정할 수 있다. 즉, 위와 같은 사용자 인터페이스를 이용해 온톨로지를 편집하거나 질의하게 되면, 온톨로지에 대한 지각이 없는 일반 사용자는 온톨로지 관련 인터페이스를 이해하는데, 상당한 어려움을 갖는다.




(그림 3) WebODE 온톨로지 에디터

하지만, 시맨틱 웹의 대두에 따라 온톨로지의 이해가 없는 일반 사용자도 온톨로지 사용의 필요성이 부각되었다. 예를 들어, 일반 사용자가 자신이 원하는 정보를 찾기 위해, 검색 사이트를 통해 온톨로지 정보를 검색한다고 가정하자. 온톨로지 정보는 계층적 구조를 갖기 때문에, (그림 1, 2, 3)처럼 트리형식의 인터페이스를 갖는다. 트리는 온톨로지 정보를 하나의 화면에 보여준다는 점에서 장점을 갖지만, 너무 많은 정보를 보여준다. 온톨로지의 구조 및 개념에 대해 전혀 알지 못하는 일반 사용자에게 트리의 정보는 혼동만을 가져온다. 또, 온톨로지는 각 개념들 사이의 관계성을 갖기 때문에 온톨로지 정보를 검색할 때, 다양한 데이터를 복합적으로 입력해야 한다. 복합적 입력 정보는 (그림 3)과 같이 여러 개의 입력 필드를 인터페이스에 보여줌으로써, 일반 사용자의 온톨로지 정보 검색을 어렵게 했다.

### 2.2 시맨틱 웹 기술 언어 및 시맨틱 질의어

본 절에서는 시맨틱 웹 서비스(DAML-S)에서 사용되는 시맨틱 웹 기술 언어 및 시맨틱 질의어에 대해 설명한다. 시맨틱 웹 서비스는 시맨틱 웹 기술언어인 DAML+OIL[3]로 구성되었는데, DAML+OIL은 RDF[9]와 RDFS[12]를 기반으로 확장하였다.

RDF(Resource Description Framework)[9]는 웹상에 존재하는 자원의 정보를 표현하기 위한 언어다. RDF는 데이터의 의미에 초점을 맞춘 메타데이터로써, 시맨틱 웹에 부적합한 XML을 대신하여 제시된 기반구조이다. 구조화된 메타데이터는 생성, 교환, 재사용등이 가능하다. RDF 모델은 자원(Resource), 특성(Property), 서술문(Statement)의 개념으로 구성된다. 웹 페이지나 웹 사이트 등의 모든 사물(thing)은 자원으로 표현되고, 각 자원의 속성, 특성, 관계 등을 특성으로 나타낸다. 서술 문는 특정 자원의 특성에 대한 값을 나타내며, RDF문의 기본 단위가 된다. 이러한 트리플(Tripple) 형태의 구조가 XML의 다양한 형태 구조 문제점을 해결한다[10,11].

RDFS(RDF Schema)[12]는 자원과 특성에 대한 정의나 사용상의 제약 사항을 기술한 것이다. 따라서 RDF의 의미는 이 스키마를 통해서 표현된다고 보면 된다. 스키마는 사전과 비슷한 개념으로 이해하면 되는데 RDF 문을 구성하는 단어를 정의하고 그 단어들에 대한 세부적인 의미를 기술하고 있다. 이를 통해 RDF의 문제점인 태그 이름의 중첩성과 모호성을 해결 할 수 있다[10,11]. 즉, 서로 다른 태그이지만 실제로는 같은 의미일 수 있고, 반대로 같은 태그이지만 사용자에 따라서 다른 의미로 쓰일 수 있는 문제점을 해결한다.

DAML+OIL[3]은 데이터의 의미적 상호 운용성을 가능하게 하기위해 DARPA 프로젝트로 개발된 언어이다. DAML+OIL은 RDF(S)와 OIL을 기반 하여 개발되었다. DAML+OIL은 기존의 RDFS보다 데이터에 대한 정의가 일반적이고, 더 많은 기능들을 추가시켰다[10]. 크게 두드러진 요소는 First-Order Logic을 기반한 공리개념을 포함시킨 것이다[11]. 기본적으로 DAML+OIL로 표현된 온톨로지는 크게 클래스 요소(class element)와 특성 요소(property element)로 구성된다.

시맨틱 웹은 기존의 웹의 확장이기 때문에, 시맨틱 웹 기술 문서에 대한 정보를 얻기 위해 시맨틱 질의어가 필요하다. XML 문서들을 질의 하기위해 XPath[13], XQuery[14] 등을 사용하는 것과 같은 맥락이다. 현재 통용되는 시맨틱 질의어인 RQL, RDQL 그리고 DQL을 비교 분석한다.

*RQL(RDF Query Language)*[15]은 하나의 RDF 문서에 대한 질의를 목적으로 한다. RQL은 RDQL과 유사하지만 RDQL[16]의 AND절의 기능이 없다. (그림 4)는 RQL의 예제를 나타낸다.

```
SELECT X, @P
FROM {X} @P {Y}
WHERE Y like "Pablo"
USING foo for <http://www.daml.org/a.rdf>
```

(그림 4) RQL 예제

- SELECT : 결과로 반환될 변수를 나타낸다.
- FROM : 트리플 형태로 변수를 정의한다.
- WHERE : 각 변수에 조건을 설정한다.
- USING : 질의 중에 사용한 이름 공간(alias)을 설정한다.

RDQL(RDF Data Query Language)[16]은 하나의 RDF 문서에 대한 질의를 목적으로 하며, SQL과 유사한 형태를 가진다. 질의는 하나의 RDF 모델에 대한 제약과 트리플 패턴(tripple pattern)으로 구성된다. RDQL은 아래의 구성요소를 갖으며, (그림 5)처럼 표현된다.

```
SELECT ?x, ?z
FROM <http://www.daml.org/example.rdf>
WHERE (<?x,<foo:hasFather>,?z>), (<?x,<foo:age>,?y>)
AND ?y >= 20
USING foo for <http://www.daml.org/a.rdf>
```

(그림 5) RDQL 예제

- SELECT : 결과로 반환될 변수를 설정한다.
- FROM : 질의 대상 문서를 설정한다.
- WHERE : 반환될 변수를 이용한 트리플의 결합(conjunctive) 질의를 형성한다.
- AND : 변수에 대한 조건을 설정한다.
- USING : 질의 중에 사용한 이름 공간(alias)을 설정한다.

DQL(DAML Query Language)[17]은 DAML+OIL에 대한 질의를 수행하며, RDQL과는 다르게 추론에 관한 정의가 포함되어 있다. DQL은 다른 시맨틱 질의어와 달리 각 에이전트를 질의할 수 있는 프로토콜에 대한 기능도 포함하고 있다. (그림 6)은 DQL의 한 예를 나타낸다.

```
<dql:query xmlns:dql="http://www.dql.org/2002/10/dql-syntax#"
        xmlns:var="http://www.dql.org/2002/10/dql-variables#">
  <dql:premise>                                              } query premise
    <rdf:RDF>
      <rdf:Description rdf:about="#C1">
        <rdf:type rdf:resource="#Seafood-Course"/>
        <drink rdf:resource="#w1"/>
      </rdf:Description>
    </rdf:RDF>
  </dql:premise>
  <dql:queryPattern>                                         } query pattern
    <rdf:RDF>
      <rdf:Description rdf:about="#w1">
        <has-color rdf:resource="http://www.daml.org/2002/10/dql-variables#x"/>
      </rdf:Description>
    </rdf:RDF>
  </dql:queryPattern>
  <dql:mustBindVars>                                         } bind variable list
    <var:x/>
  </dql:mustBindVars>
  <dql:answerKBPattern>                                      } answer KB pattern
    <dql:kbRef rdf:resource="http://ontolinqua.standford.edu/wines.daml"/>
  </dql:answerKBPattern>
  <dql:answerSizeBound>5</dql:answerSizeBound>               } answer bundle
</dql:query>                                                   size bound
```

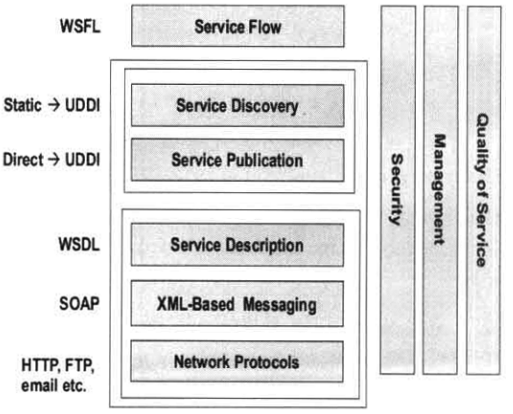
(그림 6) DQL 구조

- query premise : 질의에 대한 제한을 한다.
- query pattern : 질의하고자 하는 내용이 들어간다.
- bind variable list : query premise와 query pattern에서 사용 할 변수 중에 답변에 반드시 포함되어야 하는 변수(mustBindVars), 있을 경우만 포함시킬 변수(mayBindVars)를 설정한다. 답변에 포함시킬 필요 없는 변수(dontBindVars)는 DQL에 설정하지 않는다.
- answer pattern : 답변으로 받고 싶은 형태를 정한다.
- answer KB pattern : 질의에 사용할 KB(Knowledge Base)를 설정한다.
- answer bundle size bound : 질의 결과의 수가 많을 경우 한 번에 가져올 양을 설정한다.
- justification request : 각 질의 결과를 검증한다. 현재 사용되지 않는다.

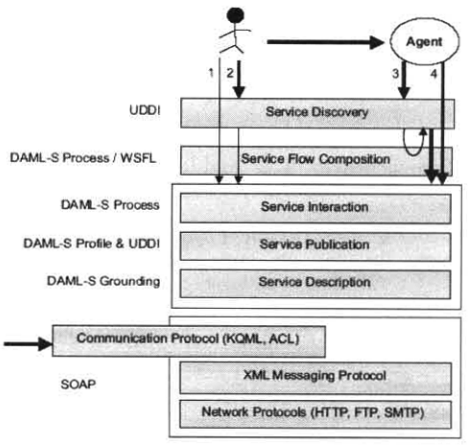
### 2.3 시맨틱 웹 서비스(Semantic Web Services)

(그림 7)은 IBM에서 제안한 웹 서비스 아키텍처 구조를 나타낸다[7]. 웹 서비스는 크게 SOAP, UDDI, WSDL[8]로 구성된다. SOAP은 각 에이전트 간의 메시지를 교환하기 위한 통신 프로토콜이다. WSDL은 각 업체에서 제공할 서비스에 대한 기술 문서이다. UDDI는 WSDL로 기술된 웹 서비스를 고객 또는 에이전트가 찾을 수 있도록 저장하고 관리하는 저장 매체이다.

(그림 8)은 시맨틱 웹 서비스인 DAML-S(DARPA Agent Markup Language-Service)의 구조를 나타낸다[6,7]. DAML-S는 웹 서비스의 확장으로써, 기존 웹 서비스가 제공했던 기능들을 모두 제공한다. 그래서 SOAP은 SOAP의 기능을 모두 받아들이고, WSDL은 DAML-S Grounding[6]이 그 기능을 확장, 보완하였으며, 그리고 UDDI는 일부가 DAML-S Profile[6]이 대신하게 되었다. UDDI는 웹 서비스를 저장할 뿐만 아니라 에이전트가 웹 서비스를 찾을 수 있도록 광고(Publication)하는 역할도 담당했는데, 이 역할을 DAML-S Profile이 담당하게 되었다.




(그림 7) IBM 웹 서비스 아키텍처




(그림 8) 시맨틱 웹 서비스 아키텍처

시맨틱 웹 서비스는 기존 웹 서비스에서 제공하는 특징들(discovery, invocation, etc)뿐만 아니라 머신이 이해할 수 있는 다양한 특징들을 제공한다. 예를 들어, 온톨로지 언어의 트리플 형식을 통해 컨텐트의 의미 모호성을 제거하거나, 시맨틱 웹의 특징을 이용해 자원 통합, 상호 운용성, 자원 복구기능들을 자동화하였다. 특히, DAML-S Process를 통한 각 에이전트 간의 상호호환성(agent interoperation)이나 계획(planning)등은 기존 웹 서비스에선 찾아볼 수 없는 기능들이다.

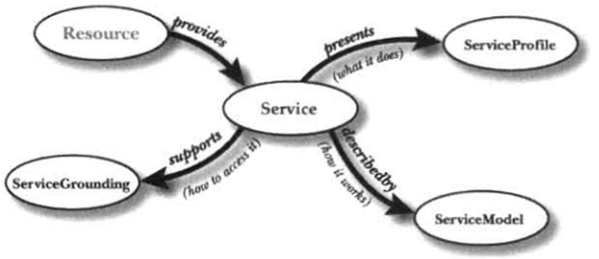
시맨틱 웹 서비스 DAML-S는 기존 웹 서비스 기능뿐만 아니라, 시맨틱 웹을 접합한 새로운 기능들을 제공하기 위해 세 가지 모델을 제시한다. (그림 9)는 하나의 자원에 대해 시맨틱 웹 서비스를 구성하기 위한 세 가지 모델(ServiceProfile, ServiceModel, ServiceGrounding)[6,20]을 보여주고 있다. 세 모델은 자원의 시맨틱 웹 서비스 화를 위해 서비스를 중심으로 분할되어 구성된다.

2.3.1 Profile Model

이 모델은 서비스 기능에 대한 설명을 하는 부분으로서, 찾고자 하는 서비스를 결정하기 위해 서비스 에이전트가 필요로 하는 정보의 타입을 기술한다.

2.3.2 Process Model

이 모델은 서비스가 어떻게 작동되는지에 관하여 기술하며, 서비스가 수행될 때 어떠한 것들이 발생되고 프로세스들이 어떻게 진행되는지 기술한다.




(그림 9) DAML-S 구조

2.3.3 Grounding Model

프로파일 모델과 프로세스 모델이 추상적인 표현을 하는 부분이었다면, 그라운딩 모델은 구체적인 표현을 하는 부분이다. 그라운딩 모델은 기존 웹 서비스와의 통합을 위해 WSDL[8]로 어떻게 바인딩 할 것인지와 에이전트가 서비스에 어떻게 접근할 것인지에 대해 세밀한 정보를 제공한다.

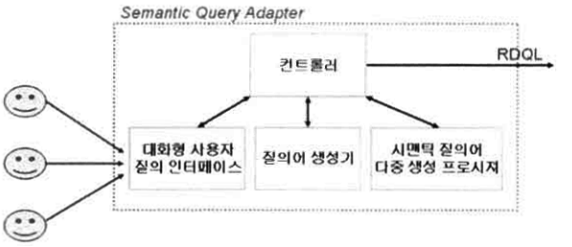
위의 세 모델은 웹 서비스 자동 검색(discovery), 웹 서비스 자동 삽입(invocation), 웹 서비스 자동 구성(composition) 및 상호 운용(interoperation), 웹 서비스 자동 실행 모니터링(execution monitoring)의 네 가지 자동화된 기능을 제공한다. 이것들은 서비스의 실행 프로세스에 따라 순서를 갖는데, 가장 첫 번째로 우선시 되는 것이 웹 서비스 자동 검색이다. 예를 들어, 어떤 고객이 아마존에서 책을 구매하려 한다고 하자. 고객은 다양한 범주에서 책을 찾고, 책을 구입하기 위해 신용카드를 사용하며, 구매한 책에 대해 확인 메일을 받는다. 이 예는 간단한 웹 서비스 시나리오이지만, 가장 먼저 처리해야 할 프로세스는 "서비스 검색"이란 것을 말해준다.

## 3. 시맨틱 질의어 자동생성기(Semantic Query Adapter) 설계

시맨틱 언어와 시맨틱 질의어는 일반 사용자가 이해하고 사용하기에 상당히 복잡하다. 그래서 본 논문은 일반 사용자가 시맨틱 언어와 시맨틱 질의어를 모른다고 가정한다. 그리고 다양한 시맨틱 질의어 중 가장 최근에 개발된 HP사의 RDQL[16]을 기본 시맨틱 질의어로 사용할 것을 가정한다. 이것은 다른 시맨틱 질의어(RQL, DQL, etc) 보다 RDQL이 완전성이나 호환성이 더욱 뛰어나기 때문이다.

3.1절은 본 논문에서 제안하는 시스템 아키텍처를 설명하고, 3.2, 3.3, 3.4절은 시스템을 구성하는 각 컴포넌트를 설계한다.

### 3.1 Semantic Query Adapter (SQA)




(그림 10) SQA 아키텍처

SQA는 컨트롤러, 절차적 사용자 질의 인터페이스, 질의어 생성기, 시맨틱 질의어 다중 생성 프로시저의 총 네 개의 컴포넌트로 구성된다. 각 컴포넌트는 MVC(Model, View, Controller) 아키텍처로 구성되어 있다. 절차적 사용자 질의 인터페이스는 시맨틱 웹 서비스를 검색하기 위한 사용자 질의를 입력받는 그래픽 사용자 인터페이스(GUI)이다. 질의어 생성기는 입력받는 사용자 질의를 시맨틱 질의어(RDQL)로 변환하는 컴포넌트이다. 시맨틱 질의어 다중 생성 프로시저는 RDQL의 OR 연산자 기능을 보완하기 위한 컴포넌트이다. 컨트롤러는 위의 세 컴포넌트의 프로세스 순서 및 오류처리등의 기능을 제공하는 컴포넌트이다. 생성된 RDQL은 데이터베이스 또는 다른 모듈의 입력 질의로 사용되어 웹 서비스를 검색하게 한다. (그림 10)은 SQA에 구성된 컴포넌트들 사이의 구조를 나타낸다.

### 3.2 DAML-S Profile 구조를 통한 절차적 사용자 질의 인터페이스 설계

본 절은 세 부분으로 구성된다. 첫째, DAML-S Profile에

서 웹 서비스 검색을 위한 입/출력 요소를 정의한다. 둘째, 일반 검색엔진에서 제공하는 논리연산자(AND, NOT, OR) 기능을 제공하기 위해, DAML-S Profile 요소들을 분석한다. 셋째, 앞서 정의한 DAML-S Profile 요소들을 사용자에게 편리하게 보여주기 위한 절차를 결정한다.

DAML-S Profile 구조는 <표 1>과 같다. <표 1>은 DAML-S Profile[20]의 모든 정보를 담고 있는데, 크게 세 가지로 구성된다. 서비스를 제공하는 업체 및 관리자에 대한 정보를 담고 있는 '조직'(Organization), 제공하는 서비스의 기능들을 담고 있는 '함수'(Function), 서비스의 지역 및 상태 등의 속성 정보를 담고 있는 '프로파일 속성'(Profile Attribute)으로 구성된다. <표 1>는 DAML-S Profile의 모든 요소 및 속성을 구분하였다.

웹 서비스의 검색을 위해서는 서비스를 찾기 위한 입력 값과 그것에 대한 출력 값을 규정지어야만 한다. 즉, 어떤 요소들이 입력 값이어야 하며, 어떤 요소들이 출력 값이어야 하는지 결정해야만 한다. DAML-S는 머신이 이해할 수 있도록 제안된 웹 서비스이지만, 모든 의미가 머신 독립적으로 구성된 것은 아니다. 특히 DAML-S Profile은 시맨틱 웹 서비스 검색을 위해 사용자와 상호작용을 하기 때문에, 머신이 이해할 수 있는 정보뿐만 아니라 사용자가 이해할 수 있는 정보도 포함한다. 그래서 DAML-S Profile을 이용한 효율적인 절차적 질의를 위해 아래와 같이 입/출력을 가정한다.

〈표 1〉 서비스 프로파일 요소 및 속성

| Type | Element | Attribute |
|---|---|---|
| Organization | serviceName | |
| | textDescription | |
| | contactInformation | |
| | Actor | name |
| | | title |
| | | phone |
| | | fax |
| | | email |
| | | physicalAddress |
| | | webURL |
| Function | input | |
| | output | |
| | precondition | |
| | effect | |
| Profile Attribute | serviceParameter | serviceParameterName |
| | | sParameter |
| | QualityRating | ratingName |
| | | rating |
| | ServiceCategory | categoryName |
| | | taxonomy |
| | | value |
| | | code |

**가정 1.** 시맨틱 웹 기술 언어에 대한 입력 요소들은 '함수(Function)', '프로파일 속성(Profile Attribute)'으로 가정한다.

**가정 2.** 시맨틱 웹 기술 언어에 대한 출력 요소들은 '조직(Organization)'으로 가정한다.

'함수'와 '프로파일'타입은 머신이 이해하기 쉽게 구성된 정보이기 때문에, 웹 서비스 검색의 입력 값으로 정의된다. 예를 들어 사용자가 책을 구입하려는 서비스를 찾을 때, 지불 방법이 신용카드인 서비스만을 찾고자 한다고 가정하자. 여기서 책을 판매하는 서비스들 중 '지불 방법' 입력 요소가 '신용카드'인 서비스만을 검색할 수 있다. 즉 검색 엔진은 사용자가 원하는 기능(Function)을 제공하는 서비스를 검색할 수 있도록 해야 한다. 또, 검색엔진은 서비스 상태(Quality)나 서비스를 제공하는 지역(Geography)과 같은 서비스 속성(Profile Attribute)을 통해 검색할 수 있도록 해야 한다. 하지만 조직(Organization) 타입은 사람이 이해하기 쉽게 구성된 정보이기 때문에, 웹 서비스 검색의 결과 값으로 정의되어야만 한다. 조직 타입은 서비스를 제공하는 사람 또는 업체의 정보를 표현함으로써, 이와 같은 정보를 얻는 고객은 업체와 개인적인 연락을 통해 직접 서비스에 대한 구체적인 정보를 요구할 수 있다. 즉, 조직이란 정보는 서비스 검색 후 필요한 정보이다.

일반 상용 검색 엔진들은 불리안 모델, 벡터 모델 등의 다양한 검색 모델을 통해 다양한 기능을 제공한다. 하지만 다양한 모델을 구성하여도, 입력 질의의 기본적인 논리 연산자는 모두 동일하다. 예를 들어 'not 사람 and 머신' 또는 'not 사람 or 머신'처럼 사용되는 기본 논리 연산자는 AND, OR, NOT이다. 표 2는 표 1에서 나타난 DAML-S Profile의 모든 입력 요소들이 논리 연산자와 함께 사용될 수 있다는 것을 보여준다.

<표 2>의 결합 표는 DAML-S Profile Specification[20]에 정의된 각 요소의 제약사항에 따라 구성되었다. AND, OR, NOT 연산자를 제한할 제약 조건들은 [20]에 정의 되어 있지 않기 때문에 <표 2>와 같은 결과가 도출되었다. 예를 들어, serviceParameter가 DAML-S Profile에서 오직 하나 밖에 없다는 제약사항이 있었다면, serviceParameter 요소는 OR 연산자 기능을 가질 수 없을 것이다. <표 2>에 구성된 요소들은 의미에 따라 구현의 미묘한 차이가 있기 때문에, 5장 구현 및 설계에서 <표 5>로 새롭게 정의된다.

〈표 2〉 DAML-S Profile과 논리 연산자의 논리적(Logical) 결합표

| Type | Element | AND | OR | NOT |
|---|---|---|---|---|
| Function | input | Yes | Yes | Yes |
| | output | Yes | Yes | Yes |
| | effect | Yes | Yes | Yes |
| | precondition | Yes | Yes | Yes |
| Profile Attribute | serviceParameter | Yes | Yes | Yes |
| | QualityRating | Yes | Yes | Yes |
| | ServiceCategory | Yes | Yes | Yes |

**가정 3.** 사용자는 논리 연산자(Boolean Algebra)를 이용한 검색에 대해 기본적인 지식을 이해하고 있다.

논리 연산자를 이용한 질의는 일반 사용자에게 상당히 간편하면서 다양한 질의를 가능하게 한다. 그러므로 본 논문은 논리 연산자(Boolean Algebra)를 절차적 사용자 인터페이스에 적용하기 위해 위와 같이 가정한다.

마지막으로 복잡한 시맨틱 웹에 대한 검색을 일반 사용자가 손쉽게 이해하기 위해서는 표1의 DAML-S Profile의 정보를 절차적으로 보여줘야 한다. 2장의 관련 연구에서 기술한 것처럼, 모든 정보를 한 화면에 보여주는 시스템들은 온톨로지에 대한 기본 이해를 요구하기 때문이다. 그래서 다음의 가정이 정의될 수 있다.

**가정 4.** 사용자는 복잡한 정보를 한 번에 모두 보여주는 것 보다, 절차적(Procedural)으로 보여주는 것이 이해가 빠르고 쉽게 적응할 수 있다.

**가정 5.** 사용자가 절차적으로 정보를 습득할 때, 다음 요인들이 적게 포함된 정보일수록 쉽게 습득할 수 있다.(Lower Information, Higher Understanding).
- 정보 양(Information Amount)
- 정보의 복잡성(Information Complexity)
- 정보의 다양성(Information Variety)

**정의 1.** 가정 1~5를 바탕으로 DAML-S Profile의 입력 요소 절차는 다음처럼 구성한다.

*프로파일 속성(ServiceCategory→QualityRating→ serviceParameter) → 함수(Function)*

가정 1~5는 복잡하고 다양한 정보를 내포하는 시맨틱 언어 및 질의어를 바탕으로 하고 있다. 그러므로 가정 1~5로부터 얻어진 정의 1은 시맨틱 언어 및 질의어에 대한 이해가 전혀 없는 사람들도 좀 더 손쉽고 적응적으로 시맨틱 정보를 추출 할 수 있게 한다. 이것은 시맨틱 웹 서비스 사용자층을 확장시키는 효과를 가져 올 것이다.

### 3.3 질의어 생성기 설계

3.2절에서 정의한 것이 사용자를 위한 절차적 사용자 질의 인터페이스를 생성한 것이라면, 본 절에서는 에이전트(Agent)를 위한 시맨틱 질의어(RDQL)를 생성하는 알고리즘을 설계한다. 즉, 2.2절에서 제시한 RDQL 문법을 바탕으로 절차적 사용자 질의 인터페이스에서 선택된 사용자 질의가 어떻게 RDQL로 생성될 수 있는 지에 대해 설명한다.

시맨틱 웹 기술 언어는 트리플들의 결합(conjunctive)으로 구성되기 때문에, 시맨틱 웹 기술언어에 대한 질의는 트리플들의 결합질의 형태로 구성될 수 있다. 3.2절에서 정의한 것처럼 DAML-S Profile의 입력 질의 요소는 '함수'와 '프로파일 속성'이다. 즉, RDQL의 WHERE절은 트리플들의 결합(conjunctive)질의 형태로 '함수'와 '프로파일 속성' 요소가 구성될 수 있다. 예를 들어, (그림 11)과 같은 RDQL문을 예로 들 수 있다.

```
SELECT ?a, ?b
WHERE (?y, <profile:serviceName>, <congo:Congo_Bookbuying_Agent),
      (?y, <profile:serviceCategory>, ?z), (?z, <rsyn:type>, <profile:UNSPSC>),
      (?z, <profile:value>,?a), (?z, <profile:code>,?b)
USING rsyn FOR <http://www.w3.org/1999/02/22-rdf syntax-ns#>
      profile FOR <http://www.daml.org/services/daml-s/0.7/Profile.daml#>
      congo FOR <http://www.daml.org/services/daml-s/0.7/CongoProfile.daml#>
```

(그림 11) RDQL 예제

본 논문의 사용자 인터페이스는 절차적이기 때문에, 하나의 질의를 형성하기 위해 사용자가 여러 가지 조건을 절차적으로 입력해야 한다. 즉, 사용자가 입력한 사용자 질의를 임시적으로 담을 수 있는 버퍼를 제공해야 한다. 규모가 크지 않고, 절차적인 데이터를 효율적으로 관리할 수 있는 자료 구조는 링크드 리스트(Linked List)이기 때문에, 사용자가 입력한 조건들은 링크드 리스트에 임시적으로 담아서 하나의 질의를 형성하게 된다.

```
LinkedList rdql_list  //임시로 저장될 사용자 질의 링크드 리스트
String input_string  //입력된 사용자 질의
if inputElement = "ServiceCategory"then
      rdql_list ← "(?x, <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>,
                  <"+input_string+">)")
else ifinputElement = "QualityRating" then
      rdql_lis ← "(?x, <http://www.daml.org/services/daml-s/0.9/Profile.daml
                  #qualityRating>, ?a"+n+"), "+"(?a"+n+",
                  <http://www.daml.org/services/daml-s/0.9/Profile.daml
                  #rating>, <"+input_string+">)")
else if inputElement = "GraphicRadius"then
      rdql list ← "(?x,<http://www.daml.org/services/daml-s/0.9/Profile.daml#
                  serviceParameter>,?a"+n+" "), "-"(?a"+n+",
                  <http://www.daml.org/services/daml-s/0.9/Profile.daml#
                  sParameter>, <"+input_string+">)")
else if inputElement = "Inputs"then
    rdql_list ← "(?x, <http://www.daml.org/services/daml-s/0.9/Profile.daml
                  #input>, ?a"+n+"), "+          "(?a"+n+",
                  <http://www.daml.org/services/daml-s/0.9/Profile.daml
                  #restrictedTo>, <"+input_string+">)"
else if inputElement = "Outputs"then
    rdql_list ← "(?x, <http://www.daml.org/services/daml-s/0.9/Profile.daml
                  #output>, ?a"+n+"), "+"(?a"+n+",
                  <http://www.daml.org/services/daml-s/0.9/Profile.daml
                  #restrictedTo>, <"+input_string+">);
end if
```

(그림 12) 사용자 질의를 RDQL로 저장하는 알고리즘

(그림 12)는 웹 브라우저에서 입력한 사용자 질의를 RDQL 형태로 변환하여 링크드 리스트에 저장하는 알고리즘이다. 사용자는 여러 단계에 걸쳐 "서비스가 제공하는 지리적 위치는 어디인가?", "서비스의 상태는 어떤가?", "서비스에 입력 파라미터는 무엇인가?"등의 서비스 조건들을 입력한다. 이러한 입력 조건들은 (그림 12)의 알고리즘에 의해 시맨틱 질의어 RDQL로 변환된다. (그림 12)의 조건 절에서 구성된 inputElement는 RDQL의 WHERE절에 구성 될 술어(predicate)들의 논리곱 정규 형(conjunctive normal form)이다.

### 3.4 시맨틱 질의어 다중 생성 프로시저 설계

RDQL은 SQL을 기반으로 만들어진 질의어이지만, 안타

깝게도 표 2에서 정의한 논리 연산자 중 OR 연산자를 수행할 수 없다. 사실, 시맨틱 웹이 트리플의 결합(conjunctive)으로만 이루어져 있기 때문에, OR 연산자를 필요로 하지 않는다. 하지만 검색의 다양성을 제공하기 위해서는 일반 검색엔진에서 제공하는 모든 불리안 연산자를 제공해야 한다. 본 논문에서 이 문제점을 해결하기 위해, SQA 모듈에 시맨틱 질의어 다중 생성 프로시저를 추가 시켰다. 이것은 OR 연산자(disjunctive)를 필요로 하는 입력 질의 요소들의 질의어를 확장 시키는 방법이다. (그림 13)은 RDQL에 OR 연산자를 적용시키기 위한 알고리즘이다.

```
n                          //입력 요소들 중 OR연산을 필요로 하는 입력 요소 개수
m                          //n개의 각 요소에서 중복을 필요로 하는 실제 값
RDQL rdql_list             //3.3절에서 정의한 알고리즘으로부터 생성된 조건 리스트
RDQL new_rdql[n][m] //새롭게 생성된 RDQL 집합

for i ← 0 to i  ← rdql_list.length do
    if rdql_list(i) include duplicated_elements then
        for  j ←  0 to j  ← m do
            rdql[i][j] ← RDQL(j, rdql_list(i))
        end for
    else
        rdql[i][0] ← RDQL(-1, rdql_list(i))
    end if
end for
```

(그림 13) RDQL에 OR 연산자 기능 제공하기 위한 확장 알고리즘

만일 질의에서 OR 연산자를 선택한 입력 요소 A가 n개, 각 요소에서 중복을 필요로 하는 실제 값 B가 m개 필요하다면, 질의어는 m*n개 생성 될 것이다. 예를 들어, 입력 요소가 'serviceParameter', 'input'처럼 정의된다면, 여기서 OR 연산자를 요구하는 입력 요소가 'serviceParameter'라고 가정하고, 'serviceParameter'의 값으로 'GeographicRadius'나 'profit'의 내부 요소를 갖는 다고 가정하자. 이것은 'service-Parameter'인 'GeographicRadius'의 값이 조건에 맞거나 'profit'의 값이 조건에 맞는 서비스를 의미한다. 다시 말해서, 사용자는 "GeographicRadius == 서울 OR profile>=1000"와 같은 조건에 만족하는 모든 서비스를 찾으려고 하는 것이다. 그래서 이와 같은 질의를 처리하기 위해 생성되는 RDQL의 총 개수는 2개이다.

시맨틱 질의어를 생성할 때, 컨트롤러는 절차적 사용자 질의 인터페이스를 통해 선택된 사용자 질의가 OR 연산자를 포함하는지 체크하여 시맨틱 질의어 다중 생성 프로시저를 통한 질의어 확장 여부를 결정한다.

## 4. SQA 구현 및 결과

3장에서 설계한 SQA는 크게 두 부분으로 구현된다. 먼저, 3.2절에서 설계한 절차적 사용자 인터페이스는 일반 검색 사이트처럼 웹 브라우저로 구현된다. 이때, 일반 검색 엔진에서 제공하는 키워드 방식보다는 사용자 친화적인 방법을 이용한다. 그 후, 절차적 사용자 인터페이스를 통해 입력된 사용자 질의가 시맨틱 질의어(RDQL)로 변환된다. 변환된 시맨틱 질의어는 추론 엔진에 전해지며, 추론된 결과는 HTML 형태로 웹 브라우저에 보여 진다.

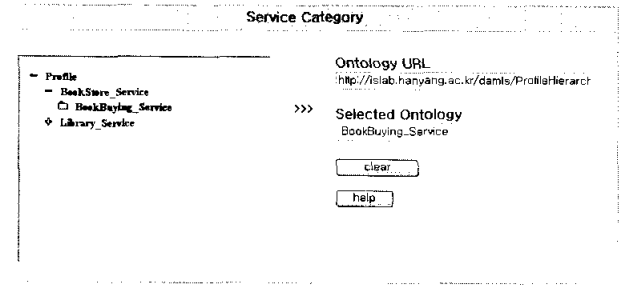
### 4.1 절차적 사용자 질의 인터페이스 구현 및 결과

〈표 3〉 DAML-S Profile과 논리 연산자의 실리적(Practical) 결합 표

| Type | Element | AND | OR | NOT | <,> |
|---|---|---|---|---|---|
| Function | input | Yes | No | Yes | No |
| | output | Yes | No | Yes | No |
| | effect | No | No | No | No |
| | precondition | No | No | No | No |
| Profile Attribute | serviceParameter | Yes | Yes | Yes | No |
| | QualityRating | Yes | No | No | Yes |
| | ServiceCategory | Yes | Yes | Yes | No |

본 절에서는 3.2절에서 정의한 설계에 따라 웹 환경에 맞게 구현한 절차적 사용자 인터페이스를 보여준다. 절차적 사용자 질의 인터페이스를 구현하기에 앞서 3.2절의 설계에 대해 몇 가지 고려 할 사항이 있다. 3.2절에서 정의한 표2의 논리 연산자와 DAML-S Profile의 결합 표는 사용자에게 일반 검색 엔진과 같은 불리안 모델을 제공하기 위해서 정의되었다. 하지만 3.2절에서 정의한 입력 요소와 논리 연산자의 관계성은 실세계의 의미와 호환되지 않는 것이 많다. 예를 들어, 어떤 사용자가 "QualityRating = Good or Quality-Rating = Bad"라는 질의를 한다고 가정하자. 이 질의를 표2의 논리적 규정에 적용하면, 어떤 문법적 잘못도 있지 않다. 하지만 이와 같은 질의는 상당히 무의미한 질의이기 때문에, 서비스 검색에서 불필요한 연산이 처리되어 검색 속도가 느려지게 된다. 그래서 본 절은 좀 더 사용자(환경) 친화적인 인터페이스 개발을 위해, 3.2절에서 정의한 논리적(Logical) 규정을 의미적 결합을 위한 실리적(Practical) 규정으로 변환하였다. <표 3>은 DAML-S Profile과 논리 연산자의 실리적 결합 표이다. 특히, OR 연산자를 제공하기 위해 다중 생성 프로시저를 자주 수행하게 되면, 결과 값을 반환하는데 상당한 시간이 소요되기 때문에, 꼭 필요한 경우가 아니라면 OR 연산자 기능을 제거하였다.

입력 화면은 정의 1의 입력 요소 절차에 의해 크게 세 가지 화면으로 구성되는데, 'ServiceCategory' 화면, 'service-Parameter'와 'QualityRating' 화면, 그리고 'Input/Output' 화




(그림 14) Category 인터페이스

면으로 사용자의 질의를 입력받는다. 모든 사용자 질의는 마우스 클릭을 통한 GUI형태로 구성되어 있기 때문에, 자신이 원하는 시맨틱 웹 서비스를 쉽게 찾을 수 있다.

<표 3>의 '프로파일 속성' 중 'ServiceCategory'는 업체에서 제공하는 서비스의 유형을 구분 짓기 위한 엔터티이다. 대표적으로, NAICS나 UNSPSC[20]의 단체에서 제공하고 있는 분류 기준을 사용해서 서비스의 유형을 기술하고 있다. 서비스를 제공하는 업체는 NAICS와 UNSPSC 중 하나만 또는 모두에 가입하여 서비스를 분류 받을 수 있다. 이것은 사용자 질의에서 모든 논리 연산자를 사용할 수 있지만, 각 인스턴스들(NAICS, UNSPSC)이 의미적으로 비교 대상이 아니기 때문에 비교 연산자를 사용할 수 없다. (그림 14)는 'Category' 인터페이스에 대한 실행 화면이다.

<표 3>의 '프로파일 속성' 중 'serviceParameter'는 Parameter에 따라 상당히 광범위 하게 이용될 수 있다. 본 논문에서는 'serviceParameter'로 'GeographicRadius'를 구현 예제로 사용하였다. 이것은 현재 서비스를 제공하는 지역에 대한 정보를 나타낸다. 각 지역에 대한 수치적 비교 정의가 되어 있지 않기 때문에 비교 연산자 사용은 불가능하다.

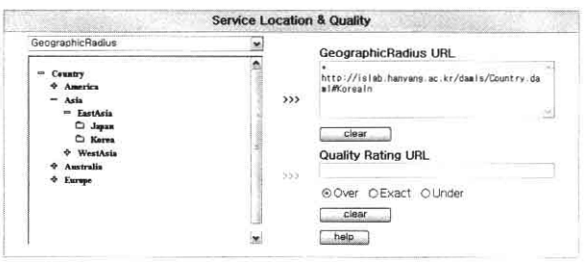
<표 3>의 '프로파일 속성' 중 'QualityRating'은 서비스의 상태를 표현하기 위한 엔터티로써, 의미적 모호성 때문에 상당한 복잡성을 요구한다. 먼저 'QualityRating'의 'NOT' 연산자는 의미적 모호성을 갖는다. 예를 들어, 만일 ①과 같은 'QualityRating' 집합(S)이 있다고 가정할 때, 'NOT'의 의미는 ② 또는 ③으로 해석될 수 있다.

① S ={'very good', 'good', 'bad', 'very bad'}

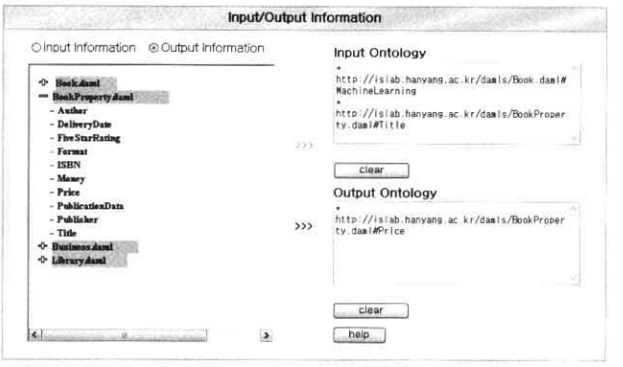
② NOT 'very good' = { 'good', 'bad', 'very bad' }

③ NOT 'very good' = { 'very bad' }

즉 사용자가 'NOT'의 의미에 대해 다양하게 해석할 수 있기 때문에, 웹 서비스의 검색 결과가 동일하지 않을 수 있다. 그래서 의미적 결합 표(표 3)에 'NOT'을 포함시키지 않았다. 또, 'QualityRating'는 다른 입력 요소와는 달리 비교 연산자를 사용할 수 있다. 이것은 사용자에게 어떤 상태 이상/이하 값을 갖는 서비스를 찾을 수 있도록 한다. 그리고 하나의 서비스가 두 가지 상태 값을 갖지 않기 때문에 'OR' 연산자를 사용하는 것은 불필요하다. (그림 15)는 'GeographicRadius'와 'QualityRating' 인터페이스에 대한 실행 화면이다. 'QualityRating'의 경우, 비교 연산자 기능을 적용할 수 있도록 (그림 15)의 오른쪽 패널에 'Over', 'Exact', 'Under'의 라디오 박스를 두었다.




(그림 15) GeographicRadius & QualityRating 인터페이스




(그림 16) Input/Output 인터페이스

절차적 사용자 질의 인터페이스는 사용자에게 질의에 해당하는 항목들을 모두 보여주기 때문에, 사용자는 필요한 항목만을 선택할 수 있도록 해야 한다. 즉, 'input'과 'output'에 대해 OR 연산자의 기능을 제공하지 않아도 사용자는 만족할 만한 서비스를 찾을 수 있다. 그리고 'effect'와 'precondition'은 서비스가 생성되거나 소멸됨으로써 얻어질 수 있는 작동이기 때문에[19], 서비스를 검색하려는 사용자 측면에서 불필요한 정보이다. 다시 말해서, 'effect'와 'precondition'은 에이전트 간의 신뢰성을 보장하기 위해 필요한 머신 독립적인 정보이다. 또, '함수' 정보는 비교 대상의 잣대가 없기 때문에 비교 연산자를 적용할 수 없다. (그림 16)은 Input/Output 인터페이스에서 대한 실행 화면을 나타낸다.

사용자에 의해 위의 세 단계 입력 요소가 모두 정의되면 (그림 17)과 같은 RDQL 문서가 생성될 수 있다. (그림 17)은 서비스 종류는 'BookBuying_Service', 서비스 지역은 'Korea', 서비스 상태는 'Average_Rating_C', 입력 인자는 'AI' 그리고 출력 인자는 'Title'인 서비스를 검색하라는 RDQL 질의를 나타낸다. 사용자 질의로부터 (그림 17)의 RDQL이 도출되기 위해서는 (그림 12)와 (그림 13)의 알고리즘이 필요하다.

```
SELECT ?x,
WHERE (?x, <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>,
      <http://islab.hanyang.ac.kr/damls/ProfileHierarchy.daml#BookBuying_Service>),
      (?x, <http://www.daml.org/services/daml-s/0.9/Profile.daml#serviceParameter>, ?a2),
      (?a2, <http://www.daml.org/services/daml-s/0.9/Profile.daml#sParameter>,
      <http://islab.hanyang.ac.kr/damls/Country.daml#Korealn>), (?x,
      <http://www.daml.org/services/daml-s/0.9/Profile.daml#qualityRating>, ?a3),
      (?a3, <http://www.daml.org/services/daml-s/0.9/Profile.daml#rating>,
      <http://islab.hanyang.ac.kr/damls/sRating.daml#Average_Rating_C>), (?x,
      <http://www.daml.org/services/daml-s/0.9/Profile.daml#input>, ?a0), (?a0,
      <http://www.daml.org/services/daml-s/0.9/Profile.daml#restrictedTo>,
      <http://islab.hanyang.ac.kr/damls/Book.daml#AI>), (?x,
      <http://www.daml.org/services/daml-s/0.9/Profile.daml#output>, ?a1), (?a1,
      <http://www.daml.org/services/daml-s/0.9/Profile.daml#restrictedTo>,
      <http://islab.hanyang.ac.kr/damls/BookProperty.daml#Title>)
```

(그림 17) RDQL 질의어(Result)

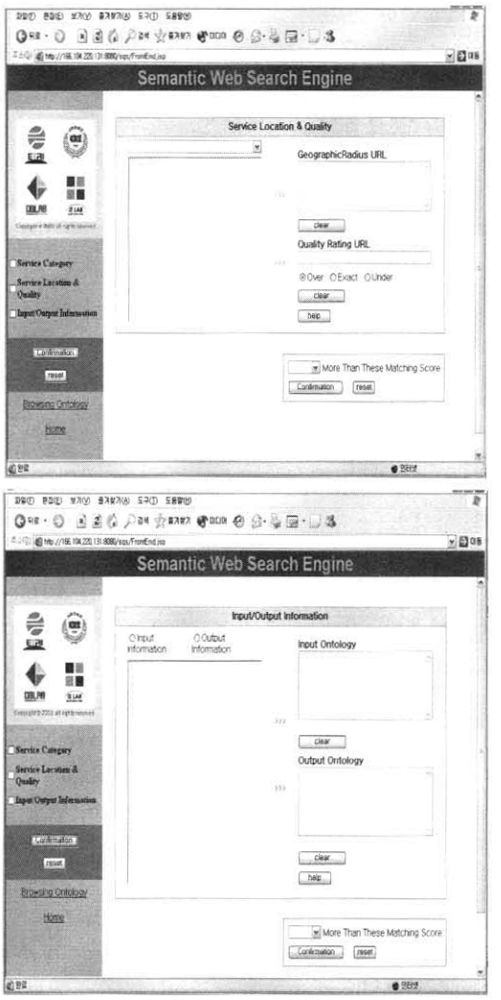
### 4.2 SQA 구현 및 결과

본 논문의 주요 특징은 일반 사용자에게 높은 시맨틱 질

의 투명성(Semantic Query Transparency)을 제공한다는 것이다. 즉, 일반 사용자가 시맨틱 질의 언어(RDQL)을 직접 사용하지 않고도 시맨틱 정보를 추출할 수 있도록 그래픽 기반의 질의 인터페이스(Query Interface)를 제시한다. 그래서 시맨틱 질의어를 처리하기 위한 추론 엔진, 질의어 처리기 등의 질의 프로세서의 기능은 제공하지 않는다고 가정한다. 하지만 SQA의 적합성을 판단하기 위해서는 질의를 추론할 수 있는 질의 프로세서가 필요하다. 그래서 본 논문은 Jena[19]에서 제공하는 RDQL 질의 프로세서를 이용하였다. 이것은 SQA로부터 입력받은 RDQL을 추론하여, 질의에 적합한 DAML-S Profile 문서를 추출한다. Jena는 Java로 온톨로지에 관련된 문서를 처리하기 위한 애플리케이션 프로그래밍 인터페이스(API)를 제공할 뿐만 아니라, OWL[4]의 추론인 포함(subsumption) 추론과 함의(entailment) 추론을 제공한다.

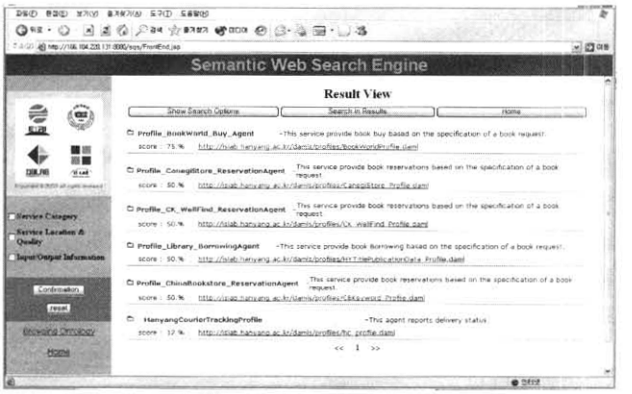
본 논문의 시스템은 모두 유니코드(Unicode)를 기반으로 하는 XML과 Java로 구성되었으며, 아래와 같은 단계로 수행된다.

1. (그림 18)의 절차적 사용자 질의 인터페이스에서 선택된 사용자 질의를 3.3절에서 설계한 질의어 생성기 컴포넌트에 의해 1단계 RDQL을 생성한다. 이때 (그림 12)의 알고리즘이 이용된다.
2. 만일 'serviceParameter'처럼 'OR' 연산자를 사용하는 질의가 선택되었을 때, 3.4절에서 설계한 시맨틱 질의어 다중 생성 프로시저에 의해 여러 개의 RDQL문을 생성한다. 이때 (그림 13)의 알고리즘이 이용된다. 여러 개의 RDQL을 생성할 때, 1단계 결과 RDQL을 덧붙여서 최종 RDQL을 생성하게 된다.
3. 최종 RDQL은 (그림 17)과 같은 구성을 갖게 된다. (그림 17)은 최종 RDQL들 중 하나만을 나타낸다.
4. 생성된 RDQL은 추론 엔진에 전해져 연관된 프로파일 문서의 유사도를 계산해서 반환한다. 반환된 결과는 (그림 19)처럼 보여지게 되는데, 각 문서의 유사도도 함께 보여준다.

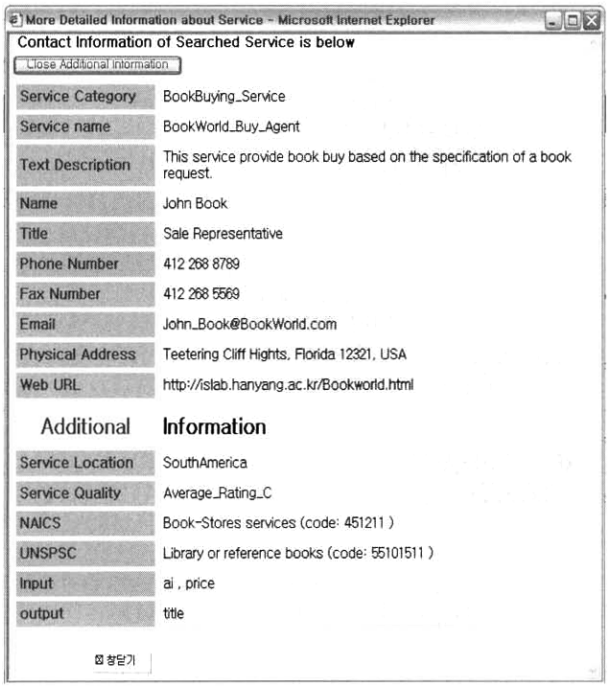

(그림 18) 절차적 사용자 질의 인터페이스

본 논문에서 데이터베이스에 미리 저장한 DAML-S Profile의 총 개수는 50개이다. (그림 19)는 서비스의 종류는 'BookBuying_Service', 서비스 지역은 'Korea', 입력 인자는 'Machine Learning, Title' 그리고 출력인자는 'Price'인 서비스를 검색하라는 질의에 대한 결과이다. (그림 20)은 각 서비스 프로파일의 모든 정보를 담고 있다. 특히, 출력 요소




(그림 19) 검색 결과

인 '조직' 정보를 (그림 20)의 상단에 보여준다. 이것은 가정 2에서 정의한 것처럼, 서비스를 검색한 유저가 '조직' 정보를 통해 웹 서비스를 제공하는 업체와 직접적으로 연결할 수 있게 하기 위한 것이다.




(그림 20) 세부적인 시맨틱 웹 서비스 정보

## 5. 결 론

시맨틱 웹에 대한 연구가 활발히 진행되면서, 시맨틱 웹의 활용분야가 점차 확대되고 있다. 특히 본 논문에서 대상으로 하고 있는 웹 서비스 분야는 이미 많은 시스템들로 개발되어 있고, 그만큼 충분한 사용자 계층을 확보하고 있다고 볼 수 있는 분야이다.

그러나 시맨틱 웹을 위한 사용자 인터페이스는 시맨틱 웹에 대해 상당한 지식을 보유한 사람들만을 위한 것이 대부분이다[22,23,24]. 시맨틱 언어나 시맨틱 질의어에 관한 이해를 갖지 못한 일반 사용자를 위해 인터페이스가 개발되지 않았다.

이 시스템을 통해 구축한 절차적 사용자 질의 인터페이스는 그러한 면에서 기존의 시스템[22,23,24]보다 진일보한 시스템이라고 할 수 있다. 그리고 생성된 RDQL은 XML을 기반으로 하기 때문에, SQA는 RDQL을 사용하는 어떤 추론 엔진에서도 조합될 수 있다. 이러한 특징뿐만 아니라 기존의 검색 시스템에서 제공한 논리 연산자 기능을 그래픽 인터페이스로 제공하기 때문에 일반 사용자에 높은 시맨틱 질의어 투명성(Query Transparency)을 제공한다. 즉, 일반 사용자가 시맨틱 질의 언어(RDQL)를 직접 사용하지 않고도 시맨틱 정보를 효과적으로 추출할 수 있다. 이런 특징들은 시맨틱 웹 서비스의 사용자층을 확장시키는 효과를 가져 올 뿐만 아니라, 시맨틱 웹이 기존의 웹과 잘 접합할 수 있는 계기를 마련할 것이다.

월드 와이드 웹 컨소시엄에서 OWL[4]을 시맨틱 언어로 추천함으로써, 이 언어를 통한 향후 발전이 있을 것이다. 본 논문의 향후 연구로 OWL-S의 절차적 사용자 질의 인터페이스를 구축하는 것이다. 또, RDQL 이외의 다른 시맨틱 질의어로의 변환을 가능하게 하는 시스템을 구현한다면 각 에이전트 간의 상당한 호환성을 이룰 것이며, 시맨틱 웹의 다양한 이용자(사용자, 에이전트 등) 층을 확보할 것이다.

## 참 고 문 헌

[1] T.Berners-Lee and M. Fischetti, "Weaving the Web : The Original Design and Ultimate Destiny of the World Wide Web by its Inventor", Harper San Francisco, ISBN: 006251586 , Oct. 1999.

[2] T. Gruber, "A translation approach to portable ontologies", Knowledge Acquisition, Vol. 5, No. 2, pp.199-220, 1993.

[3] Ian Horrocks, "DAML+OIL: a description logic for the semantic web", Bull. of the IEEE Computer Society Technical Committee on Data Engineering, Vol. 25, No.1, pp. 4-9, March 2002.

[4] Michael K, Smith , "OWL Web Ontology Language Guide" W3C Proposed Recommendation, Dec. 2003, http://www.w3.org/TR/2003/PR-owl-guide-2003101215/ (current. Dec. 2003).

[5] Gruber, T.R. "Ontolingua: A mechanism to Support Portable Ontologies". Stanford University, Knowledge Systems Laboratory, Technical Report KSL-91-66, March 1992. Revision.

[6] Ankolekar, A., Burstein, M., Hobbs, J.R., Lassila, O., Martin, D.L., McIlraithe, S.A., Narayanan, S., Paolucci, M., Payne, T.R., Sycara, K., and Zeng, H., "DAML-S: Semantic Markup for Web Services". International Semantic Web Workshop (SWWS), 2001.

[7] T. Sollazzo, S. Handschuh, S. Staab, M. Frank, "Semantic Web Service Architecture - Evolving Web Service Standards toward the Semantic Web", American Association for Artificial Intelligence, 2002.

[8] G. Orth, "The Web Services Framework: A Survey of WSDL, SOAP and UDDI", MSc Thesis, Information Systems Institute, Vienna University of Technology, May. 2002.

[9] O. Lassila and R. Webick, "Resource Description Framework (RDF) Model and Syntax Specification", W3C Recommendation, Jan. 1999, www.w3.org/TR/PR-rdf-syntax (current Feb. 2004).

[10] Dieter Fensel, "Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce", Springer Verlag, ISBN: 3540416021, Aug. 2001.

[11] Gomez-Perez, A. and O. Corcho, "Ontology Languages for the Semantic Web.", IEEE Intelligent Systems, Vol.17, No. 1, pp. 54-60, 2002.

[12] D. Brickley and R.V Guha, "RDF Vocabulary Description Language 1.0: RDF Schema", W3C Recommendation, Feb. 2004, www.w3.org/TR/rdf-schema

[13] J. Clark and S. DeRose, "XML Path Language (XPath) Version 1.0", W3C Recommendation, Nov. 1999, www.w3.org/TR/xpath

[14] S. Boag, D. Chamberlin, M. F. Fernandez, J. Robie, J. Simeon, "XQuery 1.0: An XML Query Language", W3C Working Draft, Nov. 2003, www.w3c.org/TR/xquery

[15] G. Karvounarakis, S. Alexaki, V. Christophides, D. Plexousakis, M. Scholl, "Querying community web portals", Technical Report, Institute of Computer Science, FORTH, Heraklion, Greece. See www.ics.forth.qr/proj/isst/RDF/RQL/rql.pdf

[16] A. Seaborne, HP Labs Bristol, "RDQL - A Query Language for RDF", W3C Member Submission, Jan. 2004, www.w3.org/Submission/RDQL

[17] R. Fikes, P. Hayes, I. Horrocks, "DAML Query Language (DQL)", DAML Joint Committee, Apr. 2003, www.daml.org/2003/04/dql/dql

[18] J. Broekstra, A. Kampman and F. van Harmelen. "Sesame: An Architecture for Storing and Querying RDF Data and Schema Information", In: D. Fensel, J. Hendler, H. Lieberman and W. Wahlster (eds.) Semantics for the WWW.MIT Press, 2001.

[19] Jena : www.hpl.hp.com/semweb/jena2.htm

[20] DAML-S Home page : www.daml.org/services, 2002.

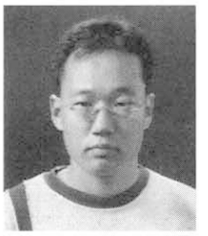
[21] S. Jhones, "VQuery: a Graphical User Interface for Boolean Query Specification and Dynamic Result Preview", Working Paper 98/3, March. 1998.

[22] Ralf Möller, Ronald Cornet, Volker Haarslev, "Graphical Interface for Racer: Querying DaML+OIL and RDF documents", Proceedings of the International Workshop on Description Logics (DL-2003), Rome, Italy, pp.255-259, Sep. 2003.

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

[24] Arpírez, J. C. Corcho, O. Fernández-López, M. Gómez-Pérez, "A. WebODE: a Workbench for Ontological Engineering", First International Conference on Knowledge Capture (K-CAP'01). Victoria B. C. (Canada). 2001.

[25] F. Baader, W. Nutt, "Basic Description Logics", Cambridge University Press, pp. 47-100, 2002.



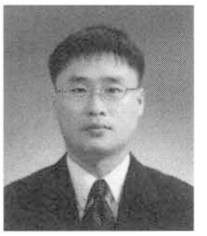
**조 명 현**

e-mail : mhjo@cse.hanyang.ac.kr

2004년 한양대학교 전자컴퓨터공학부 학사

2004년~현재 한양대학교 컴퓨터공학과 석사과정

관심분야 : 그리드 컴퓨팅, 센서 네트워크, 시멘틱 웹, e-비즈니스



**손 진 현**

e-mail : jhson@cse.hanyang.ac.kr

1996년 서강대학교 전산학과 학사.

1998년 한국과학기술원 전산학과 석사.

2001년 한국과학기술원 전자전산학과 박사

2001년 9월~2002년 8월 한국과학기술원 전자전산학과 박사후 연구원.

2002년 9월~현재 한양대학교 컴퓨터공학과 조교수.

관심분야 : 데이타베이스, e-비즈니스, 유비쿼터스 컴퓨팅, 임베디드 시스템