URC를 위한 시맨틱 검색 및 서비스 프로세스 실행 기술

Semantic Service Discovery and Service Process Execution Technology for Ubiquitous Robotics Companion

지능형 로봇 특집

송병열 (B.Y. Song)	지능형서비스플랫폼연구팀 선임연구원
김경일 (K.I. Kim)	지능형서비스플랫폼연구팀 연구원
정승우 (S.W. Jung)	지능형서비스플랫폼연구팀 기술원
정문영 (M.Y. Jung)	지능형서비스플랫폼연구팀 연구원
김록원 (R.W. Kim)	지능형서비스플랫폼연구팀 연구원
문진영 (J.Y. Moon)	지능형서비스플랫폼연구팀 연구원
이대하 (D.H. Lee)	지능형서비스플랫폼연구팀 연구원
김연준 (Y.J. Kim)	지능형서비스플랫폼연구팀 기술원
조현성 (H.S. Cho)	지능형서비스플랫폼연구팀 책임연구원, 팀장

목 차

하드웨어 중심의 기존 로봇에 웹서비스 및 기술을 적용하여 로봇의 응용 서비스 영역을 확장함으로써 로봇의 정보 서비스 고도화를 이루기 위해 웹 기반 IT 로봇 서비스 enabling 기술을 개발하고 있다. 시맨틱 검색 기술은 사용자가 원하는 서비스 또는 콘텐츠를 비교적 정확히 찾기 위해 시맨틱 웹을 사용하고 있으며 서비스 프로세스 실행 기술은 인터넷 상의 웹서비스와 로봇의 구동 서비스를 하나의 프로세스로 동적으로 구성하여 수행할 수 있도록 해준다. 로봇은 방대한 양의 시맨틱 정보를 내장하고 않고 원격지 서버상에서의 시맨틱 검색을 통해 충분히 시맨틱한 결과를 획득할 수 있으며 또한 고정되지 않은 그리고 웹이 컨텍스트와 콘텐츠를 활용할 수 있는 서비스 프로세스 형태의 로봇이 제공할 수 있는 서비스의 질을 개선하고 그 종류 또한 다양하게 변화시킬 수 있다.

Ⅰ. 서론

현재까지 보급된 로봇들은 산업현장에서 사용되기 위한 것들이 대부분이었고 실제 생활에서 활용할 수 있는 홈로봇은 거의 존재하지 않았다고 볼 수 있다. 물론, 이것은 그 동안 로봇 제작에 드는 비용이 상당히 크고 또한 그 가격에 걸맞은 역할을 가정에서 찾기 힘든 면도 있겠다. 그러나, 최근 반도체 기술의 진보 및 기타 다른 기술의 진보로 인해 로봇이 필요로 하는 컴퓨팅 파워를 적은 가격에 해결할 수 있게 되면서 실제 로봇을 제작하는 데 드는 비용이 상당히 감소되었고 생활 수준의 향상으로 인한 소비자들의 서비스에 대한 질적 욕구가 점차 높아지면서 홈로봇에 대한 인식도 점차 높아지게 되었다.

최근, Sony는 Aibo라는 엔터테인먼트를 위한 애완견 로봇을 출시했었고 출시 당시 세상의 주목을 받으면서 많은 수가 팔리기도 했었다. 그러나, 오래지 않아 Aibo에 대한 수요가 다소 줄었고 기능이 상당히 향상되어 사용자의 얼굴 인식과 음성 인식이 강화된 현재의 버전에서도 일반 소비자들에게 충분히 가깝게 다가가지는 못하고 있다.

혼다의 Asimo는 직립보행을 하고 계단을 오르는 등 인간과 유사한 행동을 보일 수 있는 놀라운 기술적인 진보를 보여주었다. 그러나, Asimo 연구는 휴머노이드, 즉 인간 행동의 재현에 중심이 있는 만큼 홈 서비스와 관계된 부분은 발표되지 않았고 가격 면에서도 대단히 고가여서 홈로봇으로 사용되기에는 아직 많은 시간이 필요할 것이다.

IRobot사의 로봇 청소기 "룸바"는 그러한 점에서 보면 가정에서의 로봇의 역할에 상당히 근접한 것으로 보여진다. 가정에서 가사의 상당시간이 청소를 위해 쓰여진다는 사실에서 착안하여 제작된 이 로봇은 출시 당시부터 상당한 반향을 일으켜 세계의 여러 전자 벤더들이 앞다투어 유사한 기능을 가진 로봇 청소기들을 출시하도록 하였다. 물론 가격적인 면에서도 일반 고급 청소기와 비교할 때 비슷한 정도로 전세계적으로 수요가 지속적으로 증가될 것으로 예견된다. 이로써 드디어 본격적인 홈로봇 시장이 열린 것이라 하겠다.

그런데, 과연 현재 로봇이 가정에서 얼마나 많은 역할, 특히 물리적인 역할을 할 수 있을 것인가를 살펴보면 현재의 로봇 기술 수준으로는 그다지 많지 않다는 것을 알 수 있다. 예를 들어, 청소의 경우는 앞서 말했듯이 청소 로봇으로 어느 정도 해결이 되지만 그것 역시 분명히 제약이 존재한다. 그 외의 경우 요리나 세탁 등은 여전히 많은 부분 사람의 개입이 필요하거나 굳이 로봇이 필요치 않은 부분이 존재하며 방범(防犯)이나 방재(防災)의 경우 어느 정도 효용성이 있지만 그것만으로는 로봇에 대한 구매 욕구를 지속적으로 불러일으킬 수 있을지는 다소 미진한 감이 없지 않다. 결국 현재의 단순한 물리적인 기능만으로는 고객이 요구하는 수준의 서비스 질을 만족하지 못한다는 것이다.

최근 이러한 서비스 질을 향상시키기 위한 대안으로 네트워크를 통한 정보 서비스를 로봇에서 이용하도록 하는 방안인 URC가 로봇 관련 연구의 중요한 테마로 주목 받기 시작했으며 본 연구 또한 그러한 관점에서 시작되었다.

본 연구에서는 많은 인터넷 관련 기술 중에서 특히 웹서비스를 기반에 둔 비즈니스 프로세스 기술과 시맨틱 웹을 활용하는 저장 및 검색 기술의 효용성에 관심을 두고 이를 URC를 위한 정보 서비스 도구로 사용하였다.

Ⅱ. 관련 기술

이 절에서는 본 연구에서 사용되는 인터넷 관련 기술들에 대해서 기술하고 관련 기술의 로봇에서의 효용성에 대해 기술한다.

1. 웹서비스

웹서비스(web service)는 인터넷상의 서비스 제공자와 사용자간의 통신을 위한 표준 프레임워크를 말하는 것으로 기존의 다른 기술들처럼 완벽한 상호작동 사양을 정의한 것이 아닌 서로 주고 받는

데이터 표준에 대한 정의를 규정하여 서로 다른 운영 시스템, 구현 언어를 가진 이기종 시스템들을 유연하게 통합할 수 있도록 한다. 웹서비스가 가지는 중요한 특징의 하나는 비교적 적은 비용으로 서로 다른 운영체제와 구현 언어를 가진 기존 시스템들을 유연하게 통합할 수 있다는 것에 있다. 또한, XML 기반 표준을 사용하여 확장성이 매우 높고 구조가 단순하여 개발 비용이 비교적 적게 드는 것이 웹서비스가 현재 주목 받고 있는 이유의 하나이다.

현재 웹서비스에 관한 표준은 W3C에서 개발되고 있으며 SOAP, WSDL, UDDI의 3가지 표준으로부터 구성된다.

SOAP은 분산 환경 하에서 정보를 교환하기 위한 통신 프로토콜로서 인터넷을 통해 서비스 제공자와 사용자가 다양한 정보를 XML 메시지 형태로 교환할 수 있도록 한다. SOAP은 현재 기본 전송 프로토콜로 HTTP를 사용하던 SOAP1.0을 개선하여 메시징 시스템과 데이터 표현 방식으로 분리하여 전송 프로토콜에 독립적인 SOAP1.2가 제안되었다[1].

SOAP은 순수 콘텐츠 통신을 지원하는 것으로 이를 사용하기 위해서는 콘텐츠를 설명하는 언어가 필요하다. SOAP 메시지가 유형에 관한 정보를 전달하기 때문에 SOAP 내부에서 정보의 유형을 동적으로 결정할 수 있지만 서비스 제공자가 제공하는 서비스에 대한 함수 이름, 매개 변수의 개수, 각 매개 변수의 유형을 알지 못한다면 제대로 서비스를 사용할 수 없다. 따라서, 서비스가 제공하는 함수에 관한 정확한 정보를 제공할 필요가 있으며 이를 위해 개발된 것이 WSDL이다. WSDL은 웹서비스 구축에서 사용자가 실제로 접하는 요소로 WSDL을 통해 자동화된 방법으로 서버와 클라이언트간의 통신을 담당하는 프록시를 생성할 수 있다. WSDL은 현재 SOAP1.2를 지원하는 2.0 버전이 W3C에서 발표되었다[2].

UDDI는 웹서비스를 위한 웹 기반 분산 레지스트리에 관한 표준으로 서비스 제공자들이 제공하는 웹서비스에 대한 정보를 등록하고 이를 사용자가 검색할 수 있도록 하는 프레임워크와 인터페이스를 정의한다. 현재 산업계 표준화 단체인 OASIS(www.oasis-open.org)가 UDDI 프로젝트와 활동을 위한 사무국 역할을 하면서 표준 개발을 지속적으로 추진하고 있으며 최근 보안 관련 부분이 강화된 UDDI3.0이 표준으로 확정되었다[3].

2. 시맨틱 웹

팀 버너스 리가 제안한 시맨틱 웹(semantic web)은 인간과 컴퓨터간의 소통에 치중한 현 단계의 웹과는 달리 컴퓨터와 컴퓨터간의 자유로운 정보 전달에 중점을 둔 새로운 웹 시스템을 가리킨다. 현재의 컴퓨터가 웹에 담겨 있는 방대한 양의 정보를 단순히 문자의 집합으로 받아 들인다면, 시맨틱 웹은 각 단어와 문장의 의미들을 컴퓨터가 파악할 수 있도록 하는 것이 목표이다. 현재 데이터베이스 등 컴퓨터마다 다른 의미를 갖는 데이터를 다양한 컴퓨터에서 사용하기 위해서는, 일일이 사람이 손으로 데이터를 변환해주는 작업을 거쳐야 하나 시맨틱 웹이 성공할 경우 이러한 복잡한 과정이 모두 생략되고, 어떤 종류의 컴퓨터나 응용 프로그램도 자유롭게 데이터에 접근할 수 있게 된다.

시맨틱 웹에서 문서의 각 부분을 컴퓨터가 이해할 수 있는 형식으로 작성한다면 복잡하게 얽혀져 있는 정보 리소스들 사이의 의미적 연관성으로 인해 웹을 통해 다양한 정보를 효과적으로 활용할 수 있게 되며 이렇게 정보 리소스들 사이의 연관성을 작성하기 위해 언어 및 기반구조가 필요하다.

현재 시맨틱 웹과 관련하여 크게 두 개의 표준이 작성되었는데 웹의 표준을 담당하는 W3C에서는 RDF 언어를 기반으로 하는 온톨로지 기술을 개발하였고 ISO에서는 XTM 언어를 기반으로 하는 topic maps 기술을 개발하였다.

W3C의 시맨틱 웹 기술은 크게 RDF, RDFS, OWL로 구성되는데 이중 가장 기초가 되는 것은 RDF로 특정자원에 대한 메타데이터를 XML 기반으로 기술할 수 있는 프레임워크를 제공한다. RDF는 (그림 1)과 같은 "Triple"을 기본 개념으로 가지

Subject(Resource)	http://www.w3.org/Home/Lassila
Predicate(Property)	Creator
Object(Literal)	“Ora Lassila”

(그림 1) RDF의 Triple 구조

고 있으며 자원들 간의 관계 설정이 속성(predicate)를 통해 무한으로 가능하게 하여 자원의 시맨틱에 관한 지식 네트워크를 구성할 수 있게 한다.

RDFS는 XML에서의 XMLS와 같은 역할로 RDF 클래스와 클래스 간의 관계, 속성과 속성간의 관계 등을 정의하여 사람이 이해하는 동시에 기계처리가 가능한 형태로 메타데이터의 속성과 클래스 간의 관계의 표현이 가능하게 한다. OWL과 같은 온톨로지 언어는 용어 사이의 관계를 정의하고 있는 일종의 사전을 작성할 수 있게 하여 RDF 등을 통해 각 정보 리소스들의 속성과 연관관계를 정의함으로써 논리적인 추론을 수행할 수 있는 근거를 마련해 준다.

Topic maps는 ISO/IEC 13250 표준으로 지식 표현 기술(knowledge representation)의 표준이다. RDF와 마찬가지로 XML 기반의 표준 기술 언어인 XTM라는 언어를 사용하여 정보와 지식의 분산 관리를 지원한다. Topic maps는 지식층과 정보층의 이중 구조를 나타내는데, 지식층은 상위 계층으로 토픽과 토픽간의 연계(association)로 구성된다. 토픽은 특정 주제를 나타내는 표현이고, 연계는 주제들 간의 관계를 나타내는 표현이다. 정보층은 디지털 콘텐츠를 나타내며, 이들 지식층과 정보층은 어커런스(occurrence)를 통해 상호 연결이 이루어진다.

URC에서 시맨틱 웹의 가치는 기계가 이해할 수 있는 정보의 표현이 가능하다는 것에 있다. 이것은 시맨틱 웹을 통해 환경에 대한 정보를 얻기가 용이하고 추론을 통해 지능적인 정보 처리 작업이 가능하다는 것을 의미한다. 따라서, 유비쿼터스 환경에서 동작하며 지능적인 작업을 요구하는 URC에 있어서 시맨틱 웹은 매우 중요한 문제 해결 도구가 될 수 있다.

3. 시맨틱 웹 서비스

로봇에게 많은 서비스를 제공한다는 관점에서 보면 웹상에 이미 존재하는 기존의 웹서비스를 사용한다는 것은 매우 유용하다. 그러나, UDDI, WSDL, SOAP과 같은 현재의 기술은 서비스 검색이나 서비스 프로세스, 서비스 비교, 자동화된 협상 등의 구현에 있어서는 한계를 지니고 있다. 이들 프로토콜들이 사람이 이해하고 사용하기 용이하도록 설계되었기 때문에 로봇이 어떤 미션을 수행하기 위해 적절한 서비스를 자동으로 찾기에는 무리가 있다. 따라서, 그러한 서비스들을 컴퓨터가 인식할 수 있도록 하는 기술, 즉 서비스의 프로퍼티, 수행 능력, 인터페이스, 효과 등을 기계가 이해할 수 있는 형태로 인코딩하는 서비스에 관한 시맨틱을 만들 필요가 있다.

시맨틱 웹서비스는 웹서비스에 대한 시맨틱을 제공하고 웹서비스 검색에 관한 프레임워크를 정의하여 웹상의 콘텐츠를 machine-processable하고 machine-interpretable하게 한다. OWL-S는 시맨틱 웹서비스를 구현하기 위한 핵심적인 컴포넌트로 DARPA에 의해 개발된 DAML+OIL 기반의 서비스 기술을 위한 온톨로지 언어이며, 2001년 5월에 0.5 버전을 발표한 후, 2003년 5월에 0.9 베타 버전이 발표되었다. 이후 W3C에서 DAML-S 0.9 베타 버전을 이어받아 OWL-S로 명칭을 변경하였으며 현재 OWL-S 1.1버전을 제공하고 있다. OWL-S는 웹서비스의 탐색(discovery), 합성(composition) 등을 에이전트의 추론 기능을 이용하여 지능적이면서 자율적으로 서비스를 구성할 수 있는 웹서비스 기술 방법을 제공한다[13].

4. 웹서비스 비즈니스 프로세스

로봇이 어떤 일(work or task)을 한다고 할 경우 관련 일이 사전에 프로그래밍되어 있다면 로봇은

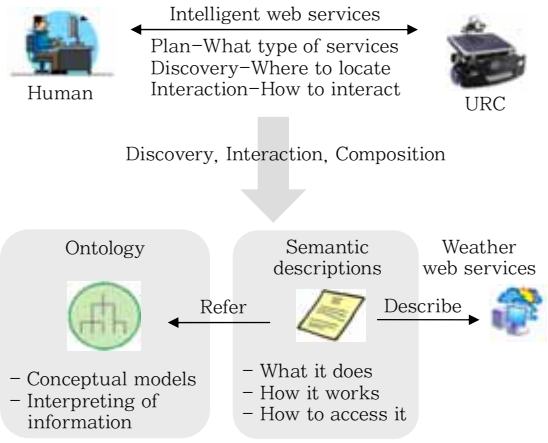
그대로 수행할 것이다. 그러나, 미리 할 일이 정해져 있는 사업현장의 로봇과 달리 생활에서 사용되어야 할 URC와 같은 로봇은 필요한 모든 기능을 예상하여 미리 프로그래밍 한다는 것은 상당한 노력을 요구하며 환경의 변화에 대처하기 매우 어렵다. 그런데 이것은 인터넷상에서의 서비스에 대해서도 같은 문제점으로 등장하기 시작하였다. 각각의 서비스에 대한 시스템을 만들었으나 이들은 서비스들을 활용하여 새로운 서비스를 조합하기 위해 새로 시스템을 구축할 필요 없는 방법이 필요하게 된 것이다. 웹서비스 기반의 비즈니스 프로세스 기술은 이를 위해 XML의 형태로 기존의 웹서비스들 통합하여 새로운 서비스를 작성할 수 있게 한다.

웹서비스 기반의 비즈니스 프로세스 기술은 BPEL4WS[16]가 대표하는 워크플로의 rule 기반 자동화의 뿌리를 반영한 웹서비스 조화(orchestration) 기술과 WSCI[17]가 대표하는 워크플로의 인간 기반 뿌리를 반영한 웹서비스 구성(choreography) 기술로 구분되어 등장하였다. BPEL4WS는 실행 가능한 비즈니스 프로세스의 생성에 중점을 두고 있지만 WSCI는 웹서비스 간의 메시지 교환에 중점을 두고 있다.

Ⅲ. URC를 위한 지능형 서비스 플랫폼

본 연구는 로봇이 내재한 단편적이고 획일화된 취약한 정보를 웹상의 방대한 정보로 확장 가능하게 하고 다양한 사용자의 서비스 요구에 즉각 대응할 수 있도록 하기 위하여 먼저 웹서비스를 시맨틱하게 마크업하는 기술과 이를 저장하기 위한 레지스트리 및 시맨틱 탐색 기술, 검색된 웹서비스를 프로세스에 따라 실행하는 서비스 실행 엔진 기술로 구성된다.

본 연구의 결과가 활용되는 URC 환경은 URC 서버와 URC 클라이언트로 구성된다. URC 클라이언트는 통신 모듈을 갖추고 인터넷에 연결된 PDA, 이동 전화, 노트북 등의 디바이스를 기반으로 음성



(그림 2) 지능형 서비스 플랫폼의 개념

인식 모듈, 영상 인식 모듈, 항행 모듈과 같은 몇 가지 컴포넌트가 추가된 간단한 로봇 시스템을 대상으로 한다. URC 서버는 원격지에서 로봇 클라이언트의 기능 및 해동을 통제하는 시스템으로 시맨틱 웹서비스 레지스트리와 서비스 프로세스 실행 엔진이 포함될 수 있다. (그림 2)는 URC와 연동된 지능형 서비스 플랫폼에 대한 개념을 표현하고 있다.

URC는 다음 단계를 따라 로봇 서비스를 제공한다.

1) 사용자는 URC에게 요구한다.
2) URC는 서비스 온톨로지를 사용해서 그 요구를 분석한다.
3) URC 서비스 플래너는 시맨틱 탐색 레지스트리에 접속하여, 사용자의 요구를 처리하기에 적합한 서비스를 발견한다.
4) 대부분의 적당한 서비스는 URC 프로세스 실행 엔진으로 실행된다.
5) 실행 결과는 URC로 반환된다.

1. 웹서비스의 시맨틱 마크업

웹서비스에 대한 시맨틱 마크업은 서비스 레지스트리나 검색 엔진이 자동으로 사용자의 요구에 맞는 서비스를 찾을 수 있기 위해 필요한 정보를 제공한다. 웹서비스에 대한 시맨틱 마크업으로 본 연구에

서 채택한 것은 OWL-S로 서비스와 관련된 온톨로지 표준으로 가장 일반적이다.

OWL-S는 각각의 서비스들이 제공하는 기능을 기반으로 웹서비스의 위치를 찾을 수 있기 위한 솔루션으로 웹서비스 탐색의 자동화를 위해 프로퍼티는 자동화된 서비스 분류 및 선택에 따라 서비스와 연결하고 웹서비스 실행 자동화를 위해 마크업은 컴퓨터 에이전트가 자동적으로 웹서비스 요청을 구축하여 실행하고, 서비스의 응답을 해석하는 것을 가능하게 해야만 한다. (그림 3)은 OWL-S로 기술된, 램프를 켜고 끄는 전등 스위치 서비스에 관한 시맨틱 웹서비스 마크업의 예이다.

```
- <service: Service rdf: ID="LightControlService">
    <service: presents rdf: resource="#LightControlServiceProcessModel" />
    <service: describedBy rdf: resource="#LightControlServiceProcessModel" />
    <service: supports rdf: resource="#LightControlServiceGrounding" />
  </service:Service>
- <profile:Profile rdf:ID="LighControlServiceProfile">
    <service:presentedBy rdf:resource="#LightControlService" />
    <profile:has_process rdf:resource="#LightControlServiceProcessModel" />
    <profile:serviceName>LightControlService</profile:serviceName>
    <rdfs:label>LightControlService</rdfs:label>
    <profile:textDescription>Control light on the Demo-Room</profile:textDescription>
    <profile:hasInput rdf:resource="#Structure" />
    <profile:hasEffect rdf:resource="http://urcsp.etri.re.kr/concepts/2004/10/Behavior#Toggle/" />
  </profile:Profile>
- <process:ProcessModel rdf:ID="LightControlServiceProcessModel">
    <service:describes rdf:resource="#LightControlService" />
  - <process:hasProcess>
    - <process:AtomicProcess rdf:ID="LightControlServiceProcess">
      - <process:hasInput>
        - <process:Input rdf:ID="Structure">
            <process:parameterType
              rdf:resource="http://urcsp.etri.re.kr/concepts/2004/10/HouseStructure#Structure/>
          </process:Input>
        </process:hasInput>
        <process:hasEffect rdf:resource="http://urcsp.etri.re.kr/concepts/2004/10/Behavior#Toggle" />
      </process:AtomicProcess>
    </process:hasProcess>
  </process:ProcessModel>
- <grounding:WsdlGrounding rdf:ID="LightControlServiceGrounding">
    <service:supportedBy rdf:resource="#LightControlService" />
  - <grounding:hasAtomicProcessGrounding>
    - <grounding:WsdlAtomicProcessGrounding rdf:ID="ToggleServiceGrounding">
        <grounding:owlsProcess rdf:resource="#LightControlServiceProcess" />
        <grounding:wsdlDocument>http://129.254.164.143:8080/axis/services/LightControlServiceService?
          wsdl</grounding:wsdlDocument>
      - <grounding:wsdlOperation>
        - <grounding:WsdlOperationRef>
            <grounding:portType>LightControlServicePort</grounding:portType>
            <grounding:operation>toggleLight</grounding:operation>
          </grounding:WsdlOperationRef>
        </grounding:wsdlOperation>
      - <grounding:wsdlInputMessageParts rdf:parseType="Collection">
        - <grounding:wsdlMessageMap>
            <grounding:owlsParameter rdf:resource="#Structure" />
            <grounding:wsdlMessagePart>structure</gronding:wsdlMessagePart>
          </grounding:wsdlMessageMap>
        </grounding:wsdlInputMessageParts>
        <grounding:wsdlOutputMessage>toggleLightResponse</grounding:wsdlOutputMessage>
        <grounding:wsdlInputMessage>toggleLightRequest</grounding:wsdlIntputMessage>
      </grounding:WsdlAtomicProcessGrounding>
    </grounding hasAtomicProcessGrounding>
  </grounding:WsdlGrounding>
```

(그림 3) 전등 스위치 서비스의 시맨틱 마크업

2. URC 서비스 탐색

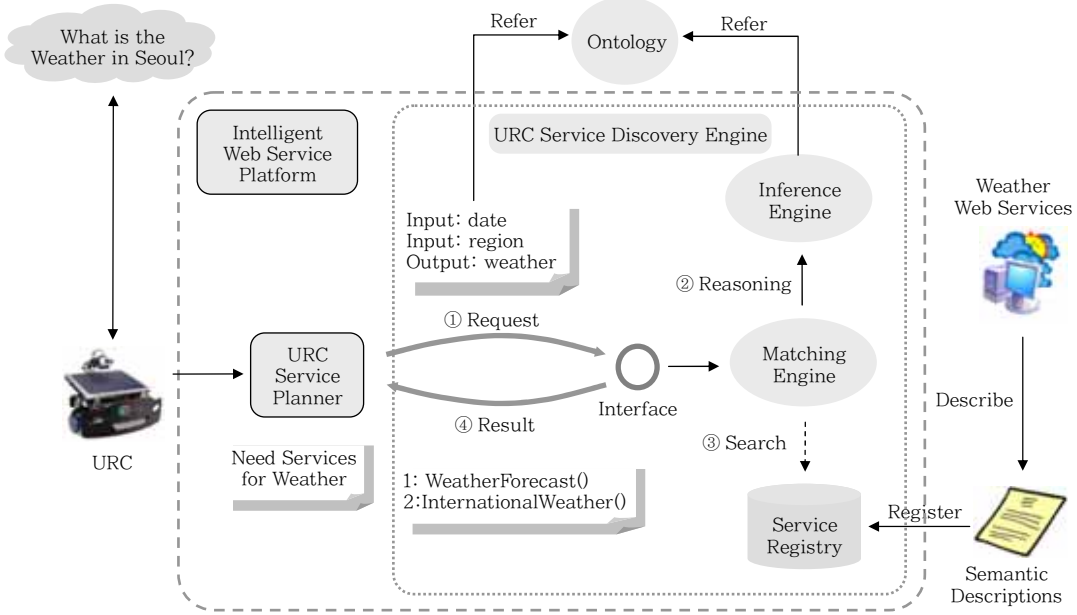
현재의 정보 검색 기술들은 검색의 결과가 인간에게만 의미가 있는 텍스트 데이터로 기계가 그 내용을 이해하고 처리하는 것이 불가능하며 따라서 로봇이 스스로 웹서비스를 검색하고 그 결과를 이해하여 사용자의 요구를 처리할 수 있는 방법이 존재하지 않는다.

URC 서비스 탐색 기술은 로봇이 사용자의 요청 또는 명령을 받아 지능적으로 태스크를 처리하기 위해 인터넷이나 로봇 외부에서 제공하는 다양한 서비스들을 시맨틱 기반 검색기법을 통해 탐색하는 방법을 제공한다.

인터넷상의 웹서비스들은 OWL-S로 기술되며 이렇게 기술된 정보들은 URC 서비스 레지스트리에 등록된다. 사용자의 요구가 있을 때 서비스 탐색 엔진은 OWL-S에서 정의된 IOPE로 표현된 수행 능력을 기반으로 사용자의 요구에 가장 적합한 웹서비스를 찾는다. 만일 검색결과가 여러 개의 웹서비스일 경우 랭킹 알고리듬을 사용하여 결과를 정렬한다. (그림 4)는 URC 서비스 탐색 엔진의 기능을 표시한다.

가. 검색 인터페이스

URC와 상호 작용하기 위해, URC 서비스 탐색 엔진은 질의어로 OWL-QL을 사용하고 통신 프로토콜로 SOAP을 사용한다. OWL-QL은 W3C의 OWL으로 표현된 지식을 사용하여 에이전트간에 질의/응답하는 프로토콜이다.



(그림 4) URC 서비스 탐색

```
(and
(|http://www.daml.org/services/owl-s/1.0/Process.owl#|::|parameterType|?w1
|http://urcsp.etri.re.kr/concepts/2004/10/HouseStructure.owl#|::|Structure|)
(|http://www.daml.org/services/owl-s/1.0/Profile.owl#|::|hasInput|?w2 ?w1)
)
```

(그림 5) 전등 스위치 서비스를 탐색하기 위한 KIF의 예

OWL-QL 질의는 매칭 엔진에서 사용되는 KIF 질의를 수반한다. URC 질의는 KIF 질의로 바뀌는데, 그 예가 (그림 5)에 나타난다.

나. 매칭 엔진

매칭 엔진이 URC 또는 또 다른 소프트웨어 에이전트로부터 질의를 받을 때, 그것은 착신 요구를 충족하기 위해 그것의 저장소를 검색한다. 어떤 웹서비스의 시맨틱 마크업이 URC로부터의 요청에 매칭될 때, 매칭 엔진은 그 요청에 대하여 그 서비스를 반환한다. 예를 들어, (그림 5)의 요청에 대해 매칭 엔진은 레지스트리에서 입력 타입이 "http://urcsp.etri.re.kr/concepts/2004/10/Hous-Structure.owl#Structure."인 시맨틱 웹서비스를 찾는다.

해당 요구에 대해 질의 처리기는 요구 질의를 확장하여 질의의 데이터 유형을 위한 계층구조 온톨로지를 생성한다. 이 과정에서는 서비스의 데이터 유형이 요청된 질의의 그것들과 정확하게 일치하지는 않지만 요청된 질의의 데이터 유형의 조상 및 자손인 더 많은 서비스를 찾아내는 것이 요구된다. 다음으로 질의 처리기는 요청 KIF 질의를 상응하는 SQL 질의로 변환한다. 레지스트리 관리 모듈은 적합한 웹서비스에 대한 시맨틱 마크업을 얻기 위해 생성된 SQL 질의를 사용하여 저장소를 검색한다. 끝으로, 검색된 결과가 여러 개인 경우 랭커는 검색된 결과상의 각각의 웹서비스와 확장된 계층구조 온톨로지 사이의 유사도를 측정하고 각 웹서비스에 대한 순위를 매긴다. (그림 6)은 이러한 과정을 통해 나온 매칭의 결과에 관한 예이다.

(그림 7)은 웹서비스의 시맨틱 탐색의 총체적인 흐름과 요구부터 응답까지의 데이터 흐름을 나타내고 있다.

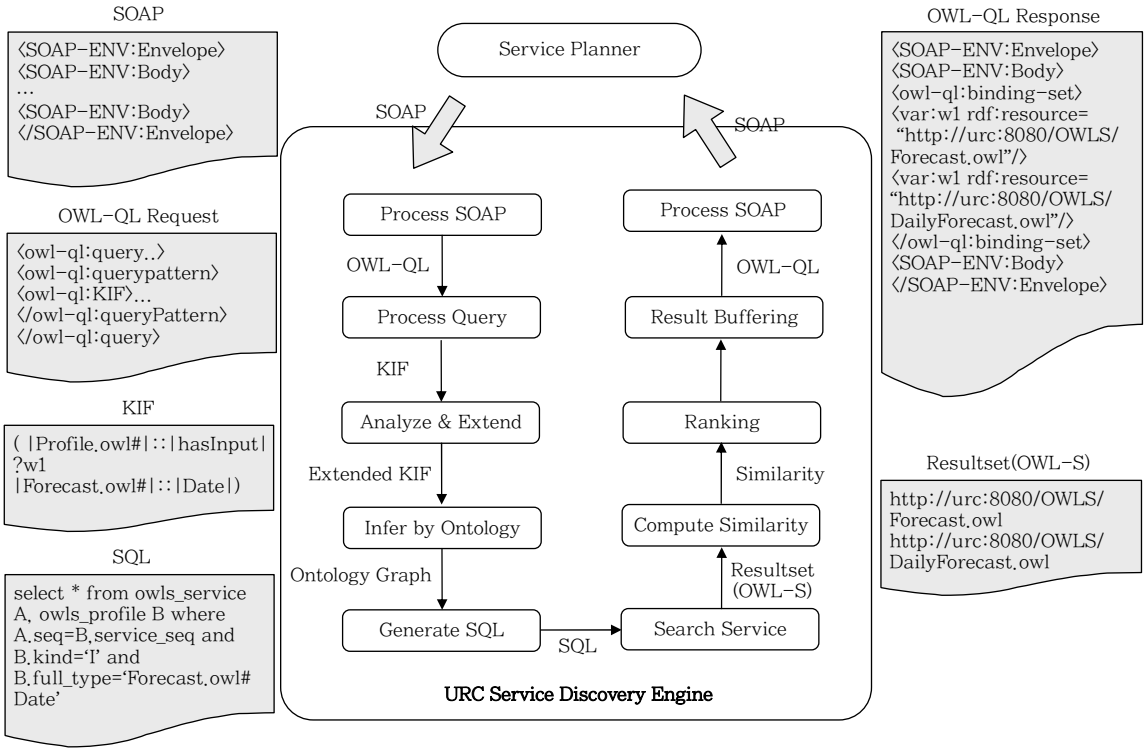
3. URC 서비스 실행

URC가 요구하는 복합적인 서비스를 수행하기 위해서는 다양한 단위 서비스를 조합하여 수행하기 위한 프로세스의 도입이 필요하며 이를 실행하기 위한 시스템 역시 필요하다.

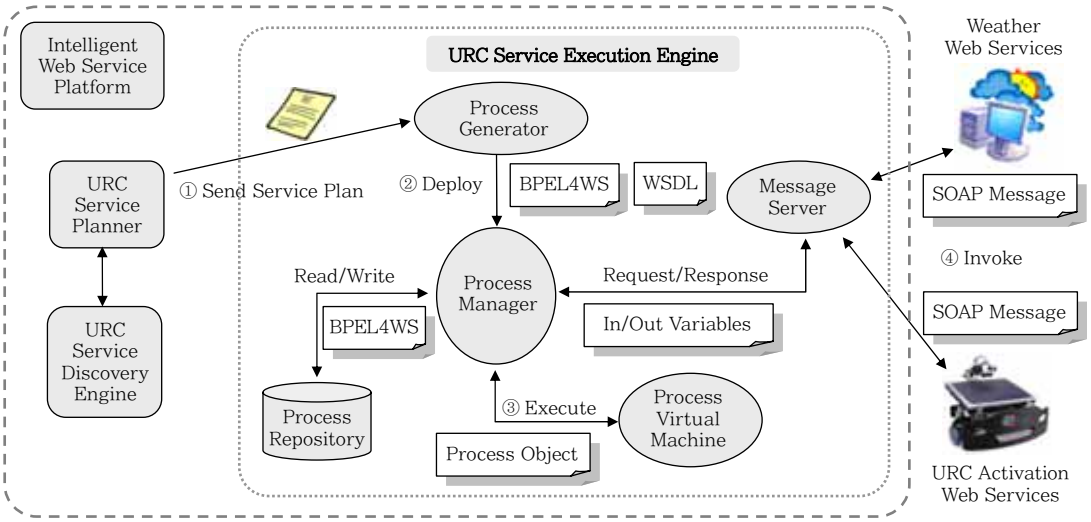
• Query:(And (|http://urcsp .etri.re.kr/composition/owl-s/1.0/Profile.owl#|::|haslnput|?w1 |http://urcsp.etri.re.kr/concepts/2004/10/Location.owl#|::|City|) (http://urcsp.etri.re.kr/composition.owl-s/1.0/Profile.owl#|::|hasOutput| ?w1 |http://urcsp.etri.re.kr/concepts/2004/10/Weather.owl#|::|WeatherStatus|))

Ranking	Service URL
1	http://urcsp.etri.re.kr/services/concepts/2004/10/getCurrentWeatherStatus.owl Input Type: http://urcsp.etri.re.kr/concepts/2004/10/Location.owl#City Output Type: http://urcsp.etri.re.kr/concepts/2004/10/Weather.owl#WeatherStatus
2	http://urcsp.etri.re.kr/services/concepts/2004/10/ShorttermWeagherStatus.owl Input Type: http://urcsp.etri.re.kr/concepts/2004/10/Location.ow#City Input Type: http://urcsp.etri.re.kr/concepts/2004/10/Time.ow#Date Output Type: http://urcsp.etri.re.kr/concepts/2004/10/Weather.owl#WeatherStatus
3	http://urcsp.etri.re.kr/services/concepts/2004/10/getCurrentWindDirection.owl Input Type: http://urcsp.etri.re.kr/concepts/2004/10/Location.owl#City Output Type: http://urcsp.etri.re.kr/concepts/2004/10/Measurement.owl#WindDirection
4	http://urcsp.etri.re.kr/services/concepts/2004/10/getCurrentPrecipitationProbability.owl Input Type: http://urcsp.etri.re.kr/concepts/2004/10/Location.owl#City Output Type: http://urcsp.etri.re.kr/concepts/2004/10/Measurement.owl#PrecipitationProbability
5	http://urcsp.etri.re.kr/services/concepts/2004/10/getCurrentWeather.owl Input Type: http://urcsp.etri.re.kr/concepts/2004/10/Location.owl#City Output Type: http://urcsp.etri.re.kr/concepts/2004/10/Weather.owl#OverlandForecast

(그림 6) 질의 결과



(그림 7) URC 탐색 엔진의 서비스 탐색 흐름도



(그림 8) URC 서비스 실행 시스템

URC 서비스 실행 시스템은 사용자의 요구를 처리하기 위해 URC가 제공하는 서비스와 인터넷 등의 URC 외부의 환경에서 제공되는 일련의 서비스를 웹서비스로 통일하고 이들을 연계하여 구성된 프로세스를 실행 시간에 바인딩하여 수행할 수 있는 환경을 제공한다.

(그림 8)은 URC 서비스 실행 시스템의 구조를 표현한 것으로 프로세스 생성기, 프로세스 배치기,

메시지 서버 그리고 프로세스 관리 및 실행기로 구성된다.

가. 프로세스 생성기

프로세스 생성기는 URC 서비스 플래너로부터 OWL-S 프로세스 모델에 정의된 IOPE 정보와 서비스 실행순서 정보를 입력 받아서 실행 가능한 BPEL4WS 인스턴스를 생성한다. BPEL4WS 인스턴스를 생성하기 위해서 먼저 프로세스 저장소를 검색한다. 만약 원하는 프로세스가 저장소에 없다면 새로운 프로세스를 생성하고 입력 받은 IOPE 정보와 서비스 계획을 프로세스 저장소에 저장한다. 프로세스 저장소에는 IOPE를 간선으로 하고 WSDL 오퍼레이션을 노드로 하는 서비스 그래프가 저장된다. 노드로 사용하는 서비스에 대해서 서비스 분류에 대한 온톨리지를 사용한다.

실제 BPEL4WS를 생성하기 위해서는 OWL-S 그라운딩 정보가 필요하다. OWL-S의 그라운딩으로 WSDL에 대한 URI를 얻어 와서 그 WSDL에서 사용되는 메시지 타입 정보와 오퍼레이션 정보를 이용하여 BPEL4WS 인스턴스를 구성한다. 이때 URC에서 입력되는 입력 데이터가 WSDL의 기본 오퍼레이션과 일치하지 않을 수도 있다. 이러한 경우는 variable을 별도로 정의하고 BPEL4WS의 assign 문을 이용하여 variable을 조작한다.

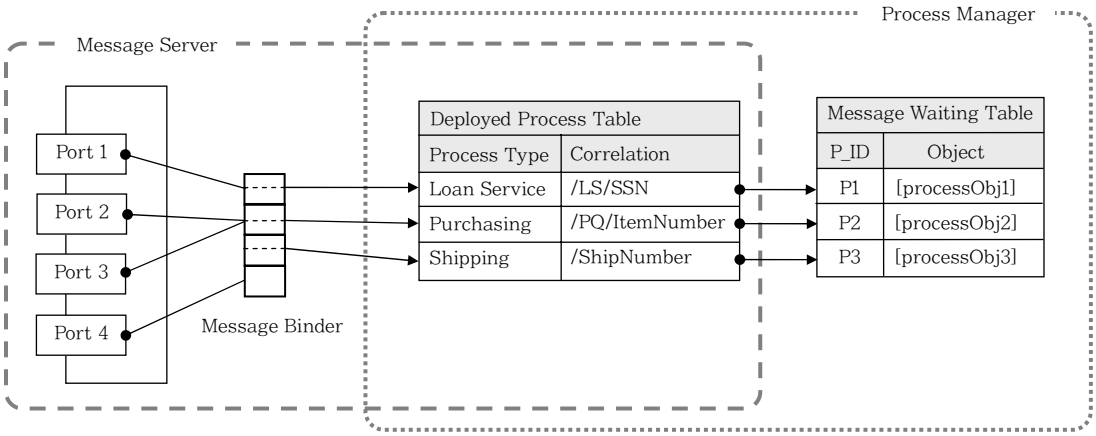
BPEL4WS를 정의할 때는 동기적 프로세스인지 비동기적 프로세스인지를 구별할 필요가 있으며 동기적 프로세스의 경우 receive/reply 문의 쌍으로 시작과 끝이 구성되고, 비동기적 프로세스의 경우 receive/invoke 문의 쌍으로 시작과 끝이 구성되며 여러 개의 receive 문이 존재할 수 있다.

나. 프로세스 배치기

프로세스 배치기는 WSDL과 생성된 BPEL4WS 인스턴스를 프로세스 생성기로부터 입력 받아서 BPEL4WS가 프로세스 관리기에서 실행 가능하도록 배치하고 BPEL4WS에서 호출하는 외부 웹서비스에 대한 클라이언트 파일을 생성한다. 또한 BPEL4WS가 비동기적인가 동기적인가에 따라 구별하여야 한다. 왜냐하면 동기적인지 비동적인지에 따라서 프로세스를 관리기에서 관리하는 메커니즘이 다르기 때문이다. BPEL4WS를 실행 가능한 런타임 모델로 변환한 것은 프로세스 가상 머신에 의해서 실행된다. 이 런타임 모델은 프로세스의 흐름 구조와 입출력되는 데이터에 대한 정보를 포함하고 있다. 프로세스 실행기는 BPEL4WS를 실행할 수 있는 프로세스 실행 가상 머신을 제공한다. 프로세스 배치기에서 생성된 런타임 모델은 프로세스 실행기에서 실행된다. 프로세스 실행기는 프로세스 관리기에 포함되어 있다.

다. 프로세스 관리기

프로세스 관리기는 프로세스 생성기에서 생성된 프로세스가 프로세스 배치기에서 배치기에 의해서 배치가 끝나면 배치된 프로세스를 직접 실행하고 관리하는 기능을 수행한다. 프로세스 관리기는 프로세스의 실행과 프로세스 인스턴스의 생명주기를 관리한다. 프로세스 관리기는 또한 다중 프로세스를 지원하고 롱러닝 프로세스(long running process)를 관리할 수 있다. 롱러닝 프로세스는 실행 프로세스가 파트너와 비동기적 대화를 포함하고 있는 경우에 발생한다. 비동기적 프로세스 여부에 대한 정보는 프로세스 배치기로부터 전달 받는다. URC 서버는 다수의 URC에 대한 요청을 처리할 수 있다. 프로세스 관리기는 프로세스 인스턴스와 사용되는 데이터를 영속적으로 관리하는 기능을 가진다. 프로세스 관리기는 실행이 끝난 프로세스 인스턴스의 결과를 URC에게 전달하기 이전에 컨텍스트를 변환시켜 전달한다. 컨텍스트를 변환하기 위한 변환 방법은 WSDL에 대한 OWL-S를 생성할 때 XSL 파일 형식으로 생성하여 OWL-S의 grounding에 기술되어 있다.



(그림 9) 입력 메시지 파이프라인

라. 메시지 서버

메시지 서버는 호출된 모든 웹서비스의 결과 메시지와 배치된 서비스에 대한 URC 요청 입력 메시지를 관리하기 위해 어떤 파트너로부터 메시지가 입력되는지를 알아야 한다. 왜냐하면 메시지 입력을 기다리는 실행중인 프로세스 인스턴스에 정확히 메시지를 전달하기 위해서이다. 보다 구체적으로 메시지 서버는 메시지 바인더를 유지한다. 메시지 바인더는 입력 포터로부터 메시지 코릴레이션을 통한 프로세스 인스턴스로 입력된 메시지의 통로의 연결을 유지한다. 메시지 바인더와 배치 프로세스 테이블을 이용하여 입력된 메시지는 대기중인 실행 프로세스 인스턴스를 실행하도록 트리거한다. (그림 9)는 메시지 서버와 프로세스 관리기 사이의 메시지 파이프라인을 표현한 것이다. 메시지 서버로부터 입력 받은 메시지는 배치 프로세스 테이블을 통해서 실행중인 프로세스들 중 하나로 전달된다. 이것은 BPEL4WS의 코릴레이션을 이용하여 만들어진다.

마. 프로세스 실행기

프로세스 실행기는 프로세스 관리기에서 받은 프로세스 객체를 실행한다. 프로세스 객체는 BPEL4WS 스펙에 기술된 구조화된 행위자와 기본적인 행위자로 구성되어 있다. 기본적인 행위자는 웹서비스를 호출하는 행위자, 일정 시간을 대기하도록 하는 행위자, 모든 행위자의 실행을 종료시키는 행위자, 실패를 나타내는 행위자로 구성되어 있다. 구조화된 행위자는 하나 이상의 행위자들의 순서를 지정해주는 행위자, 조건 분기를 위한 행위자, 반복을 위한 행위자, 이벤트의 발생에 의해 시작되는 행위자, 동시 수행을 위한 행위자들로 구성되어 있다. 프로세스 실행기는 구조화된 행위자의 흐름 속에서 기본적인 행위자들의 개별적인 실행을 수행한다. 추가로 프로세스 실행기는 실행중 발생한 실패를 대비한 예외처리를 지원한다.

Ⅳ. 결론

IT 기술의 발달과 사회의 고령화, 인간 생활의 질적 향상 및 복지, 풍요와 여유에 대한 추구 등 여러 요인으로 인해 향후 서비스 로봇에 대한 수요가 증가할 것으로 예견된다. 현재의 로봇 시장은 산업용 로봇이 근간을 이루고 있으나 점차 홈서비스나 재활, 엔터테인먼트, 교육 등 비 산업용 로봇의 시장이 활성화되고 있는 추세에 있다. 특정 작업에 특화된 산업용 로봇과 달리 비산업용 로봇은 사용자로부터 다양한 서비스 요구를 받아들이게 되며 이러한 요구를 만족시키기 위해서는 로봇 기술과 IT 기술을 접목하는 지능화 서비스 수준의 향상이 필요하다.

본 연구는 로봇의 서비스 범위를 웹 기반으로 확장시켜 로봇에 내장되어 있지 않은 서비스는 웹에 산재한 지식과 기존 서비스를 재구성하여 사용자의 필요를 충족시키는 서비스를 제공할 수 있으며 이를 통해 다양하고 유연한 로봇용 킬러 애플리케이션이나 비즈니스 모델의 개발이 가능하다.

본 기술의 응용분야는 로봇 분야에 한정하지 않으며 생활과 밀접한 백색 가전 및 자동차 등에 응용하여 인터넷을 통한 콘텐츠 송수신 및 원격 제어뿐만 아니라 웹으로부터 획득한 지식과 정보를 바탕으로 한 웹 기반의 서비스를 제공하여 해당 산업의 고부가가치 창출이 기대될 수 있다.

약어 정리

IOPE	Input, Output, Precondition, Effect
OWL	Web Ontology Language
QL	Query Language
RDF	Resource Description Framework
RDFS	RDF Schema
SOAP	Simple Object Access Protocol
UDDI	Universal Description, Discovery and Integration
URC	Ubiquitous Robotic Companion
WSDL	Web Service Description Language
XTM	XML Topic Maps

참 고 문 헌

[1] Martin Gudgin et al., "Simple Object Access Protocol(SOAP) 1.2," W3C Recommendation, 24 June 2003.

[2] Dabid Booth et al., "Web Service Description Language(WSDL) 2.0," 21 Dec. 2004.

[3] Uche Oqbuji, "Using WSDL in SOAP Applications: An introduction to WSDL for SOAP Programmers," IBM DeveloperWorks, 1 Nov. 2000.

[4] F. Buijze, P.F.A.d. Dijcker, and A.A. Hornickel, "Universal Description, Discovery, and Integration(UDDI) 3.0.1," 14 Oct. 2003.

[5] Tim Berners-Lee, "Weaving the Web," Harpur, San Francisco, 1999.

[6] Tim Berners-Lee, James Hendler, and Ora Lassila, "The Semantic Web," Scientific American, May 2001.

[7] Dieter Fensel and Mark A. Musen, "The Semantic Web: A Brain for Humankind," *IEEE Intelligent Systems*, 2001.

[8] David Martin et al., "OWL-based Web Service Ontology Ver. 1.1," 2004, http://www.daml.org/services/owl-s/

[9] David Martin et al., "OWL-S: Semantic Markup for Web Services," 22 Nov. 2004.

[10] Christoph Bussler, Dieter Fensel, and Alexander Maedche, "A Conceptual Architecture for Semantic Web Enabled Web Services," ACM Special Interest Group on Management of Data, Vol.31, No.4, Dec. 2002.

[11] Tony Andrews et al., "Business Process Execution Language for Web Services Version 1.1," 2003, http://www.128.ibm.com/developerworks/library/ws-bpel/

[12] Assaf Arkin et al., "Web Service Choreography Interface(WSCI) 1.0," 2002, http://www.w3.org/TR/wsci/

[13] Richard Fikes, Pat Hayes, and Ian Horrocks, "OWL-QL: A Language for Deductive Query Answering on the Semantic Web," KSL Technical Report 03-14, 2003.