데이터베이스와의 연동 기반 XML 에디터의 설계 및 구현

고윤희*, 김현철*, 이원규*
*고려대학교 컴퓨터교육과
e-mail:unygo@comedu.korea.ac.kr

Design and Implementation of a XML Editor For DataBases

Youn-Hee Ko*, Hyeon-Cheol Kim*, WonGyu Lee*
*Dept of Computer Science Education, Korea University

요약

전자상거래를 위한 XML 및 관련 표준은 기존의 표준을 개선하거나 새로운 표준이 워킹 드래프트로 제안되고 있는 등 계속적으로 발전되고 있다. XML이 전자상거래에서 표준화된 문서로써 사용됨에 따라 XML문서의 효율적인 작성과 유지관리를 위해 다양한 XML에디터들이 개발되고 있다. 하지만 기존 에디터들의 경우, XML문서의 유효성만 체크하면서 단순한 편집기능만을 제공해 주고 있다. 따라서 이 논문에서는 실제 XML문서를 파싱하여 데이터베이스에 저장하고 자신이 원하는 쿼리(query)를 보냄으로써 그 결과를 XML문서로 변환하여 보여주는 기능을 구현하였다. 이는 전자 상거래 뿐만 아니라 동종 업계만의 전자 문서 교환에 있어 주고 받는 XML문서를 특별히 따로 처리할 필요 없이 원한다면 바로 자신의 데이터베이스로 파싱하여 넣거나, 필요한 부분을 기존의 데이터베이스에서 쿼리하여 이를 XML문서로 변환하여 주고 받는데 용이하게 이용될 수 있다.

1. 서론

XML문서는 엘리먼트(element)간의 논리적인 계층 구조를 가지고 있는 DTD(Document Type Definition)와 DTD에 선언된 엘리먼트에 따라 기술된 XML문서로 구성된다. 그러나 사용자가 직접 작성한 XML문서는 오류를 범할 가능성이 있으므로, 현재 많은 XML문서의 편집 도구가 개발되고 있다. 이들은 보통 XML 파서(parser)를 사용하여 XML문서의 유효성 검사를 한다. 이러한 어플리케이션들은 사용자가 특별한 지식이 없이도 웹 문서를 쉽게 생성할 수 있도록 해야 하며, 접근하기 쉬운 사용자 인터페이스를 제공하여야 한다. 또한, XML문서가 전자 상거래에서 문서 교환을 위해 사용된다는 점과 XML문서 자체의 내용이 데이터베이스와 불가분의 관계라는 것을 고려할 때, 이런 XML문서와 데이터베이스의 상호 연계에 대한 부분도 고려해야 한다. 따라서 기존에 존재하는 XML문서를 데이터베이스로 저장해 준다거나 데이터베이스에 있는 내용을 XML문서로 자동으로 만들어 줄 수 있는 기능까지 에디터에서 제공해 줄 수 있다면 보다 편리하게 XML문서를 처리해 줄 수 있을 것이다.

본 논문에서는 데이터베이스로의 저장기능을 갖는 XML 에디터를 제안하고 이를 구현한다. 제안한 XML 에디터는 기존의 에디터가 갖는 XML문서 편집 및 DTD정의 기능 뿐만 아니라, XML문서를 파싱해 데이터베이스에 저장하거나, 쿼리를 통해 그 결과를 XML문서로 만들어 줄 수 있는 기능을 제공해 준다.

2. 관련연구

2.1 XML 에디터

에디터는 크게 XML문서에 대한 내부 모델링 작업과 화면에 디스플레이 하는 엔진의 두 부분으로 나눌 수 있다. XML의 기본 구조는 원칙적으로 문서의 내용과 구조를 기술하는 파일과 문서의 외양적 스타일을 기술하는 파일로 나누어지도록 되어 있어 파일의 내용은 응용 프로그램이나 플랫폼에 의존적이지 않고 호환되도록 하는 것이다. 해당 문서 DTD 규칙에 맞는 XML문서의 생성을 위해서는 문서 구조의 틀을 에디터가 미리 알고 있어야 하며 이것이 유효하게 생성되면 스타일 파일 정보를 적용하여 일반적인 에디터에서의 랜더링(rendering) 과정 즉 물리적 문서 포맷팅 과정이 진행된다.

2.2 XML문서의 저장 방법

XML 문서를 데이터베이스에 저장하기 위한 방법은 분할 저장구조와 비분할 저장구조로 분류할 수 있다. 분할 저장구조는 DTD의 구조 정보를 해석하여 각 엘리먼트 단위의 인스턴스 트리 형태의 완전히 구조화된 데이터 베이스 내부 표현을 갖는 구조이다. 반면에 비분할 저장구조는 문서 자체를 하나의 연속된 스트링으로 취급하여 BLOB 형태로 저장하여 특정 엘리먼트에 대한 접근은 옵셋(offset)을 활용하는 구조이다.

분할 저장 접근 방법은 문서의 수정이 용이하나, 전체 문서의 삽입 및 검색 등 기본적인 연산 비용이 높다. 비분할 저장 접근 방법은 저장 비용 및 기본적인 연산에 있어서 향상된 성능을 달성할 수 있지만, 갱신 등 완전히 구조화된 표현이 적절한 문제에 있어서는 결점을 지니게 된다.

따라서, 본 논문에서는 문서에 대한 기본적 수정이 용이한 분할 저장 접근 방법을 선택해 XML문서를 데이터베이스에 저장시, 이를 파싱하여 그 데이터 내용만을 저장하는 방법을 택하였다.

3. 시스템 구성도

제안한 XML 에디터는 크게 XML문서를 보여주는 모듈, 이를 편집하는 모듈, 그리고 XML문서를 데이터베이스와 연동하기 위한 연동 모듈로 구성된다. 이의 구현은 java를 사용하였고 사용자 인터페이스는 swing를 사용하여 구현하였다. XML문서의 유효성 검사를 위한 파서로는 xerces를 사용하였고, XML 문서를 데이터베이스와 연결하여 이를 저장하고 불러오기 위해 오라클의 XSU(XML SQL Utility for java)를 사용하였다.

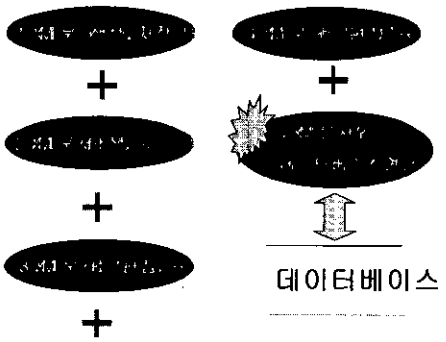

그림 1 XML 에디터의 구성도

XML 문서의 구조가 잘 정의되어지고 자주 변경이 가해지는 경우는 XML 문서의 전체 또는 일부를 오라클의 객체-관계형 테이블 형태로 저장하는 것이 효율적이다. XSU는 XML 문서의 전체 또는 일부를 객체-관계형 테이블로 사상하여 저장하는 기능과 객체-관계형 테이블의 데이터를 검색하는 기능을 제공한다. 또한, XML 데이타가 Oracle8i에 저장될 때 XML 문서의 각 엘리먼트의 이름들은 Oracle8i의 테이블에서의 컬럼명으로 사상된다. 이는 원한다면 XSL를 이용하며 다른 XML문서로 변환할 수 있다. XML에디터에서 XML문서를 데이터베이스와 연동하기 위한 내부 과정은 그림 2 및 그림 3 과 같다.

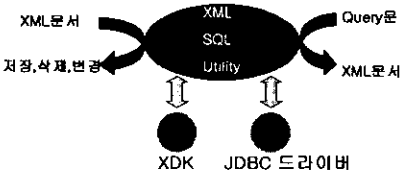

그림 2 XSU의 구성 요소

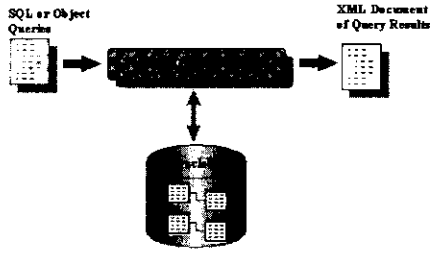

그림 3 XSU를 이용한 XML문서의 처리도

4. XML에디터의 구현

4.1 XML문서 조작 인터페이스

Open 프로세스는 메뉴바를 선택시 파일 다이얼로그 창을 보여주고 선택된 파일을 읽어, 소스창에 보여줌과 동시에 이 파일을 getRoot()함수에 넘겨준다. getRoot()함수는 파일을 읽어 null이 아닌 경우, 이를 InputSource로 바꾼 뒤, xerces의 DOMParser를 이용하여 parsing한 후, DomTree 객체로 넘겨 준다. 여기서는 이 객체의 자식노드를 얻어 노드리스트(NodeList)를 만든 후, 각 노드들의 타입을 체크하면서, 각 노드가 삽입될 노드를 결정하여 트리구조로 보여준다.

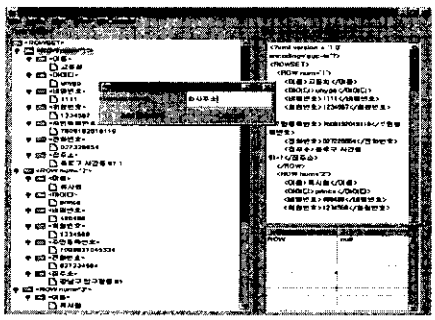

그림 4 XML문서에 엘리먼트 추가 화면

트리 관련 프로세스는 현재 선택된 엘리먼트의 정보를 일단 저장한 채, 추가될 엘리먼트의 정보를 입력 받고 현재 선택된 노드의 엘리먼트 타입을 조사해 이에 알맞은 insert매소드를 호출함으로써 적절한

곳에 엘리먼트를 추가하게 된다. 또한 트리구조에서 특정 엘리먼트를 선택 시, getNodeType 메소드를 사용하여 이에 대한 정보를 얻고 이것이 그림 4 의 오른쪽 아래창에 나타난다.

4.2 XML문서 변환 인터페이스

기존에 만들어져 있는 데이터베이스에 현재의 XML문서를 파싱하여 저장하고, 마찬가지로 기존에 만들어져 있는 데이터베이스로부터 XML문서로 만들고자 하는 내용에 대한 쿼리문을 작성함으로써 이에 대한 XML문서를 생성할 수 있다. 이의 구조는 새 창을 띄어 사용자로부터 데이터베이스에 접속하기 위한 정보를 입력받고 이를 이용해서 JDBC드라이버를 이용해 데이터베이스에 접속하고 현재 창에 보이는 XML 문서를 읽어 OracleXmlSave 객체를 만들어 이를 해당 테이블에 저장한다. 또한 SQL문을 통해 이에 대한 결과가 유효하다면 이를 적절한 사용자가 지정한 형식에 의한 XML문서를 생성하여 보여주기 위해 OracleXmlQuery 객체를 만들고 getXmlString 메소드를 이용하여 원하는 XML문서를 생성한다.

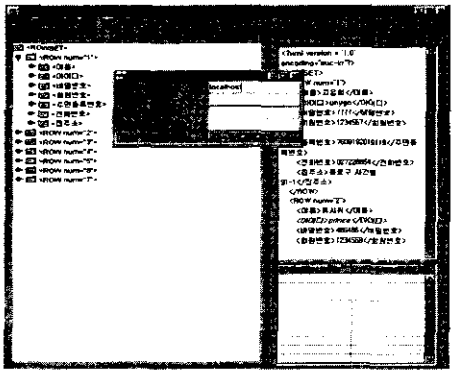

그림 5 XML문서를 데이터베이스에 저장하는 화면

그림 6 문서 저장 확인 화면

또한 XML문서는 순서 정보와 반복 정보가 있다는 특성을 고려할 경우, 이를 그대로 관계형 데이터베이스에 사상할 때 발생하는 데이터의 손실을 막기 위해 객체 관계형 데이터베이스인 오라클을 사용해 객체형(Object Type)과 중첩 테이블(Nested Table)을 사용함으로써, 복잡한 구조의 XML문서에 대해서도 처리가 가능하도록 하였다. 즉, DBA가 이에 맞게 직접 스키마만 선언하여 구성해 준다면, 어떤 구조의 XML문서도 데이터베이스로의 저장과 생성을 자유롭게 할 수 있다. 이를 위한 데이터 베이스 스키마의 구성은 그림 7과 같다.

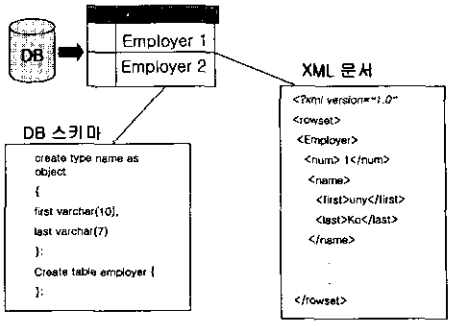

그림 7 DB 스키마의 설계

5. 결론 및 향후 연구과제

본 논문에서는 데이터베이스로의 저장기능을 갖는 XML 에디터를 제안하고 이를 구현하였다. 기존의 XML 에디터들이 XML문서를 편집하고 DTD를 정의함으로써 이에 대한 유효성 검사를 지원하는 기능은 있으나, 이러한 XML문서를 파싱하여 기존에 존재하는 데이터베이스에 저장하거나 기존의 데이터베이스의 내용을 XML로 만들어 주는 기능을 제공하지 않고 있다. XML문서가 전자상거래에서 문서 교환을 위해 사용된다는 점과 XML문서 자체의 내용이 데이터베이스와 불가분의 관계라는 것을 고려할 때, 이런 XML문서와 데이터베이스의 상호 연계에 대한 처리가 필요하다. 따라서, 본 논문에서는 실제 XML문서를 파싱하여 데이터베이스에 저장하고 자신이 원하는 쿼리를 보냄으로써 그 결과를 XML문서로 변환하여 보여주는 기능을 구현하였다. 또한 화면에서 XML문서의 트리구조와 소스코드를 동시에 보여주어 직접 구조를 보면서 소스코드를 편집할 수 있도록 함으로써, 실제 XML문서에 익숙하지 않은 사용자도 쉽게 편집기를 통해 XML문서를 만들고 편집할 수 있도록 하였다.

향후 연구 과제는 오라클 이외에 Uni-SQL이나, my-SQL, MS-Access등의 다른 데이터베이스와의 연동도 가능하도록 함으로써 어떤 데이터베이스로도 XML문서를 저장하거나 이로부터 만들어 낼 수 있도록 보완할 예정이다.

참고문헌

[1] W3C, "W3C Recommendation: Extensible Markup Language(XML) 1.0"

[2] W3 Consortium "Document Object Model (DOM)", http://www.w3.org/DOM/

[3] Refsnes Data co., "Welcome to XML School", "http://www.w3schools.com/xml

[4] SUN mircrosystem, "Java Technology and XML", http://java.sun.com/xml

[5] XML and Database, Ronald Bourret Technical University of Darmstadt September, 1999, http://www.infomatik.tudarmstadt.de