工學碩士 學位論文

XML 基盤의 異機種 DBMS간 데이터 複製 웹 에이전트 設計 및 具現

Design and Implementation of XML based Data Replication Web Agent between Heterogeneous DBMSs

指導敎授 林 宰 弘

2002年 2月

韓國海洋大學校 大學院

電子通信工學科

劉 宣 榮

工學碩士 學位論文

XML 基盤의 異機種 DBMS간 데이터 複製 웹 에이전트 設計 및 具現

Design and Implementation of XML based Data Replication Web Agent between Heterogeneous DBMSs

指導敎授 林 宰 弘

2002年 2月

韓國海洋大學校 大學院

電子通信工學科

劉 宣 榮

本 論文을 劉宣榮의 工學碩士

學位論文으로 認准함.

委員長 : 金 基 文 (印)

委 員 : 梁 圭 植 (印)

委 員 : 林 宰 弘 (印)

2002年 2月

韓國海洋大學校 大學院

電子通信工學科 劉 宣 榮

목 차

Abstract ····· ii

제 1 장 서 론 ····· 1

제 2 장 확장성 생성 언어(XML) ····· 3
2.1 SGML과 HTML의 한계 ····· 3
2.2 XML의 개요 및 특징 ····· 4
2.3 활용분야 ····· 9
2.4. XML의 연구 동향 분석 ····· 11
2.5 XML 기술 현황 ····· 14

제 3 장 웹 에이전트 시스템 설계 ····· 21
3.1 웹 에이전트의 구성 ····· 21
3.2 웹 에이전트의 데이터 교환 과정 ····· 25

제 4 장 웹 에이전트 구현 및 고찰 ····· 31
4.1 데이터베이스와 연결 ····· 31
4.2 데이터베이스의 관리 ····· 37
4.3 데이터베이스 복제 ····· 44

제 5 장 결 론 ····· 49

참 고 문 헌 ····· 51

요 약

초기의 인터넷을 전세계적으로 확산시키는데 큰 역할을 한 하이퍼텍스트 생성 언어(HTML ; Hyper Text Markup Language)는 제한된 태그(Tag)들을 사용하기 때문에 문서를 구조화시키지 못해 정보 축적과 정보추출 방법이 비효율적이고 추출한 정보의 가공이 어렵다.

HTML의 단점을 보안하기 위해 새로운 표준으로 제안된 확장성 생성 언어(XML ; Extensible Markup Language)는 사용자가 문서상에 사용될 태그를 자유롭게 정의할 수 있고 다른 사람들도 그 태그를 사용할 수 있다. 따라서, 웹에서 서로 다른 데이터베이스와 정보를 공유하고자 하는 기업의 요구를 만족시키기 위해서는 표현 위주의 HTML보다는 문서 구조에 대한 정보를 가지고 있는 XML이 더 적합하다.

본 논문에서는 웹에서 이루어지는 이기종 데이터베이스 관리 시스템(DBMS ; Database Management System)간의 데이터 복제와 XML 문서와 데이터베이스간의 정보 교환을 위해 XML을 이용하여 이기종 DBMS간의 정보를 교환할 수 있는 웹 에이전트 시스템을 설계하고 구현하였다. 구현된 시스템은 이기종 DBMS간의 데이터 교환 및 복제를 위한 매개물로 XML을 이용한다.

웹 에이전트 시스템은 서로 다른 데이터베이스간의 정보 교환과 XML 문서를 데이터베이스에 저장시킨다. 정보 교환이 웹 상에서 이루어지므로 데이터베이스 관리를 위한 명령어에 대한 지식이 없는 일반인도 쉽게 다룰 수 있다. 문서에 대한 구조정보를 제공하고, 데이터를 해석할 수 있는 XML을 정보 저장소로 이용하기 때문에 정보 공유와 정보 검색 및 정보 관리까지 손쉽게 할 수 있다.

제 1 장 서 론

오늘날 지식과 정보 교류의 기반이 웹으로 옮겨지고 거의 모든 정보가 웹을 이용해서 전달되면서 기업들도 인터넷을 활용한 정보공유와 경제활동에 초점을 맞추고 있다. 현재 우리의 생활에 가장 많은 영향을 미치고 있는 인터넷을 이용한 정보 제공은 사용자가 웹 브라우저만으로 서비스를 이용할 수 있고, 플랫폼에 상관없이 정보를 이용할 수 있다.

인터넷 보급이 확산되면서 웹 문서의 표준으로 텍스트 기반의 하이퍼텍스트 생성 언어(HTML ; Hyper Text Markup Language)를 이용하였다. HTML은 단순하고 일반인이 사용하기 편리하다는 장점으로 초기의 인터넷을 전세계적으로 확산시키는데 큰 역할을 하였다[1].

현재 웹 정보 저장에 가장 많이 사용되고 있는 웹 문서인 HTML은 초기 출판을 목적으로 탄생된 표준이다. HTML은 정보를 어떻게 화면에 표시할 것인지를 정의할 수 있고 간편하고 사용하기 쉽다.

그러나, 제한된 태그(tag)들로 인하여 구조화되지 못한 문서의 형태를 보이고 있는 HTML은 정보축적의 한계가 있을 뿐만 아니라, 임의의 웹 사이트로부터 생성된 HTML 문서를 분석하여 정보를 추출하는 방법은 비효율적이고 추출한 정보의 가공도 어렵다. 따라서 HTML의 단점을 보안하기 위해 새로운 표준 즉, 새로운 형태의 언어가 필요하게 되었다.

새로운 언어에 대한 필요성이 대두되자, 월드 와이드 웹 컨소시엄(W3C ; World Wide Web Consortium)은 1996년 웹 문서의 표준으로 다양한 기능과 구조적인 표현 능력을 가진 표준 범용문서 생성 언어(SGML ; Standard Generalized Markup Language)에 기반을 둔 확장성 생성 언어(XML ; eXtensible Markup Language)를 제안하였다.

XML은 SGML의 기능성과 구조적인 표현을 지원할 뿐 아니라 사용의 편리성을 강조하였다. 사용자가 문서상에 사용될 태그를 자유롭게 정의할 수 있고 다른 사람들도 그 태그를 사용할 수 있다. 이런 확장성과 편리함 때문에 웹 문서들이 XML로 작성되고 있다[2].

오늘날 웹에서 데이터베이스를 관리하는 기업이 늘어나면서 서로 다른 데이터베이스와의 정보공유에 대한 요구가 증가하고 있다. 데이터베이스에 저장되어 있는 정보를 웹에서 교환하기 위해서는 표현 위주의 HTML보다는 문서 구조에 대한 정보를 가지고 있는 XML이 더 적합하다.

본 논문에서는 웹에서 이기종 데이터베이스 관리 시스템(DBMS ; Database Management System)간의 데이터 복제를 위해서 XML을 이용한 웹 에이전트 시스템을 구현한다.

구현된 시스템은 이기종 DBMS간의 데이터 교환 및 복제를 위한 매개물로 XML을 이용함으로써 교환하고자 하는 데이터를 웹 브라우저를 통하여 확인하거나 수정과 삭제를 할 수 있다. 웹 에이전트는 데이터베이스의 정보교환 뿐만 아니라 XML 문서를 데이터베이스에 저장시킬 수도 있다. 또한 웹 상에서 이루어지는 정보교환이므로 데이터베이스 관리를 위한 명령어에 대한 지식이 없는 일반인도 쉽게 다룰 수 있다는 장점이 있다.

본 논문의 구성은 제 2 장에서 XML의 등장 배경과 개요 및 특성에 대해 분석하며, XML의 활용분야와 연구동향을 정리하여 XML의 기술현황을 분석하였고, 제 3 장과 제 4 장에서는 실제로 구현한 웹 에이전트에 대한 설계와 구현 내용을 기술하고, 제 5 장은 결론과 향후 연구계획에 대하여 기술하였다.

제 2 장 확장성 생성 언어(XML)

2.1 SGML과 HTML의 한계

2.1.1 SGML의 한계

하드웨어 및 소프트웨어의 종류에 관계없이 이기종간의 정보 교환이나 검색, 처리를 지원하는 표준으로 SGML이 제시되었다[3].

SGML이란 국제 표준화 기구(ISO ; International Standards Organization)에서 1986년에 제정된 문서 정보를 기술하기 위해 필요한 요소를 완벽하게 정의한 표준이다. 문서의 다양성을 정의하는 메타 언어(ML ; Meta Language)인 SGML은 소프트웨어 패키지나 소프트웨어 벤더, 하드웨어 기종과 관계없이 자료를 저장하고 정보를 교환하기 위한 개념에서 출발하였다.

SGML은 복잡하고 고가의 툴(tool)을 사용하며, 초기 투자비용이 높고 관련 소프트웨어의 개발이 어렵다. SGML의 한계를 극복하기 위해서 문서 원형 정의(DTD ; Document Type Definition)를 만들어 표현위주로 정의한 것이 바로 HTML이다.

2.1.2 HTML의 한계

1991년 팀 버너스-리(Tim Berners-Lee)라는 학자가 자신의 기술 논문을 쉽게 작성하고 다른 기종을 사용하는 학계의 사람들이 원래의 문서 모양을 그대로 볼 수 있도록 하기 위해 HTML을 만들었다.

1992년에 HTML 스펙 1.0 이 발표된 이후, 현재의 인터넷 혁명을 가능하게 한 HTML도 한계가 있다. HTML은 태그의 사용이 제한

적이고, 새로운 응용프로그램이나 웹 기술을 다루는데 부적절하다. 또한, 웹 상에서 정보를 표현하는 목적으로 설계된 HTML은 태그로 묶인 데이터 필드에 따라 데이터 추출이 어렵다.

이러한 HTML 한계를 극복하기 위해 여러 가지 스크립트 언어나 종속형 시트(CSS ; Cascading Style Sheets), 동적 HTML(DHTML ; Dynamic HTML) 등을 이용하여 한계를 극복하고자 하였으나, 결국 근본적인 문제점을 해결하지 못했다[4].

HTML이 '브라우저간의 호환성 부재'라는 심각한 문제가 발생하게 되자, 새로운 마크업 언어에 대한 필요성을 고려하게 되었다. 이러한 필요성에 의해 등장한 것이 XML이다.

2.2 XML의 개요 및 특징

XML은 HTML과 같은 생성 언어(markup language)이지만, 정확한 뜻은 생성 언어를 정의하기 위한 언어이다.

XML이 HTML과 다른 점은 태그를 정의하고 데이터를 기술할 수 있다는 것이다. XML은 데이터의 관점에서 기술하고 구조화하며, HTML은 처리된 데이터를 표현하는 관점에서 이해할 수 있다. 즉 XML은 HTML의 대체품이 아니라 HTML의 한계를 극복한 것이다.

2.2.1 XML의 개요

1996년 4월 팀 브레이(Tim Bray)는 구조화된 문서를 위한 XML의 설계목표를 발표했고, 1996년 11월 보스톤에서 열린 SGML '96 컨퍼런스에서 처음 XML 초안이 발표되었다. 1998년 2월 10일에는

W3C 권고에 따르는 XML 스펙 1.0이 발표되고 오늘날까지 꾸준히 연구되어 오고 있다[5],[6].

웹에서 구조화된 문서를 전송 가능하도록 설계된 XML은 표준화된 텍스트 형식의 생성 언어로 SGML의 일부분이며 SGML보다 간결하고 인터넷에서 사용 가능한 문서를 표현하는 표준이다.

XML은 전체를 지원하는 소프트웨어의 개발이 용이하지 않는 SGML과 제한된 태그로만 분류되어 지정되지 않은 태그의 사용이 불가능하다는 HTML의 단점을 극복하였다. 그리고 XML은 구조화된 문서를 정의하고 자유롭게 태그를 정의할 수 있는 SGML과 인터넷에서 손쉽게 하이퍼미디어 문서를 제공할 수 있는 HTML의 장점을 그대로 취합하고 있다.

HTML과 SGML의 필수적인 기능만을 취합하고 복잡하고 어렵거나 비효율적인 부분은 제외함으로써, XML은 두 언어의 핵심적인 장점을 그대로 보유하고 있다. 이와 같은 XML과 HTML, SGML의 특성을 비교해 보면 각각 <표 2-1>과 같다.

2.2.2 XML의 특징

XML은 웹에서 정보를 전달하기 위하여 제안된 메타 언어이다. XML을 위한 소프트웨어는 URL(Uniform Resource Locator)을 HTML의 링크를 접근하듯이 접근할 수 있는 기능이 내재되어 있다.

일반적으로 XML 인스턴스 중에서 링크와 DTD는 URL의 사용이 가능해야 한다. 만약, 파서 및 기타 소프트웨어에서 URL을 지원하지 않는다면, XML 스펙에서 언급한 다양한 기능들이 무용지물이 된다. XML을 지원한다는 것은 인터넷 환경을 고려한다는 의미가 되므로 XML은 웹에 대한 활용이 높다.

<표 2-1> HTML/SGML/XML의 비교

<Table 2-1> Comparison of HTML/SGML/XML

항목	HTML	SGML	XML
태그	내장된 DTD정의 사용자정의 불가능 (한정된 태그만 사용)	사용자정의 태그사용	사용자정의 태그사용
문서재사용	불가능	가능	가능
응용	문서의 표현	복잡한 문서 구조 방대한 내용의 문서 (메뉴얼, 공공분야, 출판분야 등)	SGML과 동일 웹에서 정보교환
난이도	쉬움	복잡하고 어려움	비교적 쉬움 (SGML의 단순화)
검색	검색 어려움 (검색 엔진 필요)	자료의 표현과 내용이 분리되어 정확한 검색	SGML과 동일
출력형식	CSS	DSSSL (Document Style Semantics and Specification Language)	XSL (eXtensible Stylesheet Language)
데이터 교환	교환시 방대한 작업 (부적합한 교환포맷)	표현부와 분리부가 분리되어 교환이 용이	SGML과 동일

XML에서는 기본적인 XML 문법에 맞게 인스턴스가 작성되어 있다면 DTD가 필요 없는 정형화된(well-formed) 문서가 있다. XML을 보다 효율적으로 사용하기 위해서 DTD를 작성하고, DTD에 따라 인스턴스를 생성해야 하는 과정은 초보자에게 문서 작성에 대한 부담감을 주게 된다. DTD가 웹 브라우저에 내장된 HTML과 달리, XML은 사용자에 따라 DTD의 존재여부가 결정된다. 반드시 DTD 규칙을 따라야 하는 HTML보다 XML이 문서 작성하는데 용이하다.

XML의 스타일 언어인 XSL(eXtensible Stylesheet Language)은 사용자가 정의하는 문서에 대한 표현을 적용하는 메커니즘을 제공

한다. 즉, XSL은 하나의 내용에 대하여 다양한 형식으로 표현할 수 있도록 해 준다.

예를 들어, 아래한글로 만든 제품 설명서를 웹이나 매뉴얼로 사용하기 위해서는 프로그래밍 작업을 다시 해야 한다. 즉, 동일한 내용일지라도 사용되는 환경에 따라 또는 표현형식에 따라 새로 작성을 해야 한다. 반면에, XSL을 이용하여 XML 문서의 내용을 표현하면, XML 문서 자체에 대한 수정작업 없이 XSL만 수정하면 된다.

XML은 데이터를 고유 태그로 표시하기 때문에 좀더 정확하고 빠른 검색을 제공할 수 있다. 예를 들어, 책에 대한 검색의 경우 작가, 제목, 또는 다른 분류항목 등에 의한 표준방식으로 책을 손쉽게 분류할 수 있고 사용자가 필요한 부분만 세분화하여 검색할 수 있다.

2.2.3 XML의 처리과정

XML은 웹에서 사용 가능하고, 언어에 대해 독립적이고 여러 응용 프로그램을 지원하며, 유연하고 개방적인 표준 기반 형식으로 뛰어난 상호운영성을 제공한다. XML은 SGML에서 파생되었기 때문에 SGML과 호환이 가능하고 프로그래밍이 용이해서 문서를 처리하는 프로그램의 작성이 쉽다.

또한, DTD가 보이지 않는 HTML과 달리, XML은 필요에 따라 DTD를 정의하거나 생략할 수 있다. 링크 메커니즘의 강화로 다양한 형태의 정보와 연결할 수 있고 유니코드 기반의 UTF(UCS Transformation Format)-8 및 UTF-16을 지원하므로 외국어를 표현하는데 용이하다.

이와 같은 XML의 구성을 살펴보면 <표 2-2>와 같다.

<표 2-2> XML 구성 요소

<Table 2-2> XML construction element

항목	표준
내용(contents)	XML
구조(schema)	DTD, Schema
접근(access)	DOM, Parser
변환(view/transaction)	XSL, XSLT(XSL Transformations)
획득	XQL(XML Query Language)
링크(link)	Xlink(XML Linking Language), Xpointer(XML Pointer Language)

XML 문서의 처리과정은 <그림 2-1>과 같이 이루어진다.

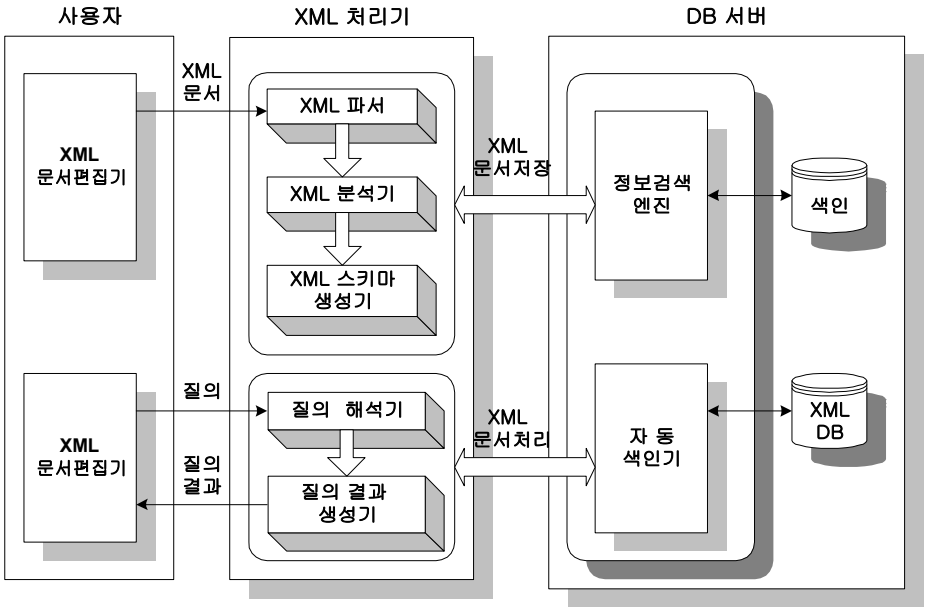
사용자가 XML 문서편집기를 이용하여 작성한 문서를 XML 처리기로 전송한다. XML 처리기는 문서를 저장하거나 XML 문서의 내용을 검색하는 기능이 있다.

XML 문서를 저장할 경우, XML 처리기는 XML 파서를 이용하여 XML 스펙 기준에 따라 문서를 파싱하고 XML 분석기는 파싱된 문서를 문서 객체 모델(DOM ; Document Object Model) 트리 구조로 변환시킨다.

XML 스키마 생성기는 DOM 트리 구조를 토대로 DTD를 작성하여 색인 데이터베이스에 저장한다. 데이터 형식을 일원적으로 정의하고 관리하는 DTD는 데이터를 교환할 경우 유용하게 쓰인다.

XML 데이터베이스의 내용을 검색할 경우, XML 질의 언어인 XQL(XML Query Language)을 이용하여 사용자는 XML 처리기로 질의한다. 질의를 받은 XML 처리기는 XQL을 해석하는 질의 해석기를 이용하여 사용자가 요청한 질의를 파악한 다음 자동 색인기를 이용하여 XML 데이터베이스에 저장된 정보 중에 사용자가 원하는 정보를 XML 처리기로 보내준다.

그리고, 데이터베이스 서버로부터 질의에 대한 응답을 받은 XML 처리기는 질의 결과 생성기를 통해 사용자가 원하는 정보를 XML 문서의 형태로 만들어 사용자에게 전송한다.



<그림 2-1> XML 문서 처리 과정

<Fig. 2-1> Management process of XML document

2.3 활용분야

2.3.1 전자상거래

인터넷 사용자가 늘어나고 인터넷 기술이 발달함에 따라 웹을 이용한 전자상거래가 빠른 속도로 증가하고 있다. 특히 전자상거래는 금융 거래와 밀접한 분야이다.

HTML을 사용할 경우 사용자는 가격 및 세금 등과 관련된 자료를 찾기 위해서 HTML 정보를 가지고 일일이 검색해야 하고, 검색된 문서 및 숫자는 고유의 문맥을 가지지 않는다. 즉 검색된 숫자가 가격인지 세금을 뜻하는 것인지 주소 및 그 외 어떤 의미를 가지는지를 알 수 있는 방법이 없다.

XML은 HTML과는 달리 이러한 점을 해결해 줄 수 있으며 전자상거래의 무한한 가능성을 보여줄 수 있는 촉매 역할을 한다. XML을 지원할 수 있는 저장소의 질의 기능은 적접적으로 관련된 정보를 검색하고 자동적인 에이전트나 사용자에 의해 처리될 수 있게 한다.

2.3.2 인터넷 검색 엔진

오늘날 웹에 등록된 대부분의 검색 엔진에서 메타 데이터에 포함된 정보를 검색하는 전형적인 검색 방법을 사용하면 수백, 수천 개의 관련 자료가 나타난다. 이는 검색 엔진이 제한된 수의 HTML 태그 때문에 내용을 차별하는 방법을 가질 수 없기 때문이다.

HTML은 문서 내용 전체를 기반으로 하는 검색 또는 메타 태그가 포함된 정보를 기반으로 하는 검색을 수행한다. 메타 태그는 자체 디스플레이를 위해 사용되지 않는 HTML의 예약 태그로서, 필요할 때 사람들이 사용하는 일종의 자리를 잡아주는 역할을 한다.

XML을 사용한 검색 엔진은 특별한 태그의 내용을 기반으로 검색할 수 있고, 문맥적인 정보를 이해할 수 있으므로 검색 속도와 정확성이 향상될 수 있다.

웹 서버에서 XML 저장 기술을 사용한다면, 인터넷 사용자에 의한 정보의 중복 과부하를 해소시켜 줄 수 있다.

2.3.3 전자 자료 교환

전자 자료 교환(EDI ; Electronic Data Interchange)이란 기업간의 표준화된 전자문서를 교환함으로써 상거래 업무를 처리할 수 있는 기술이다. 현재 EDI 데이터의 교환만을 처리할 수 있기 때문에 사용자 인터페이스가 부족하고, 실제 거래에서 필요하게 되는 업무 처리의 반영, 제품 설명서, 명세서 등과 관련된 정보 제공이 부족하다.

따라서, 이를 극복하고 보다 효율적인 EDI를 구현하기 위해 많은 곳에서 XML 기반의 EDI를 고려하고 있다. 왜냐하면, XML을 활용하면 보다 효율적인 인터페이스를 제공할 수 있으며 웹에서도 구현이 가능하기 때문이다.

이와 같이, XML의 활용분야는 주로 전자문서를 다루는 시스템에서 강점을 보이고 있다. 이외에도 전자도서관, 전자교본, 지식관리시스템, 기업정보포털, 고객관계관리 등에서도 활발하게 도입될 예정이며, 응용분야는 시간이 경과할수록 더욱 더 확대될 것으로 전망하고 있다.

2.4 XML의 연구 동향 분석

인터넷의 확산과 다양한 정보의 등장으로 인해 효율적으로 정보를 관리하고 저장할 수 있는 정보 시스템의 요구가 증가하고 있다.

대부분의 상업적인 데이터가 관계형 데이터베이스 시스템에 저장되어 있는 현재 이러한 요구를 만족시키기 위해서는 기존의 데이터베이스 관리 시스템과 연동하여 다양한 플랫폼과 이질의 데이터에 관계없이 저장과 관리가 용이한 시스템의 구축이 필요하다[7].

W3C에서 제안한 XML 표준안이 발표된 이후 인터넷 기반의 다양한 분야에서 XML 기술을 활용하고자 하는 움직임이 활발하다.

2.4.1 XML과 데이터베이스의 연동

XML은 문서에 대한 구조정보를 제공하고 XML 태그는 데이터를 해석하는데 사용되기 때문에 데이터로서 XML 역할에 대한 중요성이 인식되고 있다. 이에 따라, 구조 정보를 내포하고 있는 XML 문서를 효과적으로 관리하고 검색하기 위한 연구가 진행되고 있다.

XML 문서를 저장하고, 질의처리를 하기 위한 하부 저장소로 파일 시스템, 관계형 데이터베이스 관리 시스템(RDBMS ; Relational DataBase Management System), 객체지향형 데이터베이스 관리 시스템(OODBMS ; Object-Oriented DBMS), 그리고 반구조적 데이터에 대한 고유의 시스템을 바탕으로 설계될 수 있다.

관계형 데이터베이스에서 DTD 기반의 XML 문서 저장 구조를 설계하기 위해서 DTD를 데이터베이스 테이블로 사상시켜야 한다. 이를 위해 XML 데이터를 저장하는 방법과 관계형 테이블로 사상시키는 방법에 대한 연구가 진행되고 있다[8].

관계형 모델을 사용하여 XML 데이터를 저장하는 연구는 반구조적 데이터를 분석해서 관계형 데이터 모델로 변환하는 방법과 주어진 DTD를 바탕으로 XML 엘리먼트와 관계 테이블 사이의 사상이 자동으로 이루어지게 하거나 사용자나 시스템 관리자가 사상하는 방법을 제공해 주는 방법으로 나누어 진행되고 있다.

XML DTD를 관계형 테이블로 사상시키는 방법은 DTD에서 각 엘리먼트와 애트리뷰트를 테이블과 테이블의 필드에 어떻게 매핑시키는가에 따라 다르다. 매핑 방법에 따른 효율성을 검증하기 위해서 많은 실험이 이루어지고 있다.

객체지향 데이터베이스에서는 질의 결과를 XML 데이터로 변환하는 연구가 진행되고 있다.

객체지향 데이터베이스에 저장된 데이터의 XML로의 변환 과정은 데이터베이스 스키마의 DTD로의 변환 부분과 실제 저장된 데이터의 XML 문서로의 변환 부분으로 나눈다[9].

객체지향 데이터베이스의 스키마와 데이터, 데이터 변환의 결과가 될 DTD와 XML 문서를 정의하고, 이 정의에 따라서 두 가지 변환 알고리즘을 제시하기 위한 연구가 진행되고 있다.

2.4.2 데이터 저장관리를 위한 연구

분산 환경에서 각자의 고유한 데이터베이스 스키마를 가지고 운영되는 기존의 정보 시스템들을 XML이라는 공통 데이터 모델을 사용하여 표현함으로써 저장 및 관리 측면에서 효율성을 증대하고자 한다. 이를 위해 기존의 데이터베이스 구조를 이질의 플랫폼에 영향을 받지 않고 XML을 매개로 하여 저장 및 관리가 가능한 시스템을 구축하고자 하는 연구가 진행되고 있다.

이러한 연구는 기존의 데이터베이스 관리 시스템의 스키마 구조를 XML로 변환하는 방법과 변환된 XML 파일에서 기존의 데이터베이스 시스템처럼 조작 및 정의 기능이 가능하도록 하는 XML DTD에 유효한 정의가 필요하다[10],[11].

B2B(Business to Business) 시대에는 XML 문서의 양도 증가하기 때문에 XML 문서를 저장, 관리할 XML 저장관리 기술과 데이터베이스 관련 아키텍쳐를 XML 문서로 변환하는 기술이 필요하다.

이러한 데이터베이스 관련 아키텍쳐를 데이터 형태로 저장하기 위해서 XML 문서의 구조적 정보를 토대로 데이터베이스에 그 형태대로 저장하는 방식에 대한 연구가 진행되고 있다[12][13].

2.4.3 문서관리시스템에 대한 연구

기존의 구조문서를 저장, 관리 및 검색을 지원하는 문서관리시스템의 연구는 주로 DBMS를 활용하는 방법을 사용한다. 구조 문서의 논리적인 구조적 특성을 다루기 위한 저장 시스템으로서 기존의 DBMS를 사용하는 것은 한계가 있지만 다른 정보 저장시스템에 비해 안정적이고 DBMS의 다양한 기능을 활용할 수 있고, 널리 사용되고 있다는 장점이 있다[14].

구조 문서의 구조정보와 함께 문서를 저장하고 다양한 검색을 지원하는 방법에 대한 연구가 진행되고 있다[15].

문서를 저장하는 방법은 문서를 구성하고 있는 엘리먼트를 나누어 저장하는 분할 모델과 문서 전체를 저장한 후 각각의 엘리먼트 위치 정보를 가지고 접근하는 비분할 모델, 그리고 두 가지 모델을 혼용한 혼합모델로 나누어 연구되고 있다.

2.5 XML 기술 현황

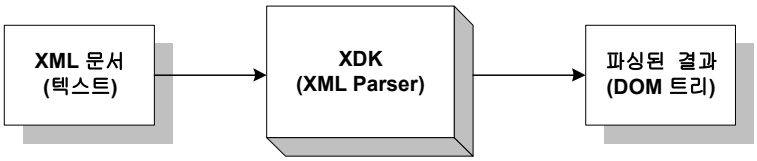
2.5.1 오라클(Oracle)

오라클에서는 차기 데이터베이스 버전에 XML을 객체 타입처럼 기본형으로 제공할 계획이다[16].

XML 개발 툴인 XDK(XML Developer's Kit)는 XML 문서를 읽고, 조작하고, 변환하는 기능이 있다. XDK는 여러 형태의 개발에 사용하기 위해서 자바, C, C++, 절차 지향 언어(PL ; Procedural Language)/구조적 질의어(SQL ; Structured Query Language)로 제

작되었다. XDK는 DOM과 SAX(Simple API for XML) 인터페이스를 사용하여 XML을 스펙에 맞게 파싱하는 XML 파서가 기본 툴로 <그림 2-2>와 같다.

XDK는 XML 파서를 이용하여 XML 문서를 받아들이고 W3C에서 표준으로 정한 DOM의 트리 구조로 파싱해 준다.

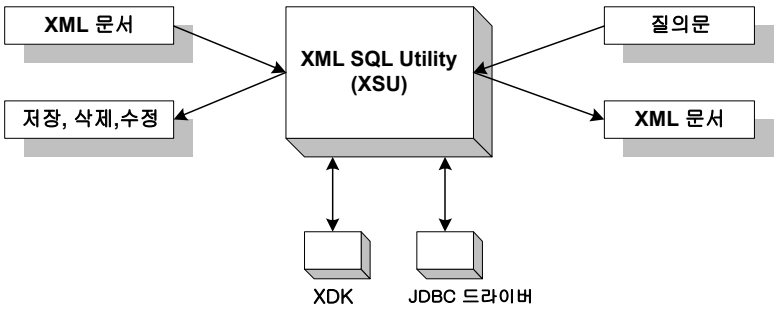


<그림 2-2> XDK 구성

<Fig. 2-2> XDK configuration

XDK와 JDBC(Java Database Connectivity) 드라이버를 포함하는 XSU(XML SQL Utility)는 <그림 2-3>과 같다.

질의어의 결과를 XML 문서로 반환해 주고, XML 문서를 저장, 삭제, 변경하는 기능을 담당한다. 또한 JDBC 드라이버를 통한 질의에 대해 XML 문서의 형태로 결과를 보여준다.



<그림 2-3> XSU 구성

<Fig. 2-3> XSU configuration

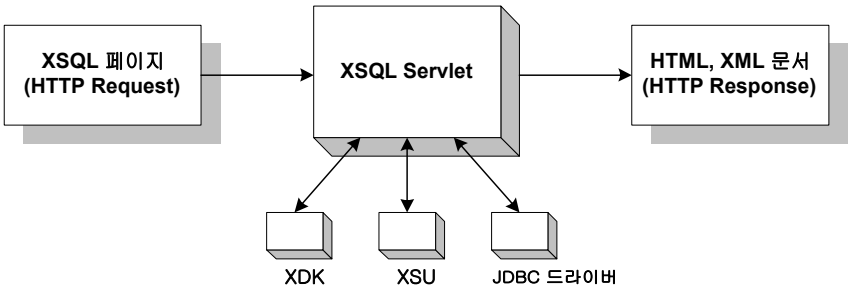
오라클에서 제공하는 자바를 위한 XSU는 오라클 8i를 데이터베이스 뷰어에 한 계층을 더 두어 XML 뷰어로 볼 수 있도록 지원하는 제품이다.

개발자는 SQL을 오라클 8i로 전달하면, XML로 질의의 결과가 반환되고, XML을 오라클 8i에 저장하면, 데이터베이스 테이블로 사상되어 저장된다. 또한, XML을 수정, 삭제 연산과 함께 오라클 8i로 적용할 수도 있다. 따라서, JSP(Java Server Page), 서블릿(servlet)과 함께 자바를 위한 XSU를 자바 빈으로 구성하면, XML 사이트를 구축할 수 있다.

XSQL 서블릿은 XML에 SQL을 포함하여 하나의 페이지로 구성된 요청을 처리하는 제품으로 web-to-go라는 웹서버와 함께 동작한다. 개발자는 XML에 SQL을 포함시키고, 결과로 반환되는 XML에 적용할 XSL을 작성함으로써 웹 사이트를 구성할 수 있다.

XSQL 서블릿은 XDK, XSU 그리고 JDBC 드라이버를 포함한다. XML 형태인 XSQL 페이지는 SQL을 내장하는 XML 문서이고, XSQL 서블릿 엔진은 해당 페이지의 요청이 들어오면, XSU를 이용하여 SQL문을 XML문으로 바꾼다. 즉, 모든 페이지가 XML 문서로 바뀌는 것이다. 이 XML 문서는 XML 파서를 이용하여 XSL 문서를 적용한 문서를 요청한 디바이스에 반환한다. 이와 같은 XSQL은 <그림 2-4>와 같다.

XML은 반구조적인 데이터 모델이며, 트리 구조 형식으로 이루어져 순서 정보와 반복 정보가 있다. 이와 반대로, 관계형 데이터베이스는 구조적인 데이터 모델이고, 관계 대수이므로 비순서적이며 비반복적이다. 따라서, XML 문서를 관계형 데이터베이스로 사상하여 저장하게 되면 데이터 손실 문제가 생긴다. 오라클 데이터베이스는 객체관계형 데이터베이스이기 때문에 데이터 손실에 의한 결점은 없다.



<그림 2-4> XSQL 서블릿 구성

<Fig. 2-4> XSQL servlet configuration

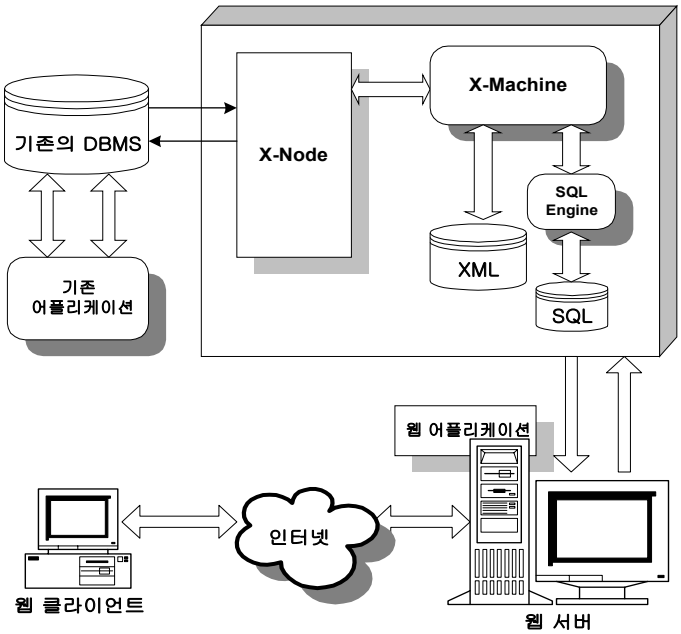
그러나, XSU의 위치가 오라클 데이터베이스에 존재해야 하는 한계가 있다. 오라클 데이터베이스와 다른 데이터베이스간의 데이터 교환이 용이할 수 있지만, 다른 데이터베이스간의 데이터 교환은 불가능하다. 이는 웹에서 이루어지는 모든 종류의 데이터베이스간의 정보교환을 만족시킬 수 없다는 치명적인 문제점이 있다.

2.5.2 펜타 시스템

펜타 시스템에서 제공하는 타미노(Tamino)는 XML 문서를 순수한 표준 XML 포맷으로 저장, 검색하는 정통 XML 데이터베이스 서버이다. 타미노 시스템의 전체 구성은 <그림 2-5>와 같다.

<그림 2-5>에서 보는 타미노의 전체 시스템은 X-Machine, 데이터 맵, X-Node, 타미노 관리자, 타미노 SDK(Software Developer's Kit), 타미노 서버 익스텐션으로 구성되어 있다.

타미노 관리자는 사용자가 타미노 서버 이용시 제일 먼저 접하는 부분으로서 표준 웹 브라우저에서 실행되는 그래픽 사용자 인터페이스(GUI ; Graphic User Interface)를 제공하고 데이터베이스 관리의 모든 형태를 포함한다.



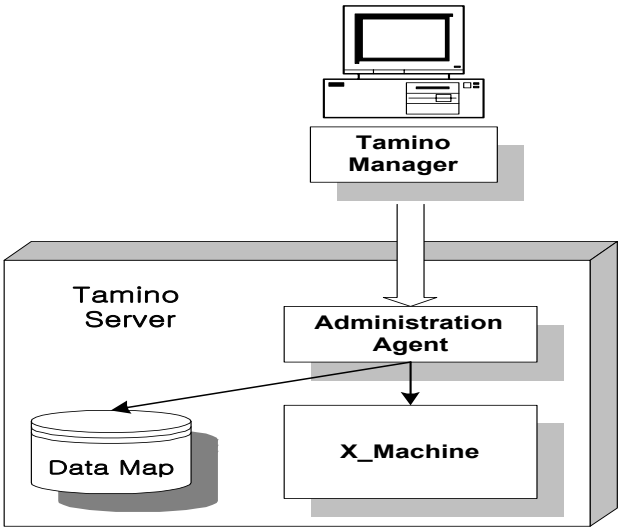
<그림 2-5> 타미노 시스템 구성

<Fig. 2-5> Tamino system configuration

<그림 2-5>와 같이 데이터베이스와 X-Machine을 연결하는 X-Node는 서로 다른 성질의 비즈니스 데이터에서부터 통합된 형태로 표시되는 자료까지 모든 형태의 데이터로 접근이 가능하고, ODBC(Open DataBase Connectivity)를 통하여 기존 데이터베이스의 자료를 이용할 수 있다. 입력된 문서는 타미노의 데이터 맵에서 정의되는 스키마에 따라 내부와 외부 데이터베이스에 보관된다.

타미노 서버 익스텐션 유지를 위해 기초적인 관리 기능을 가진 관리 에이전트 컴포넌트를 사용하는 타미노 관리자는 <그림 2-6>과 같이 구성되어 있다.

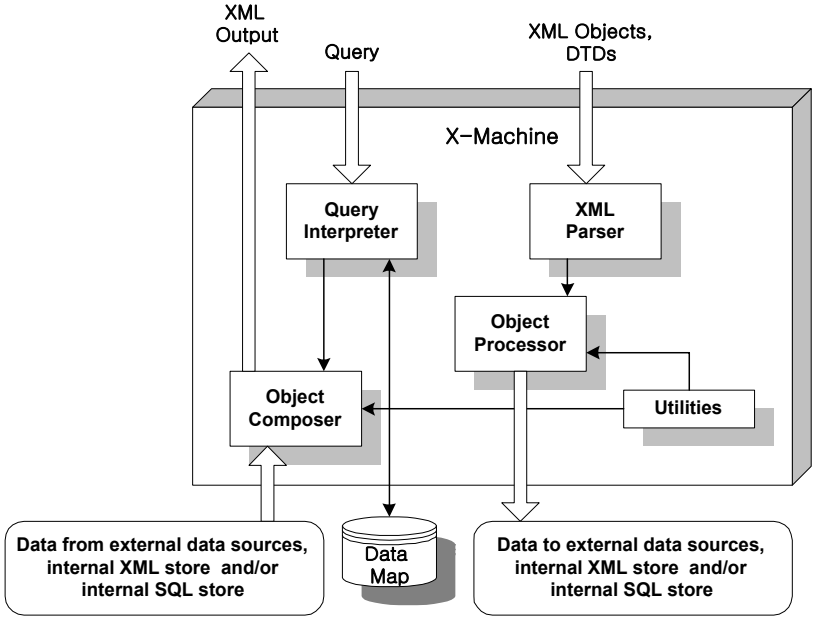
저장된 XML 문서의 데이터 소스를 검색하는 기능을 담당하는 X-Machine은 질의 인터프리터, XML 파서, 객체 Composer, 객체 처리기, 유틸리티로 구성된다.



<그림 2-6> 타미노 관리자 구성

<Fig. 2-6> Tamino manager configuration

X-Machine의 구성은 다음 <그림 2-7>과 같다.



<그림 2-7> X-Machine 구성

<Fig. 2-7> X-Machine configuration

<그림 2-7>과 같이 XML 파서는 스키마의 정당성을 점검하고, 입력되는 XML 문서가 정형화된 것인지 여부를 확인한다. XML 질의 인터프리터는 타미노의 질의 언어인 XQL의 리퀘스트를 해결하고, 객체 Composer와 상호 작용하여 데이터 맵에 저장하게 되는 스키마를 통해 XML 문서들을 검색한다. 객체 Composer는 문서를 검색할 때 정보 객체를 작성하고, XML 문서에서 정보 객체를 불러내는 역할을 한다.

계층적 구조를 갖는 XML 데이터의 순수 스토리지 기술을 채용한 최초의 데이터베이스 서버인 타미노는 XML 문서를 별도의 변환과정 없이 XML 구조 그대로 저장하기 때문에 복잡한 구조의 XML 문서의 저장 검색에 높은 성능을 보장하는 정통 XML 데이터베이스 서버이다.

최상위 계층인 타미노 SDK는 현재 윈도우 NT 플랫폼에서만 이용 가능하고, 클라이언트측과 서버측 모두 타미노에 접근할 수 있는 모듈과 라이센스를 모두 구매하여 사용해야 한다. 타미노는 데이터베이스에 대한 직접적인 관리보다는 XML 문서 자체를 관리하는 데 더 역점을 두고 있는 제품이다[17],[18].

본 논문에서 구현한 웹 에이전트 시스템은 오라클 제품과 펜타 시스템의 타미노가 가지고 있는 단점을 보완한 것이다.

구현된 웹 에이전트 시스템은 데이터베이스간의 정보 교환을 하기 위해 오라클 데이터베이스를 사용해야 하는 오라클 제품과는 달리 데이터베이스의 종류와 무관하게 데이터를 교환할 수 있다. 그리고, 서버에 접속하기 위한 모듈과 라이센스를 구매해야 하는 펜타 시스템의 타미노와는 달리, 인터넷을 사용할 수 있고 정보 교환을 하고자 하는 데이터베이스에 대한 접근 권한만 있으면 언제, 어디서든지 정보 교환을 할 수 있다는 점에서 더 효율적이다.

제 3 장 웹 에이전트 시스템 설계

XML이 웹 문서 표준으로 자리잡아감에 따라, 최근 XML 문서를 데이터베이스 시스템을 이용해 저장, 검색하고자 하는 연구가 활발히 진행되고 있다. 데이터베이스 시스템에 저장된 방대한 데이터를 어떻게 웹 상의 데이터로 표현할 것인가에 대한 연구도 필요하다. 즉, 기존의 데이터베이스 시스템에 저장된 데이터를 끌어내어 XML 형식으로 변환시켜 웹 상에서 제공한다면 현재 존재하는 정보를 보다 효율적으로 사용할 수 있는 기반을 마련하게 되는 것이다.

제 2 장에서 언급한 것처럼 이미 여러 회사에서 연구가 진행되어 제품으로 출시되고 있지만 완벽한 해결안을 제시하고 있는 것은 아니다.

따라서, 본 논문은 사용자의 입장에서 별도의 다른 소프트웨어를 추가하지 않아도 웹에서 데이터베이스의 데이터를 저장, 검색, 관리하고, XML을 이용하여 서로 다른 데이터베이스간의 데이터 복제와 관리가 가능한 웹 에이전트를 구성하였다.

3.1 웹 에이전트의 구성

3.1.1 웹 에이전트 시스템 환경

웹 에이전트에서 사용자는 자바를 지원하고, XML 문서를 볼 수 있는 익스플로러 4.0 이상, 넷스케이프 4.0 이상의 브라우저를 사용한다. 이는 서버와 클라이언트간의 별도의 소프트웨어를 설치하여 통신하는 타미노와는 달리 사용자가 자바와 XML이 지원되는 브라

우저를 통해 웹 서버에게 질의한다.

웹 서버는 웹 프로그램의 솔루션으로 자바와 연동이 되는 JSP를 이용한다. JSP로 개발이 되면 다른 환경으로 큰 변경 작업 없이 이식되고, 속도 향상은 상황이나 플랫폼에 따라 다르지만 프로세스를 생성하는 것보다 50배 이상 빠르다. JSP는 컴파일 과정에서부터 서블릿을 불러들이는 과정까지를 JSP 컨테이너가 처리해 주기 때문에 많은 부하를 줄일 수 있다. 따라서, 본 논문에서 웹 서버는 서블릿이 아닌 JSP를 이용하기로 했다.

JSP도 자바의 한 부분이므로 데이터베이스와 프로그램을 연동시키기 위해 자바에서처럼 JDBC(Java DataBase Connectivity)를 사용해야 한다.

JDBC는 자바나 JSP 등으로 작성된 프로그램을 데이터베이스의 데이터들과 연결하기 위한 응용프로그램 인터페이스(API ; Application Program Interface)이다. 이런 JDBC는 비주얼베이직이나 ASP의 ODBC와 유사한 형태로 모든 데이터베이스와 자바 프로그램을 연결시켜주는 연결자 역할을 한다. JDBC 드라이버에는 JDBC-ODBC 브리지 드라이버, Native-API partly 자바 드라이버, JDBC-Net 순수 자바 드라이버, Native protocol 순수 자바 드라이버가 있다.

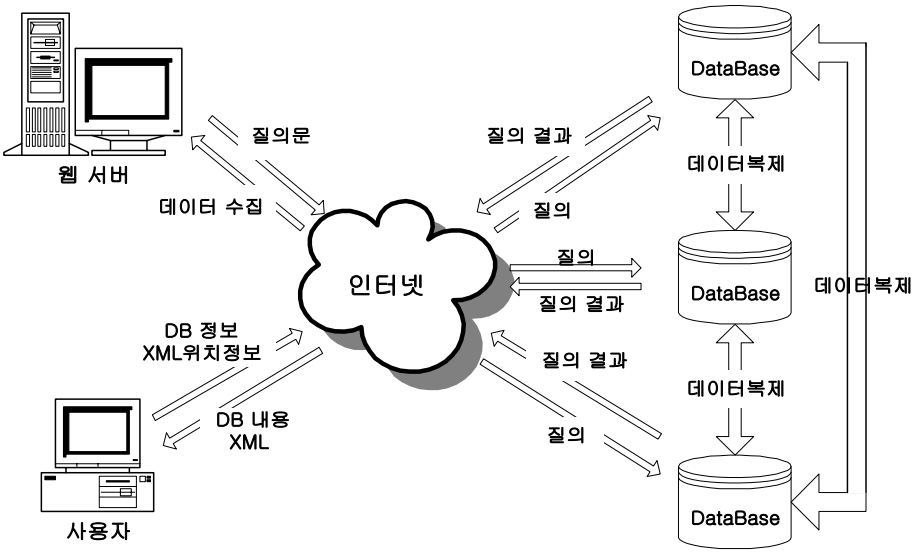
본 논문의 웹 서버에는 데이터베이스마다 전용 드라이버를 설치하여 속도나 성능을 향상시키고, 환경에 구애받지 않고 폭넓게 사용할 수 있는 Native protocol 순수 자바 드라이버를 이용한다.

3.1.2 웹 에이전트 구성

본 논문에서 구현한 이기종 DBMS간 데이터 복제를 위한 웹 에이전트 시스템은 기존의 데이터베이스에서 이루어지던 데이터의 저장 및 삭제, 수정 등의 관리를 웹에서 동일하게 할 수 있다.

그리고, XML을 중간 매개물로 이용하여 서로 다른 DBMS간의 데이터 교환을 가능하도록 한 것이 본 논문에서 구현한 웹 에이전트의 주요한 기능이다.

웹 에이전트의 전체 시스템 구성은 <그림 3-1>과 같다.



<그림 3-1> 웹 에이전트 시스템 구성

<Fig. 3-1> Configuration of web agent system

<그림 3-1>과 같이 구성된 웹 에이전트는 인터넷을 통하여 사용자와 웹 서버간, 웹 서버와 데이터베이스간의 정보를 교환하고, 사용자는 웹 브라우저를 통하여 웹 서버가 보여주는 결과를 확인한다.

웹 에이전트 시스템은 웹에서 데이터베이스를 관리하는 과정과 XML 문서와 데이터베이스간의 정보를 교환하는 과정, 그리고 데이터베이스간의 데이터 복제과정으로 나누어진다.

웹에서 데이터베이스를 관리하는 경우, 사용자는 웹 브라우저를 통해 데이터베이스에 대한 정보를 입력하여 웹 서버로 전송한다. 웹 서버는 사용자가 원하는 데이터베이스와 연결하기 위한 질의문을

완성하여 데이터베이스로 전송한다.

질의문을 전송받은 데이터베이스는 사용자가 입력한 아이디와 패스워드를 확인하여 웹 서버와 데이터베이스를 연결한다. 웹 서버와 데이터베이스의 연결이 이루어지면 데이터베이스의 데이터 관리를 위한 추가정보를 사용자가 입력한다. 사용자가 입력한 정보를 이용하여 웹 서버가 질의문을 완성하고, 완성된 질의문을 데이터베이스로 전송하여 데이터베이스에 저장된 데이터를 추가, 삭제 및 수정과 같은 작업을 한다.

XML 문서를 데이터베이스에 저장할 경우, 사용자는 XML 문서의 위치에 대한 정보를 입력한다. 웹 서버는 입력된 정보를 이용하여 XML 문서에 접근하여 XML 문서를 파싱하고, 파싱된 XML 문서의 내용을 사용자가 보기 편하도록 테이블 형태로 만들어 보여준다.

웹 서버가 보여주는 자료를 바탕으로 사용자는 XML 문서의 내용을 추가, 삭제 및 수정작업을 한다. 수정이 끝난 XML 문서의 내용을 데이터베이스에 저장하기 위해서 사용자는 연결하고자 하는 데이터베이스에 대한 정보를 입력한다. 웹 서버는 입력된 정보를 이용하여 데이터베이스와 연결하고 XML 파일의 이름과 동일한 테이블을 만들어 XML 문서의 내용을 저장한다.

서로 다른 데이터베이스간 데이터를 복제할 경우, 사용자는 복제하고자 하는 데이터를 가진 소스 데이터베이스에 대한 정보를 입력한다.

입력된 정보를 이용하여 웹 서버는 소스 데이터베이스와 연결한다. 소스 데이터베이스와 연결이 이루어지면, 사용자가 복제하고자 하는 테이블을 선택한다. 웹 서버는 선택된 테이블의 내용을 소스 데이터베이스로부터 전달받고, 그 내용을 XML 문서로 저장한다. 이때, XML 문서의 이름은 “테이블이름.xml”이고, 사용자가 지정한 위치에 저장된다.

사용자가 복제한 데이터를 저장하고자 하는 목적지 데이터베이스에 대한 정보를 입력하여 웹 서버로 전송하면, 웹 서버는 입력된 정보를 이용하여 데이터베이스와 연결한다. 웹 서버는 소스 데이터베이스에서 선택한 테이블과 같은 이름과 구조의 테이블을 생성하는 질의문을 완성하여 데이터베이스에게 전송한다. 질의문을 전송 받은 데이터베이스는 테이블을 생성한다.

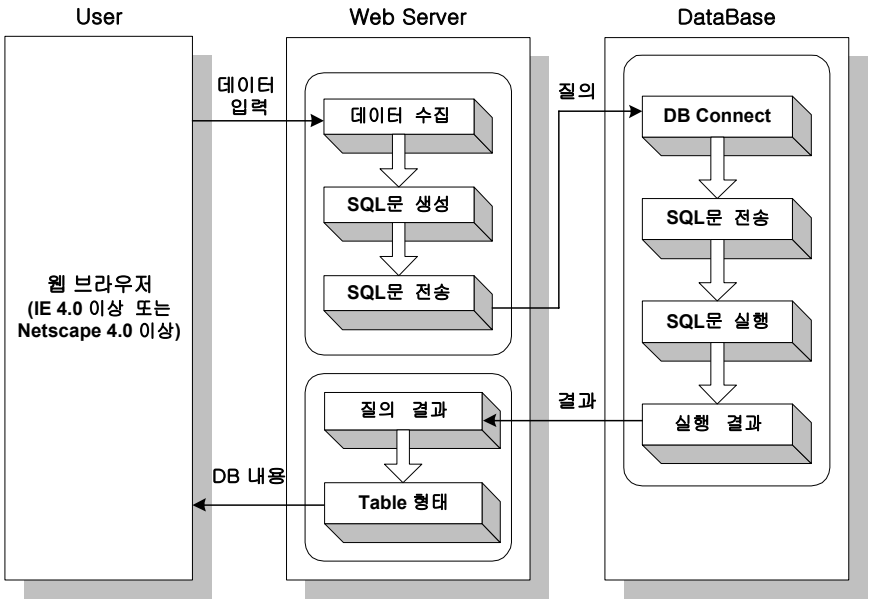
웹 서버가 복제하고자 하는 데이터의 내용을 가진 XML 문서에 대한 정보를 전달하면 목적지 데이터베이스는 생성한 테이블에 XML 문서의 내용을 저장한다. 이처럼 데이터베이스간의 데이터 복제는 XML 문서를 이용하여 이루어진다.

사용자는 데이터베이스에 대한 간단한 정보를 입력하면 웹을 통해 데이터베이스를 관리할 수 있다. 이는 기존의 데이터베이스를 관리하기 위해서 질의 언어나 데이터베이스에 관련된 전문적인 지식이 요구되어 초보자가 데이터베이스 관리가 불가능하던 문제점을 개선한 것이다.

본 논문에서 구현한 웹 에이전트는 인터넷에 연결되어 있는 컴퓨터에 사용자가 브라우저를 통해 웹 서버에 접속한다. 웹 서버에 접속하게 되면, 언제 어디서나 데이터베이스를 관리할 수 있고, 초보자라도 관리하기가 수월하다는 장점이 있다.

3.2 웹 에이전트의 데이터 교환 과정

본 논문에서 구현한 웹 에이전트 시스템에서 데이터베이스의 데이터 관리를 위한 데이터의 흐름은 <그림 3-2>와 같다.



<그림 3-2> 웹 에이전트의 데이터 흐름

<Fig. 3-2> Data flow of web agent

본 논문에서 구현한 웹 에이전트 시스템의 데이터 전달과정을 살펴보면 다음과 같다. 데이터베이스에 접속하여 작업하기를 원하는 사용자는 웹서버에 접속한다. 웹 서버에 접속한 사용자는 원하는 데이터베이스의 주소와 그 데이터베이스에 접속할 때 필요한 아이디와 패스워드를 입력하면 웹서버는 입력된 정보를 토대로 질의문을 생성한다.

웹 서버는 사용자가 입력한 정보를 이용하여 해당 데이터베이스에 접속한 뒤, 생성된 질의문을 실행한다. 웹 서버로부터 질의를 받은 데이터베이스는 질의에 대한 응답을 웹 서버로 보내고, 응답을 받은 웹 서버는 사용자가 브라우저를 통해 보기 편하게 데이터베이스에 있는 데이터의 내용을 테이블 형태로 보여준다.

웹 서버가 데이터베이스에 접속하게 되면 사용자는 브라우저를

통해 데이터베이스의 내용을 검색하여 데이터를 관리할 수 있다. 사용자가 데이터 관리를 하기 위해 필요한 추가 정보를 입력하면 웹 서버는 입력된 정보를 이용하여 데이터베이스에게 질의할 질의문을 완성한다. 완성된 질의문을 데이터베이스가 수행하게 되면 테이블 생성과 삭제 및 테이블 레코드의 내용을 수정할 수 있다.

본 논문에서 구현한 웹 에이전트 시스템은 단순히 웹에서 데이터베이스의 데이터 관리만 하는 것이 아니라, 테이블의 내용을 XML 문서로 저장하거나 다른 데이터베이스로 데이터를 복제할 수 있다.

사용자는 웹 서버를 통하여 저장된 XML 문서를 호출하고, 그 내용을 추가 및 삭제, 수정하여 데이터베이스에 저장하는데, 이는 실제적으로 데이터베이스의 내용을 수정하는 것이 아니라, 단순히 XML 문서의 내용을 수정하는 것이다.

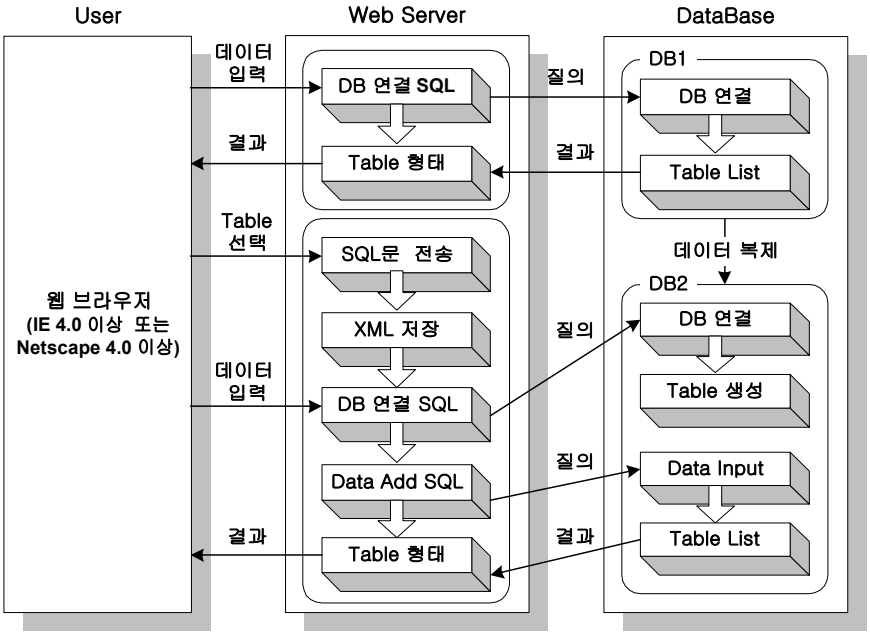
수정된 XML 문서의 내용을 원래 데이터베이스에 저장하여 데이터의 내용을 수정하기도 하지만, 다른 데이터베이스에 저장하여 새로운 형태의 테이블로 만들 수도 있다. 그리고, 임의의 데이터베이스에서 다른 데이터베이스로 테이블의 내용을 복제하는 기능을 가지고 있는 웹 에이전트 시스템은 데이터베이스간의 테이블 내용 복제를 위한 중간 매개물로 XML 문서를 이용한다.

XML을 이용하여 데이터베이스간의 내용 복제가 가능하게 되면 웹에서 사용되고 있는 서로 다른 데이터베이스간의 정보 교환이 가능하게 된다. 이는 본 논문에서 구현한 웹 에이전트 시스템이 웹에서 다른 데이터베이스와의 정보교환을 원하는 기업의 욕구를 충족시킬 수 있다는 것이다.

사용자는 단순히 웹 브라우저를 통해 연결하고자 하는 데이터베이스에 대한 정보를 입력하면 웹 서버는 입력된 정보를 토대로 연결하고자 하는 데이터베이스에 접속할 수 있는 질의문을 생성하여 데이터베이스에게 전송한다.

질의를 받은 데이터베이스는 질의문에 대한 질의 결과를 웹 서버에게 전송하고, 응답을 받은 웹 서버는 사용자가 보기 편하도록 테이블 형태로 보여지도록 만들어 사용자에게 보여준다.

웹 에이전트 시스템의 주된 기능은 XML 문서를 이용하여 데이터베이스에 저장된 데이터를 다른 데이터베이스로 복제하는 것이다. 이러한 과정을 수행하는 데이터 흐름은 <그림 3-3>과 같다.



<그림 3-3> 이기종 DBMS간의 복제

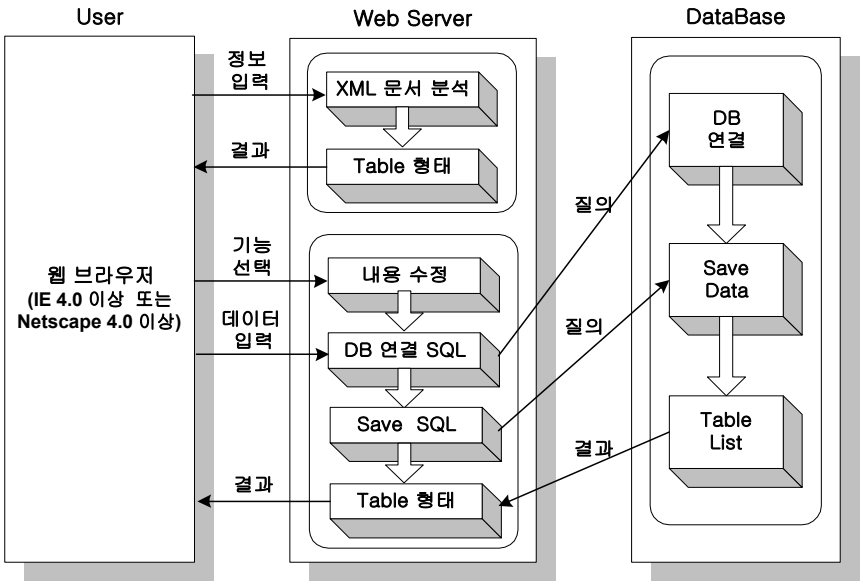
<Fig. 3-3> Data replication between heterogeneous DBMSs

<그림 3-2>와 비슷하게 사용자로부터 복제할 데이터를 갖고 있는 정보원 데이터베이스에 대한 정보를 입력받아 웹 서버는 정보원 데이터베이스와 연결하여 데이터베이스에 저장된 테이블의 목록을 사용자에게 보여준다.

사용자는 테이블 목록 중에서 복제하고 싶은 테이블을 선택하면, 웹 서버는 이 테이블의 내용을 XML 문서로 저장한다. 복제하고자 하는 테이블의 내용을 저장할 목적지 데이터베이스에 대한 정보를 사용자가 입력하면 웹 서버는 이를 토대로 목적지 데이터베이스에 테이블을 생성하여 데이터를 저장할 수 있는 질의문을 생성하여 데이터베이스에게 전송한다.

웹 서버로부터 질의문을 전송받은 목적지 데이터베이스는 질의문을 실행하여 복제하고자 하는 테이블의 이름과 같은 테이블을 생성하고 복제된 테이블의 이름을 포함한 테이블 리스트 목록을 웹 서버에게 전송한다. 목적지 데이터베이스로부터 응답을 받은 웹 서버는 이를 사용자가 보기 편한 테이블 형태로 만들어 보여준다.

끝으로, 웹에 저장된 XML 문서를 데이터베이스로 저장하는 과정은 <그림 3-4>와 같다.



<그림 3-4> XML 문서를 DB로 저장하는 과정

<Fig. 3-4> Save process from XML document to DB

사용자는 데이터베이스에 저장하고자 하는 XML 문서에 대한 정보를 입력하여 웹 서버로 전송한다. 웹 서버는 입력받은 정보를 분석하여 XML 문서의 내용을 테이블 형태로 사용자에게 보여준다.

제 4 장 웹 에이전트 구현 및 고찰

제 3 장에서 언급한 내용을 토대로 구현한 웹 에이전트 시스템은 사용자가 웹 서버를 통해 데이터베이스에 접근하여 데이터를 수정하거나 서로 다른 데이터베이스간의 데이터를 복제할 수 있도록 구현하였다.

구현된 웹 에이전트 시스템 환경은 윈도우 2000 서버이고, 데이터베이스는 오라클과 MS SQL 서버 7.0을 사용한다. 웹 서버는 윈도우에서 지원하는 인터넷 정보 서버(IIS ; Internet Information Server) 5.0을 사용하여 웹 서버를 구동하고, 동적인 웹 페이지를 생성하기 위해 사용하는 JSP 엔진으로 레신(resin)을 이용하고, JDK (Java Development Kit)를 설치하여 JDBC 드라이버와 연동이 되도록 설정하였다.

4.1 데이터베이스와 연결

제 3 장에서 언급한 것과 같이, 본 논문에서 구현한 웹 에이전트 시스템은 데이터베이스와 연결하기 위해서 Native protocol 순수 자바 드라이버를 사용한다. 이는 데이터베이스마다 다른 드라이버를 사용하기 때문에 데이터베이스의 종류에 따라 프로그램 과정에 필요한 드라이버의 이름이 다르다. 또한, 데이터베이스에 따라 입력하는 주소의 형태가 달라진다.

따라서, 구현된 웹 에이전트 시스템은 사용자가 웹 서버에 접속하여 데이터베이스의 종류를 선택하면 해당 드라이버의 이름이 저장된 웹 페이지로 이동하여 데이터베이스의 종류에 상관없이 데이터

를 관리할 수 있도록 하였다.

예를 들어, 데이터베이스의 종류를 오라클 데이터베이스로 설정하고, 연결하고자 하는 데이터베이스의 주소와 데이터베이스와 연결할 때 필요로 하는 사용자 아이디와 패스워드를 입력하는 화면은 <그림 4-1>과 같다.

<그림 4-1> 데이터베이스 로그온 웹 인터페이스

<Fig. 4-1> Web interface of database logon

이 때, 사용하는 아이디와 패스워드는 연결하고자 하는 데이터베이스에 대한 접근권한을 가지고 있어야 한다. <그림 4-1>에서 보면 연결하고자 하는 데이터베이스의 주소는 203.230.253.8이고, 연결 포트번호는 1521이고, 데이터베이스 이름은 nice이다.

<그림 4-1>과 같이 데이터베이스에 대한 정보를 사용자가 입력하여 웹 서버로 전송하면, 웹 서버는 입력된 정보를 이용하여 데이

터베이스에 연결할 수 있는 질의문을 완성한다. 이러한 과정의 프로그램 소스는 <그림 4-2>와 같다.

```
String url1 = "jdbc:oracle:thin:@";
String url2 = null;
String user = null;
String pw = null;
String chk = "chk";
String list = "select * from tab";

url2 = request.getParameter("txtadd");
user= request.getParameter("txtid");
pw = request.getParameter("txtpw");

Connection conn=null;
Statement stmt=null;
ResultSet rs=null;

try{
    Class.forName("oracle.jdbc.driver.OracleDriver");
    conn=java.sql.DriverManager.getConnection(url1+url2, user, pw);
    stmt=conn.createStatement();
    rs=stmt.executeQuery(list);
}catch(java.sql.SQLException e){
     out.println(e);
}
```

<그림 4-2> 로그온 프로그램 소스

<Fig. 4-2> Logon program source

<그림 4-2>에서 나오는 "url"이라는 변수는 접속하고자 하는 데이터베이스 종류에 따라 주소영역에 붙은 접두어이다.

<그림 4-1>에서 데이터베이스의 종류를 오라클로 선택하였으므로 주소영역에 붙는 접두어는 "jdbc:oracle:thin:@"가 된다. 또한, "list"라는 변수는 데이터베이스에 저장되어 있는 테이블의 목록을 가져오기 위한 질의문을 나타낸다.

<그림 4-1>에서 사용자가 로그온 버튼을 선택하면 텍스트 박스에서 입력된 정보를 이용하여 웹 서버는 <그림 4-2>의 try문 내용을 수행하여 웹 서버와 데이터베이스를 연결하고, 데이터베이스에 저장된 테이블 목록을 가져오는 질의문인 변수 “list”를 수행한 결과를 웹 서버에게 전송한다.

드라이버의 이름은 데이터베이스의 종류에 따라 다르다. 오라클 데이터베이스인 경우는 “oracle.jdbc.driver.OracleDriver”이고, SQL 서버를 사용할 경우는 “com.inet.tds.TdsDriver”이다. <그림 4-2>에서 보면 try 구문 안에 들어있는 Class.forName 부분을 이용하여 원하는 드라이버를 로딩할 수 있도록 하였다.

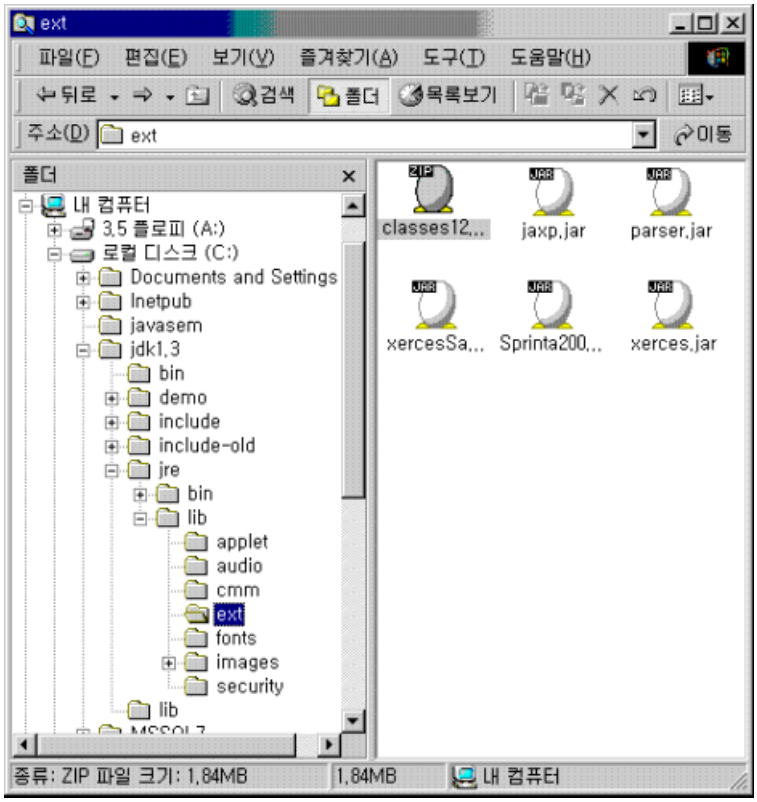
데이터베이스 회사별로 제공하고 있는 드라이버를 사용하기 위해서는 각 데이터베이스 회사로부터 파일을 다운 받아 JDK가 설치된 디렉토리 밑에 저장하고 배치 파일을 수정하여야 한다.

본 논문에서 구현한 웹 에이전트 시스템에서는 경로를 설정하기 위해서 배치 파일을 <그림 4-3>과 같이 설정하고, 드라이버를 사용하기 위한 파일은 <그림 4-4>와 같이 “c:\jdk1.3\jre\lib\ext”에 저장하였다.

```
set path=c:\jdk1.3\bin;
set classpath
=c:\orant\jdbc\lib\classes12.zip;c:\jdk1.3\jre\lib\ext\Sprinta2000.jar;
 c:\jdk1.3\jre\lib\ext\classes12.zip;c:\jdk1.3\jre\lib\ext\parser.jar;
 c:\jdk1.3\jre\lib\ext\jaxp.jar;c:\jdk1.3\jre\lib\ext\xerces.jar;
 c:\jdk1.3\jre\lib\ext\xercesSamples.jar;
```

<그림 4-3> 자바와 JDBC 드라이버의 경로 설정

<Fig. 4-3> Path configuration of java and JDBC driver



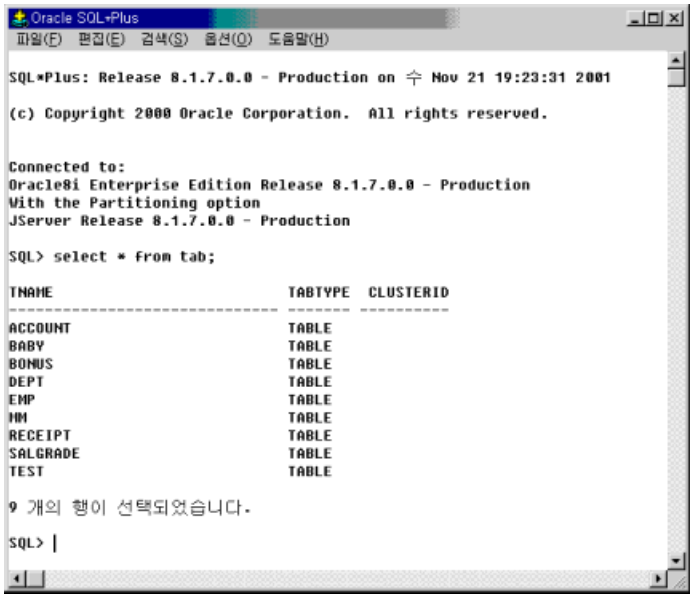
<그림 4-4> JDBC 드라이버 설정 화면

<Fig. 4-4> Setting window of JDBC driver

데이터베이스와 연결이 되면 웹 서버는 데이터베이스로부터 테이블 목록에 대한 정보를 얻게 된다. 웹 서버는 데이터베이스에 저장되어 있는 테이블 목록에 대한 정보를 사용자가 웹 브라우저를 통해 보기 편한 테이블 형태로 표현한다.

실제 오라클 데이터베이스와 연결하여 테이블 목록을 알아본 결과는 <그림 4-5>와 같고, 웹 서버가 보여주는 화면은 <그림 4-6>과 같다.

<그림 4-5>와 <그림 4-6>에서 보는 것처럼 테이블 목록의 내용이 같다는 것을 확인할 수 있다.



<그림 4-5> 오라클 데이터베이스의 테이블 목록

<Fig. 4-5> Table list of oracle database

<그림 4-6> DB 연결 후 결과 화면

<Fig. 4-6> Result window after DB connection

4.2 데이터베이스의 관리

<그림 4-6>에서 보는 것처럼 사용자는 테이블의 생성과 삭제 및 수정하는 데이터 관리 기능뿐만 아니라, 사용자가 선택한 특정 테이블을 다른 데이터베이스로 복제하는 기능도 수행할 수 있다.

<그림 4-6>에서 사용자가 원하는 작업에 따라 추가적으로 정보를 입력하면, 웹 서버는 입력된 정보를 이용하여 새로운 질의문을 만들어 데이터베이스로 전송한다. 데이터베이스는 질의문을 수행하여 데이터를 처리한 다음, 그 결과를 웹 서버에게 알려준다.

사용자는 데이터 관리를 위한 전문적인 지식이 없더라도 추가 정보를 입력하면 웹 서버에서 해당 질의문을 생성하여 사용자가 원하는 작업을 할 수 있도록 하고 있다.

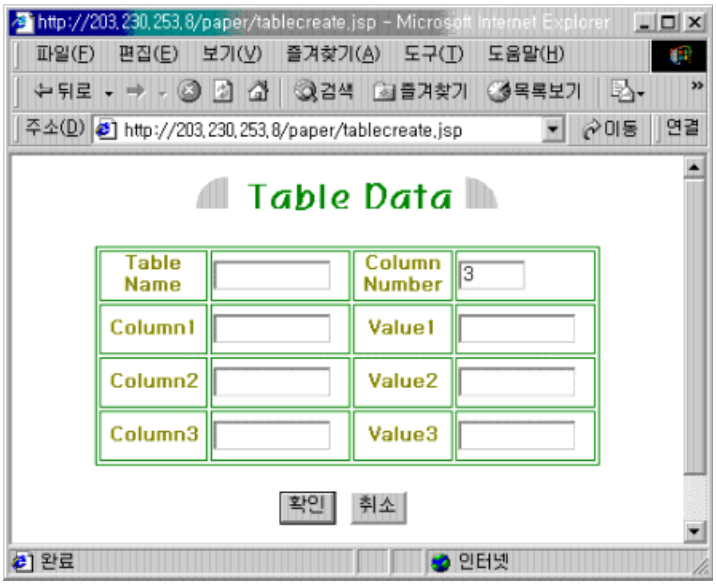
4.2.1 Create 버튼

Create 버튼은 테이블을 생성하는 버튼으로서 생성하고자 하는 테이블 이름, 컬럼 이름과 같은 추가 정보가 필요하다. 우선, 사용자는 생성하고자 하는 테이블의 컬럼 수를 입력하면 웹 서버는 입력된 정보를 이용하여 다음 <그림 4-7>과 같이 컬럼 수에 해당하는 입력 가능한 텍스트 박스를 만들어 사용자에게 보여준다.

사용자는 생성하고자 하는 테이블의 이름과 컬럼 이름 및 데이터 타입을 입력하고 웹 서버는 입력된 정보를 이용하여 새로운 테이블을 생성한다.

웹 서버는 웹 브라우저를 통해 입력된 추가 정보를 이용하여 테이블을 생성하는 질의문을 완성한다. 완성된 질의문을 웹 서버는 데이터베이스에게 전송하여 테이블을 생성하고, 그 결과를 웹 서버에

게 되돌려준다. 웹 서버는 추가된 테이블 이름을 포함한 테이블 목록 화면을 웹 브라우저를 통해 사용자에게 보여준다.



<그림 4-7> 테이블 생성 화면

<Fig. 4-7> Table create window

테이블을 생성하기 위한 질의문을 만드는 과정의 프로그램 소스는 <그림 4-8>과 같다.

프로그램 소스에서 for문은 <그림 4-7>에서 사용자가 입력한 테이블 이름을 “name”이라는 변수에 저장하고, 컬럼 이름과 데이터 타입을 조합하여 “sum”이라는 변수에 저장한다. 테이블을 생성하기 위해 “createqy”라는 변수에 “name”과 “sum”을 조합하여 질의문을 완성하는 과정을 나타내고 있다.

완성된 질의문을 데이터베이스에서 실행하도록 하는 프로그램 소스는 <그림 4-2>에서 보여준 try문, catch문과 유사하다.

```
    for(j=1; j<= n; j++){
        String cj = c + j;
        String vj = v + j;
        col = request.getParameter(cj);
        val = request.getParameter(vj);
        total[j] = col + " " + val;
    }
    for(j=1; j<= n; j++){
         sum += total[j];
           if( j < n){
               sum = sum + "," + " ";}
           else{
               sum = sum;
           }
    }

String name = null;
name = request.getParameter("tbnm");
String createqy = "create table " + name + "(" + sum + ")";
```

<그림 4-8> 테이블 생성 프로그램 소스

<Fig. 4-8> Table create program source

4.2.2 Delete 버튼

테이블을 삭제하는 Delete 버튼은 추가 정보를 입력하지 않고 사용자가 선택한 테이블을 데이터베이스에서 삭제하는 기능을 수행한다. 테이블을 삭제하기 위한 질의문을 완성하여 테이블을 삭제하고, 웹 서버는 테이블 목록에서 삭제된 테이블의 이름이 빠진 테이블 목록을 사용자에게 보여준다.

테이블을 삭제하는 프로그램 소스는 <그림 4-9>와 같다.

<그림 4-9>에서 for 구문은 사용자가 선택한 테이블의 이름을 알아내는 내용이고, "del"이라는 변수는 선택한 테이블을 삭제하기 위한 질의문이다.

```
int checklen = 0;
checklen = checks.length;
for(int i=0;i<checklen;i++){
    num = checks[i];
    int j = Integer.parseInt(num);
    table = nm[j-1];

}
Connection conn2=null;
Statement stmt2=null;
String del =“drop table ″ + table;

try{
    Class.forName("oracle.jdbc.driver.OracleDriver");
    conn2=java.sql.DriverManager.getConnection(url1+url2, user, pw);
    stmt2=conn2.createStatement();
    stmt2.executeUpdate(del);
}
catch(java.sql.SQLException e){
    out.println(e);
}
     conn2.close();
```

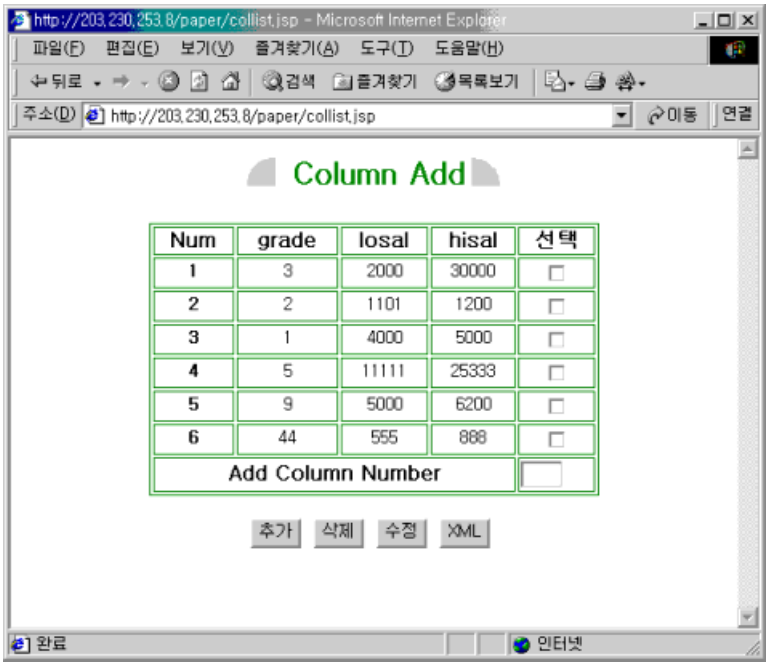
<그림 4-9> 테이블 삭제 프로그램 소스

<Fig. 4-9> Table delete program source

4.2.3 Update 버튼

Update 버튼을 선택할 경우, 웹 서버는 데이터베이스에게 사용자가 선택한 테이블의 내용을 알아내기 위한 질의문을 전송한다. 데이터베이스가 이에 대한 응답으로 테이블에 저장되어 있는 컬럼과 레코드의 내용을 웹 서버에게 전달하면, 웹 서버는 이를 테이블의 형태로 바꾸어 사용자에게 보여준다.

예를 들어, SALGRADE라는 테이블을 체크하여 Update 버튼을 선택할 경우, 웹 서버는 <그림 4-10>과 같이 테이블에 저장되어 있는 데이터를 보여준다.



<그림 4-10> 테이블 수정을 위한 웹 인터페이스

<Fig. 4-10> Web interface for table update

<그림 4-10>에서 추가, 삭제, 수정 버튼의 기능은 <그림 4-6>에 있는 버튼들과 같은 비슷한 기능을 수행하지만, 관리하는 대상이 테이블 자체가 아닌 테이블에 저장되어 있는 컬럼과 레코드의 내용이라는 점에서 차이가 있다.

<그림 4-6>과 달리, <그림 4-10>에는 XML 버튼이 있는데, 이는 테이블의 내용을 XML 문서로 저장하는 기능을 수행한다.

XML 문서의 루트 태그는 테이블 이름으로 하고, 루트 태그의 자식 엘리먼트로 <record> 태그를 두고 데이터베이스에 저장된 컬럼의 이름은 XML 문서에서 <record>의 자식 엘리먼트인 태그 이름으로 매핑시킨다.

본 논문에서는 루트 디렉토리에 “테이블이름.xml”이라는 이름으로 저장되도록 JSP 프로그램을 만들었다. 예를 들어, SALGRADE라는 테이블의 내용을 XML 문서로 저장하는 프로그램은 <그림 4-11>과 같고, 브라우저로 확인해 보면 <그림 4-12>와 같다.

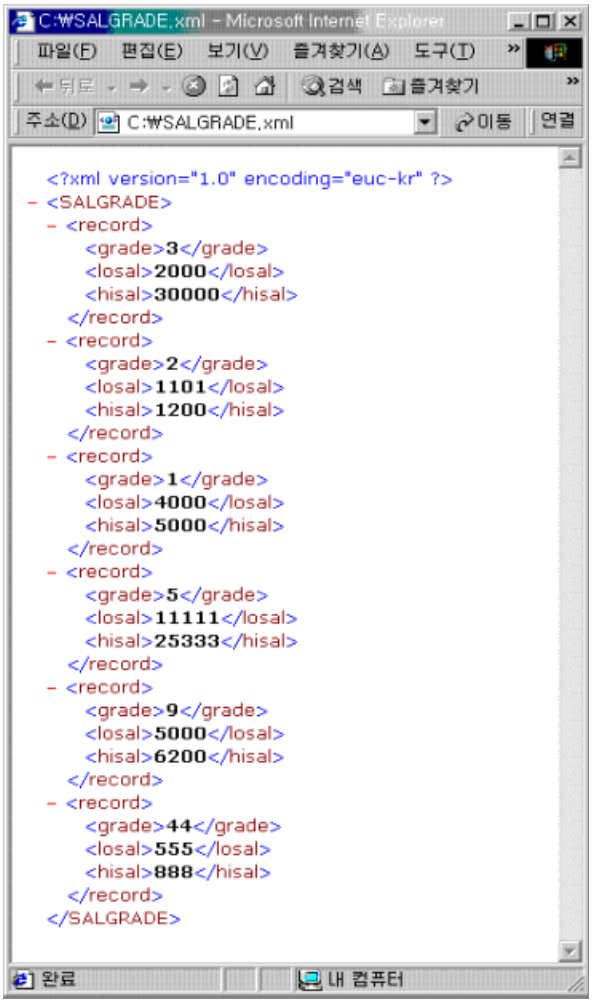
```
try{
  PrintWriter pw
      = new PrintWriter(new FileOutputStream(nameOfTextfile));
 pw.println(title);
 pw.println(root);...
   for(int j=1; j<=num; j++){
      String colnmk = "colnm" + j;
      String colvalue = request.getParameter(colnmk);
      colname[j] = colvalue;
   }
   for(int i=1; i<=n; i++){
      String starttag = "<" + record + ">";
      pw.println(starttag);
           for(int k=1; k<=num; k++){
             String itemik =  "item" + i + k;
             String item = request.getParameter(itemik);
             String tag = "<" + colname[k] + ">";
                                .....
             String endtag = "</" + colname[k] + ">";
               pw.println(endtag);
           }
        String lasttag = "</" + record + ">";
        pw.println(lasttag);
    }
```

<그림 4-11> XML 저장 프로그램 소스

<Fig. 4-11> XML save program source

<그림 4-11>에서 “pw”라는 변수는 <그림 4-10>에서 보여주는 테이블의 내용을 텍스트 파일로 저장하고 변수 “title”은 XML 문서의 선언 부분인 “<?xml version='1.0' encoding='euc-kr'?>”을 표시

하고 변수 “root”는 테이블의 이름을 나타내는 것이다. for문의 내용은 <그림 4-10>의 컬럼인 grad, losal, hisal을 <record>라는 엘리먼트의 자식 엘리먼트로 두어 테이블의 내용을 저장하도록 하였다.



<그림 4-12> SALGRADE.xml 문서

<Fig. 4-12> SALGRADE.xml document

4.3 데이터베이스 복제

데이터베이스간의 데이터 복제를 위해 사용되는 DBCopy 버튼은 목적지 데이터베이스에 대한 주소와 아이디 및 패스워드를 사용자가 입력하는 화면이 나타나게 된다. 사용자가 정보를 입력하면 웹 서버는 전송 받은 정보를 토대로 테이블을 생성하여 복제하고자 하는 데이터 내용을 추가한다.

DBCopy 버튼을 선택하면 데이터베이스에 연결하는 <그림 4-1>과 같은 화면이 나타나지만 차이가 있다. 이것은 복제할 데이터가 있는 데이터베이스가 아니라 복제해서 저장할 목적지 데이터베이스에 대한 정보를 입력한다는 것이다.

웹 서버는 이 정보를 토대로 데이터베이스에 연결하는 질의문을 생성하여 데이터베이스와 연결하여 복제할 테이블과 같은 이름의 테이블을 생성한다.

우선, 복제할 데이터가 있는 데이터베이스의 테이블 목록을 보여주는 화면에서 하나의 테이블을 체크하여 DBCopy라는 버튼을 선택한다. 웹 서버는 복제하고자 하는 테이블의 이름, 테이블의 컬럼 이름, 데이터 타입에 대한 정보가 필요하게 된다. 테이블 목록에서 사용자가 복제하고자 하는 테이블을 체크하면 웹 서버는 테이블의 이름을 쉽게 알아낼 수 있다.

웹 서버는 테이블의 이름을 이용하여 테이블의 컬럼 이름과 컬럼의 데이터 타입을 알아내는 질의문을 이용한다. 데이터베이스에 저장되어 있는 테이블의 정보를 기억하고 있는 user_tab_column이라는 오브젝트를 이용한다.

사용자가 선택한 테이블에 대한 정보를 알아내는 JSP 프로그램 소스는 다음 <그림 4-13>과 같다.

```
String colup
 = “select column_name, data_type from user_tab_columns where
     table_name = SALGRADE”
try{
     Class.forName("oracle.jdbc.driver.OracleDriver");
     conn6=java.sql.DriverManager.getConnection(url, user, pw);
     stmt6=conn6.createStatement();
     stmt6.executeUpdate(colup);
}
catch(java.sql.SQLException e){
       out.println(e);
}
```

<그림 4-13> 테이블 정보 질의 프로그램 소스

<Fig. 4-13> Table information query program source

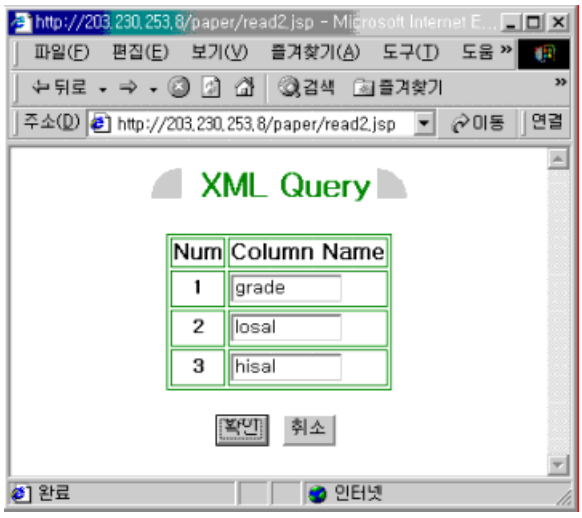
웹에 존재하고 있는 XML 문서 중에서 사용자가 데이터베이스가 저장하고자 하는 특정 XML 문서를 저장하는 과정을 살펴보면 다음과 같다. 우선, 사용자는 특정 XML 문서에 대한 위치 정보와 데이터베이스에 저장할 경우 테이블의 이름이 될 루트 태그 이름 등의 정보를 사용자가 입력하는 화면은 <그림 4-14>와 같다.

웹 서버는 사용자가 데이터베이스에 저장하고자 하는 엘리먼트의 이름을 입력할 수 있는 <그림 4-15>와 같은 화면을 통해서 XML 문서를 데이터베이스에 저장할 경우 컬럼이 되는 엘리먼트의 이름에 대한 정보를 얻는다.

이 같은 수작업은 데이터베이스에 저장하고자 하는 엘리먼트의 수가 많을수록 번거로운 작업이 되지만, 필요한 정보만 입력하여 XML 문서 전체의 저장이 아니라 필요한 부분만 저장할 수 있다는 장점을 가지게 된다.

<그림 4-14> XML 문서를 읽기 위한 화면

<Fig. 4-14> Window for XML document reading



<그림 4-15> 태그 이름 입력 화면

<Fig. 4-15> Tag name input window

<그림 4-15>와 같이 사용자가 보고자 하는 XML 문서의 태그 정보를 입력하면 웹 서버는 XML 문서의 내용을 분석하여 원하는 정보만 사용자가 보기 편한 <그림 4-16>과 같은 화면을 보여준다. 이는 XML 문서의 내용을 데이터베이스에 저장할 모양과 테이블과 유사한 형태이다.

<그림 4-16> 요청한 XML 문서 결과 화면

<Fig. 4-16> Result window of request XML document

데이터베이스의 데이터 관리와 유사하게 XML 문서의 내용을 수정 버튼을 이용하여 추가와 삭제, 수정할 수 있다. 그리고, 수정 과정을 거치지 않고 XML 문서의 내용을 원하는 데이터베이스에 저장할 수 있다. 이는 웹에 존재하는 XML 형식의 웹 페이지를 데이터베이스에 저장하여 관리함으로써 정보의 재사용과 정보 공유를 더욱 원활하게 한다.

지금까지, 본 논문에서 구현한 웹 에이전트 시스템에 대하여 설명하였다. 이미 나와 있는 제품과 달리 별도의 소프트웨어 없이 서로 다른 데이터베이스간의 데이터 복제를 할 수 있다는 것이 이 웹 에이전트의 주된 장점이다.

그리고, 웹에서 정보검색 능력만 가진 초보자라 할지라도 데이터베이스와 정보교환을 할 수 있을 만큼 쉽게 구성되어 있다.

제 5 장 결 론

오늘날, 웹이 지식과 정보 교류의 기반이 되면서 기업들은 정보 공유와 웹을 이용한 경제활동에 대한 관심이 증가하였다. 기업들은 데이터베이스에 저장된 상업적인 정보를 공유하고 관리하기 위해 문서에 대한 구조 정보를 내포하고 있는 XML을 정보 저장소로 이용한다. XML은 문서에 대한 구조정보를 제공하고, 데이터를 해석할 수 있다. 이러한 XML을 정보 저장소로 이용한다면 웹을 이용한 정보 공유와 정보 검색 및 정보 관리까지 손쉽게 할 수 있다.

본 논문에서는 웹에서 이루어지는 이기종 DBMS간의 데이터 복제와 XML 문서와 데이터베이스간의 정보 교환에 대하여 연구하고 XML을 이용하여 이기종 DBMS간의 정보를 교환할 수 있는 웹 에이전트 시스템을 설계하고 구현하였다. 웹 에이전트 시스템의 구성과 데이터 흐름은 제 3 장과 제 4 장에서 언급하였다. 실제로 구현된 웹 에이전트 시스템의 주된 기능은 이기종 DBMS간의 데이터 복제와 XML 문서를 데이터베이스에 저장하는 것이다.

제 2 장에서 소개된 펜타 시스템의 타미노와 오라클 제품은 본 논문에서 구현한 웹 에이전트 시스템과 유사한 기능을 제공하지만 웹 에이전트 시스템은 기존의 제품이 갖는 한계를 극복하였다.

XML 문서를 XML 포맷으로 저장하고 관리하는 타미노는 클라이언트와 서버간의 연결을 위하여 별도의 소프트웨어와 라이센스를 구매해야 하고, 이를 통하여 데이터를 교환해야 하지만 웹 에이전트 시스템은 추가 소프트웨어 없이 보편적인 웹 브라우저를 사용한다는 것이 타미노와 다르다.

데이터베이스간 데이터를 교환할 수 있는 오라클 제품은 오라클에서 제작한 XSU를 가진 오라클 데이터베이스와 다른 데이터베이

스간의 정보 교환만 가능하지만 웹 에이전트 시스템은 웹 서버가 가지고 있는 JDBC의 드라이버에 의해 연결된 데이터베이스간의 정보 교환이 가능하다. 즉, 웹 서버가 JDBC를 이용하여 연결할 수 있는 모든 데이터베이스간의 정보 교환이 가능하다.

웹 에이전트 시스템의 주된 기능인 이기종 데이터베이스간의 데이터 복제를 위해 데이터베이스에 저장된 데이터를 XML 문서로 변환한다. 사용자의 입장에서는 복제하고자 하는 데이터베이스의 데이터를 직접 목적지 데이터베이스로 데이터를 보내는 것처럼 보이지만 실제는 복제할 데이터의 내용을 XML 문서로 변환하고 변환된 XML의 내용을 목적지 데이터베이스에 저장하는 것이다. 이와 같이 웹 에이전트 시스템은 이기종 DBMS간의 데이터 복제를 위해 XML 문서의 내용을 데이터베이스에 저장한다.

그리고, 웹 에이전트 시스템은 사용자에게 웹 브라우저를 통해 데이터베이스의 데이터와 XML 문서의 내용을 보여준다. 사용자는 데이터베이스를 관리하기 위한 질의문에 대한 사전지식 없이 단순히 원하는 기능에 해당하는 버튼을 선택하고 필요한 추가 정보를 입력하면 원하는 작업을 수행할 수 있다. 즉, 사용자가 웹 서버에 접속한다면, 언제 어디서든지 데이터베이스 관리가 이루어진다.

향후 연구 방향으로는 웹에서 이루어지는 정보교환 과정에 보안을 고려하고, XML 질의언어인 XQL을 이용하여 XML 문서에 대한 효율적인 정보검색 기능을 가진 웹 에이전트 시스템에 대하여 연구할 계획이다.

참 고 문 헌

[1] 이호섭, 홍충선, “분산환경에서의 CORBA와 XML의 연동구조”, 한국정보과학회 봄학술발표논문집, Vol. 28, No. 1, pp. 424-426, 2001

[2] 이강찬, 손홍, 박기식, “XML 표준화 동향”, 정보과학회지, 제 19 권, 제 1 호, pp. 6-14, 2001

[3] W3C HTML, http://www.w3.org/MarkUp/#historical

[4] A Beginner's Guide to HTML, http://archive.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimerAll.html

[5] W3C Extensible Markup Language(XML) 1.0 (second edition), http://www.w3.org/TR/2000/REC-xml-20001006

[6] W3C XML 1.1, http://www.w3.org/TR/xml11/

[7] Martin Bryan, “An Introduction to the Extensible Markup Language”, http://www.personal.u-net.com/~sgml/xmlintro.htm

[8] 박은경, 정채영, 김현주, 배종민, “XML DTD로부터 관계형 테이블로의 사상구조 설계”, 한국정보과학회 학술논문집, Vol. 28, No. 1, pp. 133-135, 2001

[9] 윤정희, 박창원, 정진완, “객체 식별자를 이용한 객체지향 데이터베이스의 XML 문서로의 변환”, 정보과학회논문지 : 데이터베이스, 제 28 권, 제 2 호, pp. 131-139, 2001

[10] Frakes, Baeza-yates, *Information Retrieval : Data Structure and Algorithms*, Prentice-Hall, 1992

[11] 강형일, 최영길, 이종설, 유재수, 조기형, “RDBMS와 IRS를 이용한 XML 저장관리 시스템 설계 및 구현”, 정보과학회논문지 : 컴퓨팅의 실제, 제 7 권, 제 1 호, pp. 1-10, 2001. 2

[12] 장우혁, 김홍식, “XML기반의 효율적인 데이터 저장관리를 위한 DB2XML 변환 Wrapper의 설계”, 한국정보과학회 학술발표논문집, Vol. 28, No. 1, pp. 106-108, 2001

[13] 박철현외 정재헌, 심대익, 이상구, “구조화된 문서에 대한 DBMS와 IRS의 성능 비교”, ‘99 한국 데이터베이스 학술대회 논문집, 15권, 1호, pp. 218-225, 1999

[14] 손정한, 이희주, 장재우, 심부성, 주종철, “SGML 정보 검색을 위한 인덱스 관리자의 설계 및 구현”, 정보과학회논문지(C), Vol. 5, No. 2, pp. 135-146, 1999

[15] 유재수외 8명, “전자도서관 표준문서 관리를 위한 XML 저장관리기 기술 개발”, 한국지식웨어 최종보고서, 1999

[16] “오라클 iAS와 DB에서의 XML 프로그래밍”, http://xmlgo.net/document/db/iAS_oracle_xml.htm

[17] “Transaction Architecture for Management of INternet Object”, http://www.ebm.co.kr/product/tamino/intro_tamino.htm

[18] Panta 회사 홈페이지, http://www.tamino.co.kr/information/sagtamintro.htm

Abstract

XML(eXtensible Markup Language) has recently emerged as a new standard for data representation and exchange on the Internet. Web servers and applications encoding their data in XML can quickly make their information available in a simple and usable format, and such information providers can inter-operate easily. Information content is separated from information rendering, making it easy to provide multiple views of the same data. XML data files can be rendered via specifications in XSL (eXtensible Stylesheet Language). Most of us are familiar with web sites that contain vast databases of useful information, but whose query and search facilities are surprisingly primitive.

This paper proposes web agent for data replication between heterogeneous DBMSs(Database Management System). Web agent system manages database on the web and exchanges data in heterogeneous database using XML.

Using web page, user sends to web server information about database, web server creates query sentence for database connection. Using created query sentence, web server exchanges data with database server.

Web server makes XML document from source database data, XML document saves destination database. Using this way, web agent system exchange data between databases. So, this web agent system manage database data on the web, exchange data between heterogeneous DBMSs.