e-비즈니스 레지스트리 통합 질의 시스템 설계*

김계용() 황윤영 유정연 이규철
충남대학교 컴퓨터공학과
{kykim1(), yyhwang, jyyou, kclee}@ce.cnu.ac.kr

Design of Integrated Query System for e-Business Registries

Kye-Yong Kim(), Youn-Young Hwang, Jeong-Youn Yu, Kyu-Chul Lee
Dept. of Computer Engineering, Chungnam National University

요 약

레지스트리는 웹을 통한 정보의 생산, 분배, 발견을 가능하게 하는 기반 구조를 말한다. 이러한 레지스트리를 통해 동적으로 B2B 거래를 편리하게 할 수 있으며, 자원을 공유할 수 있게 된다. 따라서 레지스트리는 e-비즈니스 서비스에 핵심 기능이라고 할 수 있다. 그러나 현재 이러한 레지스트리들에 대한 개발이 독립적으로 개발되고 있으며, 따라서 이들 레지스트리 간의 상호 연동이라든지, 사용자 입장에서의 레지스트리 이용은 제한점을 가질 수밖에 없다. 현재 SUN에서 JAXR이라는 패키지를 내놓고 있으며, 이 패키지를 이용하여 어떤 특정 레지스트리에 제한되지 않는 포괄적인 API를 사용하여 레지스트리를 검색하거나 저장할 수 있도록 하고 있다. 본 논문에서는 JAXR을 이용하여 여러 레지스트리를 효율적으로 검색할 수 있고, e-비즈니스 구축에 있어서 보다 넓은 유연성을 제공하며 높은 레벨의 서비스 구조를 제공하기 위한 시스템 구조를 제안했다.

1. 서 론

e-비즈니스 서비스에 관한 프레임워크 및 관련 응용의 개발이 활발해지면서 기업간의 e-비즈니스 서비스를 이용한 실제적인 전자 거래가 증가하고 있다. 따라서 기업 간 e-비즈니스에서 사용자의 요구 사항에 맞는 서비스를 찾아주고 서비스 이용에 대한 자동화를 제공해줌으로써 기업의 생산성과 효율을 높여줄 수 있는 기반 기술의 개발이 필요하다.

e-비즈니스 레지스트리로서 ebXML[1]과 UDDI[2]가 제정되면서, 이들 레지스트리 간의 상호운용에 대한 관심이 증가하고 있다. 서로 다른 정보 모델을 가진 레지스트리 간의 통합을 위해서는 그들의 정보 모델에 대한 매핑이 필요하며, 필요에 따라 보다 넓고 수용적인 새로운 정보 모델을 정의할 필요가 있다.

이러한 취지에서 SUN에서 JAXR[3]이라는 스펙을 발표했으며, 이를 이용한 레지스트리 통합에 대한 연구들이 진행 중이다. 그 예로서 IBM의 BE4WS에서의 USML[4]은 JAXR을 이용하여 UDDI 레지스트리 간의 질의 통합을 만들고자 했다.

본 논문에서는 JAXR을 이용하여 현재 가용한 UDDI와 ebXML의 공용 레지스트리에 보다 효율적이고 발전된 질의를 할 수 있는 방법을 제공하고자 했다. 즉, 기본적으로 분산된 UDDI 레지스트리와 ebXML 레지스트리를 검색할 수 있으며, 동일한 질의를 여러 레지스트리에 실행하여 그 결과물 조합하거나, 다양한 질의를 각각의 레지스트리에 실행하여 그 결과를 조합하는 것이다. 또한 질의 표현의 정확성을 위해 검색 객체와 관련된 메타 정보들을 추가하여 질의에 표현함으로써 보다 사용자가 정확한 질의를 할 수 있도록 하였다.

2. 관련 연구

2.1 ebXML[1]

ebXML은 국제 EDI 표준을 추진해왔던 UN/CEFACT와 OASIS가 주축이 되어 1999년 11월부터 시작된 XML[2]을 이용한 인터넷 기반의 e-비즈니스 표준이다. ebXML은 "Creating A Single Global Electronic Market"이라는 기치 아래 국제적인 하나의 전자시장을 형성하려는 취지로 설립되었으며 여러 기업에서 실제적인 적용을 위해 노력하고 있다.

현재 ebXML Registry/Repository는 버전2까지 스펙이 발표되어 있다. 홍콩대학에서 SUN과 협력하여 베타 버전을 개발하고 있으며, 공용 ebXML 레지스트리로서 테스트 중에 있다. 그 밖에 Global Technologies, Inc.사와 같은 기업에서도 개발 중에 있으며, 한국에서도 전자거래진흥원이 ebXML 중앙등록저장소를 운영 중에 있다.

그러나, ebXML에 대한 연구 개발이 활발히 진행 되면서 다양한 ebXML 레지스트리들간의 호환성에 대한 문제가 고려되고 있다. 그러한 이유로 차기 버전에서는 ebXML 레지스트리들간의 통합을 위해 필요한 내용들을 기술하고 있는 스펙[3]이 만들어지고 있다.

2.2 UDDI[4]

Microsoft, IBM 등 300개가 넘는 회사들이 참여해 공동 개방형 표준과 비독점적 기술을 기반으로 한 레지스트리를 위한 스펙을 개발하였다. UDDI는 비즈니스와 해당 서비스에 대한 정보를 구조화된 방법으로 수용하기 위해 디자인된 공용 레지스트리로서 UDDI를 통해 회사와 회사가 제공하는 웹 서비스에 대한 정보를 게시하고 검색할 수 있다. 이와 같은 방법으로 UDDI는 웹 서비스를 기반으로 하는 소프트웨어 환경의 인프라 역할을 하게 된다. UDDI는 현재 IBM, Microsoft, SAP에서 public 레지스트리를 운영 중에 있다.

그러나 UDDI도 ebXML과 마찬가지로 다양한 UDDI 레지스트리 간의 데이터 호환이 문제가 된다. 현재 UDDI도 레지스트리 간의 통합을 위한 스펙[5]을 정의하고 있다.

2.3 USML[6]

IBM에서 JAXR의 BusinessQuery를 이용하여 동시에 여러 UDDI 레지스트리에 질의를 할 수 있도록 USML라는 패키지를 개발하였다. USML은 UDDI 레지스트리들에 질의를 하고, 그

*본 연구는 소프트웨어연구센터와 BK21 충남대학교 정보통신인력양성사업단의 지원을 받았음.

결과를 조합하는 방식을 취한다.

그러나 USML UDDI 레지스트리에만 질의할 수 있도록 한정되어 있고, Business, Service, ServiceType의 3가지 UDDI 데이터 모델에 상당히 한정된 객체들만을 질의할 수 있다. 또한 AND와 OR 두 가지의 연산자만을 지원하여 URL상의 관계를 나타낸다. 이러한 USML을 이용해서는 비교적 단순하고 평면적인 질의 밖에 할 수 없다. 따라서 본 논문에서는 UDDI 뿐만 아니라 ebXML 레지스트리를 검색할 수 있고, 보다 많은 객체들의 검색을 지원하며, url 사이의 조인이 가능한 질의 구조를 제안하였다.

3. 설계

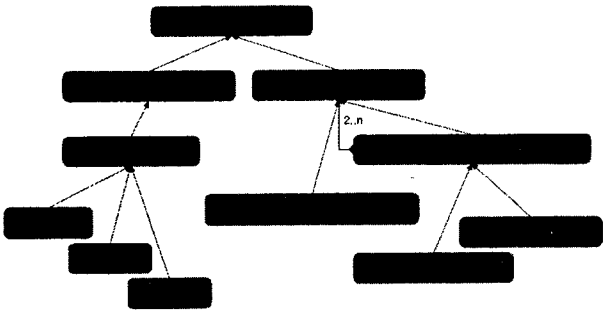
3.1 질의 스키마

본 시스템은 XML 메시지를 기반으로 작용한다. 따라서 질의문과 응답문 모두 XML 인스턴스이다. XML 인스턴스를 메시지로 사용하기 위해서는 XML 스키마가 필요하다. 이러한 스키마는 XML 인스턴스의 유효성을 검사할 뿐 아니라, 사용자가 정해진 형식의 질의를 생성하도록 요구한다.

질의 스키마는 파일로 저장이 되어 있으며, [그림 3]에서의 통합 질의 생성기에서 사용되고, 응답 스키마는 결과 처리기에 의해 사용된다. 또한, 통합 질의 처리기에서는 질의 스키마를 이용해 사용자가 질의문 생성시 필요한 정보를 제대로 입력했는지 검증한다.

스카마 구조는 JAXR의 정보 모델을 기반으로 만들어졌다. JAXR에서 정의하고 있는 DeclarativeQuery와 BusinessQuery를 이용했다. 검색할 수 있는 객체 타입으로는 BusinessQuery의 경우 Organization, ServiceBinding, Service, Concept, RegistryPackage, ClassificationScheme, Association의 7가지를 지원하며, DeclarativeQuery의 경우 JAXR의 정보 모델이 정의하고 있는 모든 객체의 검색이 가능하다.

[그림 1]는 본 시스템에서 제안한 질의 스키마의 일부로서 최상의 노드로부터 질의를 형성하는 핵심 앨리먼트들 간의 관계를 보여주고 있다.



[그림 1] 질의 스키마 구조

질의 스키마의 최상위 앨리먼트는 FederatedQuery이며 DeclarativeQueryClause와 BusinessQueryClause 앨리먼트를 갖는다. 다시 DeclarativeQueryClause는 DeclarativeQuery를 자식으로 갖는다. DeclarativeQuery는 FilterQuery, XQuery, SQLQuery를 하위 앨리먼트로 갖는다.

BusinessQueryClause는 BusinessQuerySimpleClause와 BusinessQueryCompoundClause를 하위 앨리먼트로 갖는다. BusinessQueryCompoundClause는 순환적인 질의 형태를 지원하기 위해 다시 여러 개의 BusinessQueryClause를 하위 앨리먼트로 갖는다. BusinesQuerySimpleClause는 위에서의 정의한 검색이 지원되는 7가지 객체를 하위 앨리먼트로 갖는다.

DeclarativeQuery와 BusinessQuerySimpleClause는 속성으로 url을 갖는데, 이 url 값에 각각의 레지스트리에 대한 URL 값을 지정하게 된다. BusinessQueryCompoundClause 아래에는 BusinessQuerySimpleClause 사이의 관계를 나타내기 위한 두 가지 앨리먼트들이 있다.

- ConnectivePredicate: url 사이의 AND/OR
- RationalPredicate: 지정된 url의 객체들 간의 EQ/NE
  - 객체들의 UUID를 이용하여 비교
  - url 사이의 조인연산을 위해 사용

그리고, BusinessQueryCompoundClause는 returnType을 속성으로 가지는데 그 값으로 BusinessQueryCompoundClause에서 최종적으로 반환되는 객체의 타입을 지정하게 된다.

또한, 어떤 객체를 검색할 때, 검색 조건으로 이름과 같은 기본적인 정보 이외에 그 객체의 다양한 메타 정보들을 지정할 수 있다. 예를 들어, 지리적으로 North America에 위치해 있고, 소프트웨어를 개발하는 회사들을 묶은 Software라는 Package와 Association 관계를 맺고 있는 SUN이라는 이름을 가진 회사를 검색하는 것이 가능하다. 여기서 지리정보를 나타내기 위해 사용되는 Classification 객체와 Software Package와의 관계를 나타내기 위해 사용되는 Association 객체의 관계는 AND이다. 이렇게 하나의 객체에 대한 메타 정보들은 같은 레벨에 있을 경우 AND의 관계를 가진다.

[그림 2]은 통합 질의 생성기에 의해 생성된 질의문으로 Microsoft에서 Software라는 서비스를 제공하거나, IBM에서 Hardware라는 서비스를 제공하는 Organization을 검색하는 질의문의 예이다.

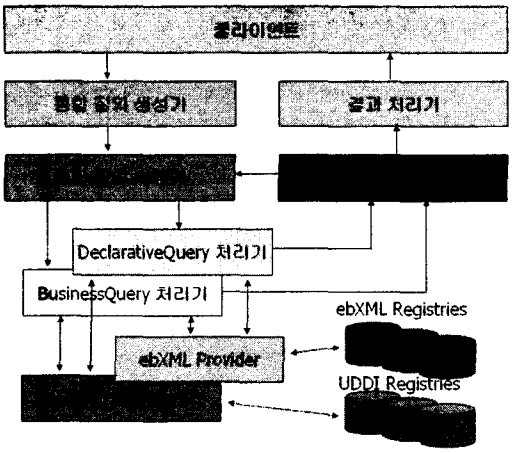
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FederatedQuery SYSTEM "FederatedQuery.dtd">
<FederatedQuery>
 <BusinessQueryClause>
  <BusinessQueryCompoundClause
    returnType="Organization">
   <ConnectivePredicate cp="OR">
   <BusinessQueryClause>
    <BusinessQuerySimpleClause
      url="http://uddi.microsoft.com/inquire">
     <Service>
      <NamePatterns>
       <NamePattern np="Software"/>
      </NamePatterns>
     </Service>
    </BusinessQuerySimpleClause>
   </BusinessQueryClause>
   <BusinessQueryClause>
    <BusinessQuerySimpleClause
      url="http://www-3.ibm.com/services/uddi/v2beta
           /inquiryapi">
     <Service>
      <NamePatterns>
       <NamePattern np="Hardware"/>
      </NamePatterns>
      <FindQualifiers>
       <FindQualifier type="CASE_SENSITIVE_MATCH"/>
      </FindQualifiers>
     </Service>
    </BusinessQuerySimpleClause>
   </BusinessQueryClause>
  </BusinessQueryCompoundClause>
 </BusinessQueryClause>
</FederatedQuery>
```

[그림 2] 질의 인스턴스 예

3.2 시스템 구조

본 논문에서 제안된 시스템은 분산된 이질 레지스트리에 대해 통합된 질의를 수행할 수 있는 기능을 제공한다. 클라이언트에서 생성된 질의를 UDDI나 ebXML등의 다양한 레지스트리를 통하여 검색을 하고 각각의 레지스트리로부터 반환된 값을 통합의 과정을 거쳐 보여주게 된다. [그림 3]는 본 논문에서

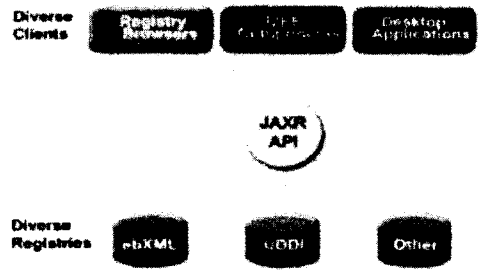
제안한 시스템의 개략적인 구성이다.



[그림 3] 시스템 구조

3.2.1 JAXR[7]

JAXR은 다양한 XML 레지스트리들을 접근할 수 있도록 하기 위한 목적으로 SUN에서 발표한 스펙이다. JAXR은 B2B 통합을 가능하게 하기 위해 분산된 레지스트리 서비스에 대한 API를 제공한다. 따라서 각각의 레지스트리 간의 데이터 매핑이 필요한데, JAXR은 ebXML 레지스트리와 UDDI 레지스트리의 JAXR 정보 모델과의 바인딩을 포함하고 있다. [그림 4]은 JAXR을 통한 레지스트리의 상호운영성을 보여주는 그림이다.



[그림 4] JAXR의 상호운영성

JAXR이 제공하는 서비스는 크게 Life Cycle Management와 Query Management로 나눠진다. Life Cycle Management는 저장, 삭제 등의 기능을, Query Management는 SQL이나 ebXML의 Filter Query 형식을 지원하는 DeclarativeQuery와 단순한 비즈니스 레벨의 API를 제공하는 BusinessQuery를 지원한다. 현재 스펙 버전 1을 발표한 상태이고, 구현된 패키지를 함께 배포하고 있다.

본 시스템은 JAXR의 Query Management의 API들을 이용하여 질의를 실행한다. 즉 [그림 3]에서 DeclarativeQuery 처리기와 BusinessQuery 처리기, 그리고 ebXML Provider와 UDDI Provider 부분은 JAXR을 이용한다.

3.2.2 모듈 설명

■ 클라이언트

클라이언트는 사용자로부터 검색에 필요한 정보를 입력받는 역할을 한다.

■ 통합 질의 생성기

클라이언트로부터 받은 정보를 바탕으로 XML 질의문을 생성하는 일을 한다. 여기서 질의문은 질의 스키마에 유효한 질의문을 생성한다. 또한, 질의문을 생성하기 위해 스크립트 언어로 JSP를 이용한다.

■ 통합 질의 처리기

통합 질의 생성기로부터 XML 질의문을 받으면 질의문을 파싱하고 질의문의 구조을 분석한다. 파싱된 내용을 바탕으로 DeclarativeQuery나 BusinessQuery를 생성한다. 통합 질의 처리기는 순환적인 질의 형태를 지원하기 때문에 이 질의를 실행하고 그 결과를 다시 이용할 수 있다. 그 밖에 URL 사이의 조인이나 연산 등의 일을 처리하는 등 통합 질의 처리를 위한 전반적인 일을 담당하다.

■ DeclarativeQuery 처리기

DeclarativeQuery를 실행하고 그 결과를 결과 추출기에 반환한다.

■ BusinessQuery 처리기

BusinessQuery를 실행하고 그 결과를 결과 추출기에 반환한다.

■ ebXML Provider

DeclarativeQuery나 BusinessQuery를 ebXML 질의로 매핑하고 질의를 실행하여 그 결과를 반환한다.

■ UDDI Provider

DeclarativeQuery나 BusinessQuery를 UDDI 질의로 매핑하고 질의를 실행하여 그 결과를 반환한다.

■ 결과 추출기

반환된 결과로부터 필요한 정보를 얻어오고, 그 값을 통합 질의 처리기에 반환하거나 결과 처리기로 넘겨준다.

■ 결과 처리기

결과 추출기로부터 반환된 정보를 이용하여 응답 스키마에 유효한 응답문을 생성하다. 이 응답문은 XML 형태로 되어 있으며 클라이언트를 통해 사용자에게 보여진다.

4. 결론 및 향후 연구 과제

인터넷 기반기술의 발전과 함께 개방형 e-비즈니스 레지스트리에 대한 연구가 활발히 진행되고 있다. ebXML이나 UDDI가 그 대표적인 예라고 할 수 있다.

그러나, ebXML은 분산된 ebXML 레지스트리들 간의 상호연동이, UDDI도 여러 UDDI 레지스트리들 간의 데이터 연동이 해결해야 할 문제로 남아있다. 본 논문은 이러한 공용 레지스트리를 기반으로 이질적인 ebXML과 UDDI 레지스트리 간의 통합 질의를 지원하여 사용자에게 질의의 편의성과 보다 상세한 질의를 할 수 있는 방법을 제공하는 시스템을 설계했다.

향후 과제로는 응답 스키마에 대한 정의와 시스템 구현이 남아있다.

5. 참고 문헌

[1] ebXML.org, ebXML, http://www.ebxml.org
[2] W3C, Extensible Markup Language(XML) 1.0, http://www.w3c.org
[3] ebXML.org, Cooperating Registries, http://ebXML.org
[4] UDDI.org, UDDI, http://uddi.org,
[5] UDDI.org, UDDI Replication Specification, http://uddi.org
[6] IBM, UDDI Search Markup Language(USML), http://www.alphaworks.ibm.com/tech/be4ws/
[7] SUN, Java API for XML Registries, http://jcp.org/jsr/detail/93.jsp