CORBA를 기반으로 한 XML 통합 정보 검색 시스템 개발에 관한 연구

정병인* · 이재완 · 이정식

*군산대학교 정보통신공학과

A Study on the Development of Integrated XML Information Retrieval System Based on CORBA

Byoung-in Jung* · Jae-wan Lee · Jung-sik Lee

*Dept . of Telecommunication Engineering, Kunsan National University

E-mail : indan@dslab.kunsan.ac.kr

요 약

컴퓨터와 통신기술의 발달로 분산되고 이질적인 시스템들간에 상호연동이 가능해졌다. 그렇지만 전통적인 정보 시스템은 서버의 수가 많아질수록 애플리케이션이 복잡해지는 문제점이 있다. 이러한 점을 극복하기 위해 XML을 공통 데이터 형식으로 사용하여 데이터에 동적으로 접근 할 수 있도록 하고, 또한 CORBA를 기반으로 하여 각각의 데이터베이스에 접근의 투명성을 보장하였으며, 인터페이스의 복잡도를 감소시킬수 있는 3-Tier 클라이언트/서버 시스템을 개발한다.

1. 서 론

인터넷은 이용이 편리하며 단일 인터페이스를 통해 분산된 정보자원의 활용을 가능하게 함으로써 1990년 이래로 가장 효과적인 컴퓨팅 환경을 제공하고 있다. 그러나 현재의 웹은 HTML의 비구조적성으로 인해 데이터 관리 및 확장성의 한계를 드러내고 있다. 이 문제점을 극복하기 위해 개발된 차세대 언어인 XML을 사용하여 데이터의 관리 및 처리를 효율적으로 할 수 있게 되었다.

이러한 장점을 살린 XML은 현재 대규모의 전자상거래, EDI, 의료관리 시스템 등에 활용되고 있다. 그러나 대부분의 어플리케이션은 자바를 기반으로 한 2계층 아키텍처로 만들어져 있어 복잡한 업무를 처리하고자 할 경우 애플릿의 다운로드 속도 및 업무 처리 속도가 느려지는 결과를 초래하고 있다.

이와 같이 제한적인 한계를 극복하기 위한 근본적인 해결책으로 OMG에서 제시한 CORBA가 활용되고 있다.

CORBA는 분산 이기종 플랫폼 환경 하에서 분산 객체 기술에 기반하여 클라이언트/서버 아키텍처를 규정하고 있으며, 객체간의 서비스 요청 및 전달을 ORB에 전담시킴으로써 자바 언어가 지향하고 있지 않은 포괄적인 원거리 호출을 지원한다. 뿐만 아니라, IDL을 이용하여 인터페이스와 구현의 상세를 분리시킴으로써 클라이언트와 서버의 독립적인 개발을 보장하며 IDL로 정의된 표준 인터페이스를 통한 시스템 통합을 가능하게 해준다.

본 연구에서는 XML과 CORBA의 장점을 살려 문서 데이터에 동적으로 접근 할 수 있는 CORBA 기반 분산 객체 기술을 사용한 3 Tier 클라이언트/서버 시스템을 개발한다.

2. 관련연구

2.1 XML

전통적인 데이터 공유방법은 데이터 생산 및 소비자의 데이터 교환을 위해 어플리케이션을 작성하는 것이다. 이 방법은 통합 서버가 많아질수록 기하 급수적으로 코드가 늘어나는 문제점을 가지며 또 코드를 재사용 하기가 힘들다.

이런 문제의 대안으로 XML을 공용 데이터 교환양식으로 사용해 서버/클라이언트 모두가 하나의 공통된 논리적 포맷을 가진 데이터에 접근 할 수 있게하는 것이다.

XML문서를 사용할 경우 각각의 통합된 컴포넌트들은 오직 하나의 XML Parser만을 필요로 하

며, 컴포넌트를 추가하는 경우에도 코드가 기하급수적으로 증가하지 않는다.[1] 이는 XML데이터가 스스로 해석할 수 있는 태그를 가지고 있기 때문이며, XML데이터는 어댑터를 통해 전달될 때마다 동적으로 평가될 수 있다. 분산 데이터 모델에 XML을 적용하면 데이터의 통합에서 야기되는 혼란을 피해 효율적인 분산 데이터베이스를 구성할 수 있다.

2.2 CORBA

CORBA는 분산 객체 컴퓨팅 미들웨어 표준으로서 유동적이고 재사용이 가능한 서비스들과 응용프로그램의 개발을 지원하기 위해 설계되었다. 기본구조의 중요한 구성요소는 ORB(Object Request Broker)이며, 주요 기능은 투명성(Transparent) 있게 요구를 객체들에 전달하는데 있다.[4]

클라이언트와 서버는 ORB를 사용함으로서 상호간에 대한 정보를 저장할 필요가 없으며, ORB는 클라이언트가 서버에게 보내는 요청을 위해 중간 코드처럼 동작한다. 즉, ORB는 클라이언트 어플리케이션으로부터 요청을 받아들여서 서버에 있는 구현(Implementation)들 중에서 요청을 수행할 적당한 구현을 선택하여 요청을 보낸다.[5]

3. 시스템 구성

3.1 시스템 구성

본 연구에서는 XML 문서 데이터에 동적으로 접근 할 수 있는 CORBA 기반 분산 객체 기술을 사용한 3 Tier 클라이언트/서버 시스템을 제안한다.

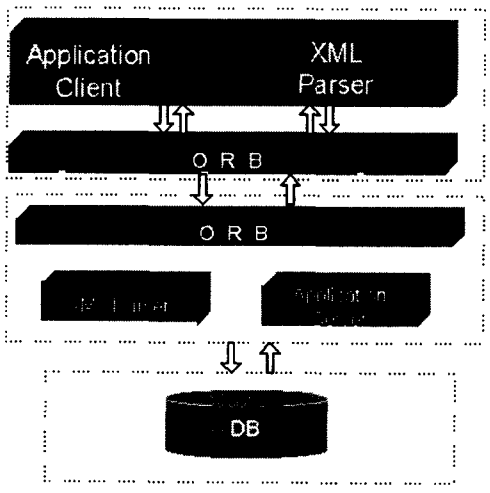

그림 1 시스템 구조

시스템의 3계층은 그림 1과 같이 클라이언트, 미들 서버, 데이터베이스 서버로 사용되며 변경없이 다중 플랫폼들을 운영할 수 있다.

첫 번째 층에서는 다양한 컴퓨터들에서 존재하고 지속되는 자바 클라이언트 응용프로그램을 형성한다면, 클라이언트들은 자바가 가능한 브라우저들에서 애플릿들을 운영 할 수 있다.

두 번째 층은 미들웨어 서버로 구성된다. 서버의 객체들은 비즈니스 논리로 구성되며, 클라이언트들이 서버들로 ORB를 통해서 실시간으로 CORBA에 응답을 보낸다

미들서버는 3층의 데이터저장장치에 끊임없이 접속되고 통합된 인터페이스를 가진 클라이언트를 제공하며, 클라이언트와 미들웨어 사이의 인터페이스와 미들웨어서버와 응용프로그램 사이의 인터페이스를 정의하는 CORBA의 IDL를 사용한다.

세 번째 층은 최종적으로 원하는 기능을 수행할 데이터베이스 서버가 들어간다.

Application Client에서 만들어진 데이터는 어플리케이션 측 XML Parser를 통하여 구조화 되고 정형화된 데이터로 걸러진 후 CORBA ORB 객체에 전해지게 된다. 이렇게 전송된 데이터를 다시 XML Parser를 통해 DB 구조에 적합한 데이터로 변환된 후 역시 Application Server를 사용하여 DB에 저장하게 된다.

2개의 XML Parser와 CORBA ORB를 사용하고 마지막에 DB Server가 위치한 이러한 데이터 처리 구조는 이기종간의 데이터호환문제, 서로 상이한 네트웍에서의 연결문제, 이종의 DB간의 데이터 공유 문제 등 현존하는 수많은 문제들에 대한 가장 이상적인 모델이며 3Tier 구조뿐만 아닌 어떠한 네트웍환경에서도 완벽한 데이터 통합을 위한 모델이다.

3.2 애플리케이션 클라이언트

이 구조의 특징은 클라이언트에게 하나의 일관성 있는 사용자 인터페이스를 통한 다양한 데이터베이스 및 이형의 시스템 자원들의 접근에 대한 투명성을 제공하는 것이다. 즉 클라이언트는 미들웨어 중개자의 구성요소인 로케이션서비스, 변환 서비스 등의 분산 서비스들을 제공받고 CORBA를 통해서 다른 응용 서버들과 통신한다.

3.3 애플리케이션 서버

서버는 XML 파서, 로케이션 서비스, 변환 서비스, 정보 관리자 객체로 구성되며 기능은 다음과 같다.

XML 파서는 문서를 읽고 문서의 컨텐트와 켄텐트를 표현하는데 사용하는 마크업을 바탕으로 결과를 작성하며, XML 문서가 Well-formed의 문서인지 확인하여 문서의 유효성을 검사한다.

로케이션 서비스는 특정 응용 서버에 관한 호스트 및 객체구현 정보를 가지고 있어 클라이언트의 초기 세션 설정시, 응용 서버의 객체가 바인딩 되고, 바인딩된 객체 참조를 가지고 클라이언트는 검색 질의를 요구한다.

변환 서비스는 SQL 문을 XML로 변환을 해주며, XML을 SQL로 변환해주기 위해 중간에 데이터 객체 모델로 바꾸어 주는 역할이 추가하게 되며, 검색 질의 표준 형식으로 변환하거나 검색 결과로 생성되는 출력 형식을 클라이언트가 요구하는 형식으로 반환한다. 즉 서로 다른 정보검색시스템의 실행결과로 변환되는 정보를 표준 또는 일관성 있는 출력 형식으로 변환하여 클라이언트에게는 XML 데이터를 제공한다.

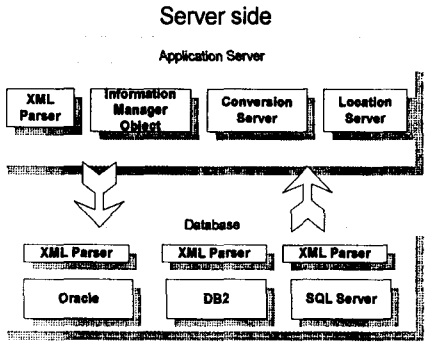

그림 2 서버 시스템 구조

정보관리자 객체는 데이터 객체들의 생성, 복사, 제거 등에 관한 연산을 수행한다. 데이터 객체는 데이터 교환에 필요한 임시 저장소로서 검색요구와 결과 전달시 필요한 자료구조이다. 데이터 객체는 데이터 및 값의 단순 쌍으로 구성된 단순 데이터 객체 구조와 복잡한 레코드 구조를 처리할 수 있는 구조적 데이터 객체 구조로 구분된다.

3.4 통신구조

클라이언트의 검색 질의에 대한 시스템의 정보검색시스템의 연산을 기동하는 과정은 다음과 같다

클라이언트는 일관성있는 검색을 통해 질의를 요구한다.

미들웨어 중개자의 로케이션 서비스는 클라이언트의 요구에 대한 검색 서버의 위치 정보를 찾고 로케이션 서비스에 등록된 객체 구현의 객체 참조를 반환한다.

정보관리자 객체로부터 각 지역에 구축되어 있는 데이터베이스의 검색 인터페이스 정보를 얻는다.

미들웨어 중개자는 로케이션 서비스로부터 얻어진 객체 참조와 검색 연산을 수행하도록 요구한다. 여기서 정보관리자 객체는 검색된 데이터 객체를 생성하고 각 데이터 객체 수만큼의 데이터 리스트를 작성한다.

미들웨어 중개자는 두 정보시스템으로부터 얻어진 검색 결과를 변환 서버를 이용하여 각 레코드들을 합병하고 데이터를 클라이언트에게 전달한다.

클라이언트는 검색 질의 결과를 검색 결과 화면에 디스플레이 한다. 만일 각각의 응용 서버에서 오류가 발생하면 예외 처리를 한다.

서비스를 표현하면 다음과 같다.

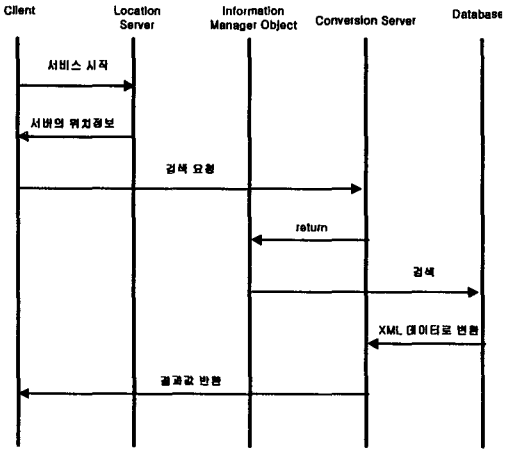

그림 3 시스템 운영 절차

4. XML 문서 처리 및 질의처리

본 시스템은 XML 파일을 보내거나 질의를 보내 원하는 데이터를 받기 위해 내부에는 크게 세 가지의 역할을 하는 함수가 존재한다.

1. XML 문서를 보내기 위해 변환해주는 함수
2. 서버로 ORB의 인터페이스를 통해 XML 파일을 보내는 역할을 하는 함수가 있다. 이 함수의 내부에는 XML 파일을 보낸 후 반환되는 값을 받는 인스턴스가 존재한다.
3. CORBA ORB 인터페이스로부터 받은 XML 파일을 Parsing하여 화면에 뿌려주는 함수가 있다.

표1은 클라이언트와 서버사이에서 데이터를 교환하기 위해 XML문서를 Byte코드화 시키는 함수이다.

```
public static byte[] send(String filename)
{
    FileInputStream file;
    byte[] XMLfile = new byte[];
    try{
        int c;
        file = new FileInputStream(filename);
        byte[] stre = new byte[file.available()];
        while(( c = file.read(XMLfile)) ! = -1) {break;}
      XMLfile = stre;
    } catch () {}
 return XMLfile;
}
```

표 1 code변환 예

XML 파일을 FileInputStream 함수를 이용하여 이를 바이트 단위로 하나하나 EOF를 만날 때까지 읽어들여 byte 배열에 저장한다.

표2는 데이터를 SQL 문을 내장하는 XML 문서를 보내는 함수이다. 첫 번째 인자는 보내게될 XML문서의 이름이고 두 번째 변수는 검색할 데이터베이스의 이름을 받는 부분이다.

이 함수의 내부를 살펴보면 제일 먼저 하는 역할은 보내기를 원하는 XML파일을 앞에서 설명한 send함수를 이용하여 byte 코드화 한다.
먼저 byte 배열화한 XML파일과 코드를 CORBA ORB 인터페이스 함수를 이용하여 서버로 전송하게 된다.

이 함수에 대한 반환값은 반환 받는 파일명과 바이트 배열코드에 저장된다.

```
public String dispatch(String filename, String att)
{
    String stdk = null;
    try {
        XMLfile = send(filename);
        argument[0] = filename;
        argument[1] = ":name="+ att;
        str = argument[1];
        resultFile = SendXML();
        stdk = new String();
     } catch() {}
    try {
        fout = new FileOutputStream();
        fout.close();
    }catch(){}
 return resfile.value;
}
```

표 2 전송 interface 예

5. 결론

인터넷 기술발전에 따른 급속한 사무환경의 온라인 화에 따라, Semi-structured 데이터가 양산되고 있다, Semi-structured 데이터는 비정형, 불명확한 구조, 데이터와 스키마가 혼재되어 있다.

기존의 문서관리 시스템의 제한점인 포맷에 따른 문서의 내용, 구조, 표현의 분리 문제, 메타정보의 자동화 문제, 문서단위의 검색 문제는, XML을 사용하여 해결할 수 있으며, 또 분산컴포넌트 기술을 사용하고 있는 CORBA의 IDL을 이용하여 인터페이스와 구현의 상세를 분리시킴으로써 클라이언트와 서버의 독립적인 개발을 보장하며 IDL로 정의된 표준 인터페이스를 통한 시스템 통합을 가능하게 해준다.

본 시스템은 3-Tier 방식으로 시스템을 구성하여 2-Tier 아키텍처의 단점인 복잡한 업무를 처리하고자 할 때 애플릿의 다운로드 속도 및 업무처리 속도가 느려지는 결과를 미들서버의 어플리케이션 서버의 (정보관리자 객체, 변환 서버, 로케이션 서버 등의 객체들을 통해서 회피할 수 있으며, 서로 다른 하드웨어-플랫폼-데이터베이스 등에 관계없이 원할한 정보검색 및 모든 데이터에 투명한 접근을 보장하도록 개발하였다.

6. 참고문헌

[1] W3C, Extensible Markup Language(XML) 1.0,Recommendation, Feb 1998. (www.w3.org/TR/REC-xml)

[2] Jon Bosak, 1997.3, "XML, Java, and the Future of the Web", http://sunsite.unc.edu/pub/sun-info/standards/xml/why/xmlapps.htm

[3] Richard Light. Pressenting XML. SAMSNET

[4] OMG, "CORBA 2.0 Specification" (1999) http://www.img.org/corba/corbaiiop.htm

[5] J. Siegel (1996), CORBA Fundamentals and Programming, John Wiley & Sons Inc,

[6] 류진영, "관계형 데이터베이스에서 XML 저장 방안 연구 및 구현", 부산대학교 석사 학위 논문