버저닝을 지원하는 XML 저장관리시스템 설계 및 구현

손충범°, 유재수
충북대학교 정보통신공학과 및 컴퓨터·정보통신연구소
cbson@netdb.chungbuk.ac.kr, yjs@cbucc.chungbuk.ac.kr

Design and Implementation of an XML Repository System Supporting Versioning

Chung Beom Son°, Jae Soo Yoo
Dept. of Computer & Communication Engineering. and Computer & Information Communication Research,
ChungBuk National University

요 약

최근 웹을 이용한 전자문서의 중요성이 부각되면서 대용량의 XML 문서에 대해 효율적으로 저장하고, 검색하며, 관리할 수 있는 XML 저장관리 시스템의 연구가 활발히 진행되고 있다. XML 응용 중에서 특허 문서 관리, 소프트웨어 설계, 시스템 매뉴얼 등의 응용과 같이 수정된 기존의 문서들이 관리되어야 하는 분야에서 버전 관리 기능이 필요하다. 본 논문에서는 문서의 수정을 효율적으로 지원하는 분할모델을 이용하여 문서 수정에 따른 버저닝을 지원하는 데이터 모델을 제안하고, 버저닝을 지원하는 XML 저장관리 시스템을 설계하고 구현한다.

1. 서 론

월드와이드웹 컨소시움에 의해 제안된 XML(eXtensible Markup Language)은 HTML과 SGML의 장점을 수용하여 문서의 논리적인 구조를 표현하면서도 쉽게 사용할 수 있도록 만든 것으로 1998년 W3C에서 XML을 권고안으로 채택하였다. 현재 인터넷 웹 문서, 전자도서관, CSCW, EDI, EC, CALS 등을 포함한 다양한 분야, 수학 분야의 MathML, 채널 기술의 CDF, 무선인터넷에서의 WML 등에서 XML 연구는 상당히 활발하게 진행되고 있으며 이러한 분야에서의 많은 양의 XML 문서를 효율적으로 저장하고, 검색하며, 관리할 수 있는 시스템의 필요성이 점차 증대되고 있다[1].

이와 같이 XML을 활용한 다양한 응용분야에 적용되는 정보시스템을 구축하기 위해 가장 중요한 저장관리 시스템(repository system)의 기능은 XML 문서에 대한 정보검색과 정보관리를 위한 것이다. 이는 XML로 표현된 구조화된 정보의 특성을 반영하여 가장 적절하게 구현하고 기능을 제공하는 시스템으로 XML의 특징인 정보 구조화의 패러다임을 그대로 유지시킬 수 있는 정보검색이 가능하여야 한다. 즉, 엘리먼트 단위의 검색뿐만 아니라 엘리먼트 간의 논리적인 계층 구조를 이용한 검색 등을 수용할 수 있어야 한다. 또한 XML 문서에 표현할 수 있는 모든 정보에 대한 사용자의 요구에 부응하기 위해서는 XML 문서 내에 포함된 모든 정보를 손실 없이 저장하고 수정하며 관리 할 수 있어야 한다. 즉, 일반 텍스트 문서에 비해 XML 문서는 내용 정보뿐만 아니라 문서의 논리적인 구성을 기술하는 구조정보를 가지고 있기 때문에 이에 대한 대응이 필요하며, XML 문서가 가지고 있는 내용은 단순한 텍스트에서 다양한 멀티미디어를 포함할 수 있다. 이러한 XML 문서는 구조정보를 이용하여 문서를 효율적으로 관리하기 때문에 데이터베이스에 XML 문서, DTD, 구조정보 및 다양한 미디어를 저장, 관리해야 된다[1]. 또한 XML 문서에 대한 버전닝은 특허 문서 관리, 소프트웨어 설계, 시스템 매뉴얼 등의 응용과 같이 수정된 기존의 문서들이 관리되어야 하는 분야에서 필요한 기능이지만 대부분의 시스템들은 이를 적절하게 지원하지 못하고 있다.

본 논문에서는 문서의 수정을 효율적으로 반영할 수 있는 분할모델을 이용하여 문서 수정에 따른 버전닝을 지원하는 데이터 모델을 제안하고, 내용변경과 구조변경에 따른 버전닝 방법을 제시한다. 또한 테스트를 통해 설계한 버전닝을 지원하는 XML 저장관리 시스템을 검증한다.

본 논문의 구성은 다음과 같다. 2절에서 XML 저장관리 시스템의 기존 연구들을 살펴보고, 3절에서는 XML 저장관리 시스템의 설계한 스키마와 구성도 및 버전닝 방법을 설명한다. 4절에서는 구현한 XML 저장관리 시스템에 대해 기술하고, 마지막으로 결론 및 향후 연구 방향을 제시한다.

2. 관련연구

대용량의 XML 문서에 대한 효율적인 저장, 관리 및 검색을 지원하는 저장관리 시스템은 많은 정보 시스템 응용 분야에서 반드시 필요하다. 기존에 XML 저장관리 시스템의 연구는 XML 문서의 저장방식에 따른 연구를 크게 구분하면 분할모델과 비분할 모델로 구분할 수 있다. 분할 모델이란 XML 문서를 저장할 때 문서를 구성하고 있는 엘리먼트를 나누어 저장하는 방법으로 문서의 수정에 대해서는 효율적인 반면 해당 문서에 대한 검색이 발생한 경우 엘리먼트들을 재구성하여 검색 결과를 반환하는 과정에서 시스템의 성능을 저하시키는 문제가 발생한다. 비분할 모델은 XML 문서 전체를 저장한 후 각각의 엘리먼트는 위치정보를 가지고 접근하는 방식이다. 이는 문서를 한꺼번에 저장하였기 때문에 통합 과정이 필요 없어 문서 참조를 빨리 할 수 있지만, 내용의 일부만이 수정되었을 때도 문서 전체를 재구성해야 한다는 단점이 있다.

다음은 XML 문서의 버전 관리에 대해서 연구된 내용들을 살펴본다. RCS(Revision Control System)[2]와 같은 기존의 문서 버전 관리시스템은 라인 중심적이고 많은 제약과 성능 문제를 가지고 있다. RCS는 가장 최신의 버전을 완전히 저장하는 반면에 다른 모든 버전들은 역방향 편집 스크립트로서 저장된다. 이 스크립트들은 문서의 개발 이력에서 어떻게 역방향으로 가야하는지를 기술한다. 최신 버전을 제외한 다른 버전을 위해 구버전을 생성하기 위해 역방향 편집 스크립트를 적용해야 하는 추가의 프로세싱을 필요로 한다. RCS는 원본 문서의 논리적 구조를 유지하지 않는다. 이것은 XML 문서의 구조 검색을 어렵게 만들고 버전을 재구성하는데 비용이 많이 들게 된다.

편집기반 UBCC(Usefulness Based Copy Control)[3]는 편집에 바탕을 둔 버저닝 방법들의 성능향상을 위해서 제안되었다. UBCC는 주어진 버전의 유효한 객체들을 데이터 페이지들에서 클러스터를 한다. 이것은 페이지 유용성의 개념을 사용한다. 한 페이지에서 유효한 많은 객체들이 어떤 임계값 이하면 유효한 모든 페이지 객체들이 다른 페이지로 복사되어진다. 하나의 버전은 그 다음에 유용한 페이지들을 단지 액세스함으로써 재구성되어진다. 특정 문서의 버전을 재구성하는 것은 버전을 포함하는 객체들을 찾아서 그것들을 정확한 논리적 순서로 생성하는 것과 같다. 그러나 버전들의 편집에 바탕을 둔 표현은 문서 내용을 재배열한다든지 문서를 재구성하는 변경에 대해선 효율적으로 표현을 할 수가 없다. 이런 문제점을 해결하기 위해 편집스크립트를 제거하고 일반 세그먼트의 개념을 사용하는 복사에 바탕을 둔 버저닝 방법인 복사기반 UBCC[3]가 나왔다. 이 방법은 편집기반 UBCC에서 사용한 유용성을 기본으로 하는 페이지 관리를 확장하여 각 버전을 편집 스크립트없는 서브리스트들의 리스트를 구현함으로써 저장 및 검색을 수행한다. 이 복사 기반 방법은 변경없는 세그먼트 레코드들에 대해 편집 스크립트를 분리해서 가지고 있을 필요없이 리스트로 객체들을 저장할 수 있으며 그 리스트들 중간에서 변경이 생기면 단지 분할만 하면 된다. 그리고 문서 재구성과 같은 변경을 융통성 있게 처리할 수 있으며 버전 검색 I/O 오버헤드를 줄일 수 있다.

3. 버저닝을 지원하는 XML 저장관리시스템 설계

본절에서는 설계한 XML 저장관리시스템의 시스템구성도와 시스템을 구성하는 주요 모듈들에 대해 기술한다. 하지만 지면관계상 버저닝을 지원하는 데이터모델과 버전관리자, 문서병합관리자를 중심으로 설계한 내용을 기술한다.

3.1 시스템 구성도

그림 2는 구현한 XML 저장관리 시스템 구조를 보여 준다. 구현한 XML 저장관리 시스템은 XML 객체 관리자, XML 인덱스 관리자, 버전관리자, 문서병합관리자, 구조질의 처리기, 검색결과생성기로 구성된다. XML 저장관리 시스템을 구성하는 각 모듈의 역할은 다음과 같다.

XML 저장관리 시스템에서 가장 핵심적인 기능을 담당하는 XML 객체관리자에서는 실제 XML 문서를 저장하기 위한 스키마 생성 및 XML 문서 인스턴스의 저장 및 추출을 담당한다. 버전관리자는 XML 문서에 대한 버전관리를 담당하고, 문서병합관리자는 요청한 문서에 대한 병합을 담당한다.

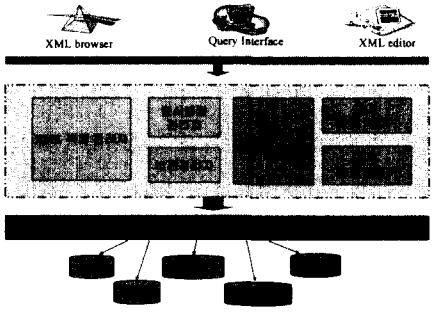

그림 2 저장 관리 시스템 구성도

XML 인덱스 관리자에서는 내용 검색만을 위한 내용 색인기, 구조 검색을 지원하는 구조 색인기, 애트리뷰트 검색을 지원하기 위한 애트리뷰트 색인기로 구성된다. 구조질의 처리기는 XML 문서에 대한 다양한 검색을 처리하기 위한 모듈로 실제 질의 처리 시 각 질의 타입에 따라 SQL을 생성하는 모듈이다. 또한 검색결과생성기에서는 구조질의 처리기에서 처리한 검색결과를 가지고 XML 객체관리자의 XML 인스턴스 관리기를 통하여 문서 전체 또는 일부분을 사용자에게 제공하는 역할을 수행한다.

3.2 스키마 설계

본 논문에서 설계한 XML 데이터 모델은 그림 1과 같다. DTD 테이블은 저장할 XML 문서의 DTD를 저장하고 관리한다. 엘리먼트 테이블은 XML 문서를 분할하여 각 엘리먼트별로 저장하고 관리한다. 멀티미디어 테이블은 XML 문서 내에 포함된 멀티미디어를 저장하고 관리한다. 버전 테이블은 XML 문서에 대한 버전닝이 발생할 경우 가장 마지막에 버전된 문서의 정보를 유지하는 테이블이다. 마지막으로 DOC 테이블은 XML 문서에 대한 파일명을 유지하는 테이블이다.

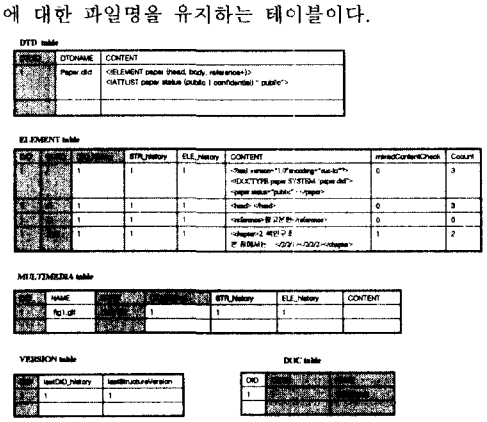

그림 1 XML 데이터 모델링

3.3 버전 관리자

기존의 문서와는 다른 구조화된 특성을 갖는 XML 문서는 내용만을 삽입하거나 삭제할 수도 있지만 문서의 구조가 변하면서 내용이 추가되거나 삭제되는 경우도 있다. 그렇기 때문에 XML 문서에 대한 수정은 크게 내용 변경과 구조 변경을 동시에 고려하여야만 한다.

이에 본 논문에서는 버전 관리를 크게 두 가지 즉, 문서의 구조는 변하지 않고 단지 내용만이 수정되는 경우와 문서의 구조가 바뀌면서 내용이 수정되는 경우로 나누어 내용 변경 시에는 내용 부분에 대해서만 버전을 유지하는 엘리먼트 단위 버전을 제공하고, 구조 변경 시에는 구조정보가 달라지기 때문에 문서 단위 버전을 제공한다.

XML 버전 관리자는 수정된 기존의 XML 문서에 대한 버전 관리를 할 수 있도록 지원하는 모듈이다. 처음 새로 생성된 XML 문서 인스턴스는 엘리먼트 테이블의 DID_history, ELE_history, STR_history 필드에 버전 1을 부여받은 이후에 문서에 대한 수정이 이루어지면 사용자는 새로운 버전에 추가할 것인지를 판단한다. XML 문서에 대한 버전 관리는 엘리먼트 테이블의 DID_history 필드에서 관리를 한다. DID_history 필드 값은 원본 XML 문서에 대한 버전이 수행될 경우 지속적으로 유지하는 버전 번호이다. XML 문서에 대한 수정이 내용만이 변경될 경우에는 엘리먼트 테이블의 ELE_history 필드에 버전 정보를 기술하여 엘리먼트 단위 버전을 지원하며, 문서의 구조가 변경되었을 경우에는 전체 XML 문서의 구조정보가 바뀌게 되므로 내용 변경과 구분하기 위해 엘리먼트 테이블의 STR_history 필드에 버전 정보를 기술한다. 이때 이전의 버전닝된 문서가 내용만이 변경된 경우에는 ELE_history 필드 값을 리셋 시킨다. 또한 사용자가 특정 문서를 요구할 때 디폴트로 마지막에 버전닝된 문서를 빠르게 추출하기 위하여 VERSION 테이블의 lastDID_history 필드 값을 DID_history 값으로 대치시킨다. 이때 문서의 구조가 변경된 경우에는 lastStructureVersion 필드에 엘리먼트 테이블의 STR_history 필드 값으로 대치시켜서 구별한다.

그림 3은 DID가 1인 문서에 대한 내용 변경이 발생할 경우 ELEMENT 테이블과 VERSION 테이블을 보여주고 있다.

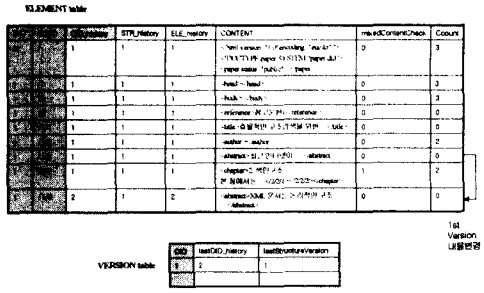

그림 3 내용 변경시 테이블

3.4 문서병합 관리자

XML 인스턴스 관리기는 사용자가 요구하는 문서와 문서에 포함된 멀티미디어를 데이터베이스로부터 추출하는 기능을 담당한다. XML 인스턴스 관리기는 문서 전체에 대해서는 해당 문서에 대한 DID나 문서 명을 통해서 추출이 된다. 문서의 일부분인 엘리먼트에 대해서는 해당 문서의 DID와 해당 엘리먼트의 SORD를 통해 추출한다.

본 논문에서는 XML 문서를 각 엘리먼트별로 분할하여 저장관리하기 때문에 문서를 추출하는 경우 먼저, 문서를 구성하는 엘리먼트들을 추출한 다음 이를 이용하여 문서를 통합하는 과정이 필요하다. 먼저, 문서를 구성하는 엘리먼트 추출 시 XML 인스턴스 관리기는 DID_history와 STR_history 값을 구한 다음 문서를 구성하는 전체 엘리먼트들을 추출한다. 이를 위해 가장 마지막에 구조 변경된 문서를 구성하는 엘리먼트들을 추출하고 이후에 내용 변경된 엘리먼트들이 존재한다면 이 부분의 엘리먼트들도 추출한다. 추출한 엘리먼트에서 SORD 값을 비교하여 중복되는 SORD에 대해 DID_history가 제일 큰 엘리먼트만 남기고 나머지는 제거한다. 문서 통합 과정은 ELEMENT 테이블에서 추출한 SORD, CONTENT, mixedContentCheck, Ccount를 이용하여 문서 병합을 한다. 먼저 SORD 값으로 sorting을 하고 나서 문서 병합은 다음에 방법을 이용한다. Root 엘리먼트는 XML 선언부분 및 DTD 선언 부분과 시작 태그까지 기록한다. mixedContentCheck가 0 이면서 Ccount 값이 없으면 끝 태그까지 기록하고 mixedContentCheck가 0 이면서 Ccount 값이 있으면 하위 엘리먼트의 SORD 값을 가지는 엘리먼트로 이동하여 시작 태그를 기록한다. mixedContentCheck가 1 이면 Ccount 값은 반드시 존재하므로 내용이 먼저 나오는 경우 내용을 기록하고 하위 엘리먼트로 이동한다. "<" 가 먼저 나오면 해당 SORD 값을 갖는 하위 엘리먼트로 이동한다.

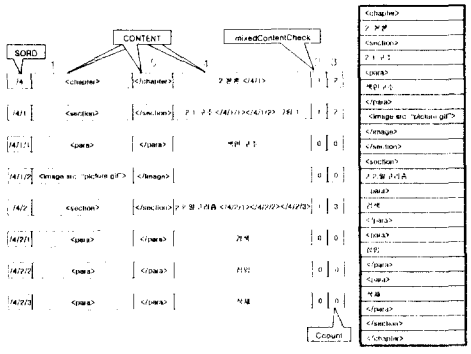

그림 4 문서 병합 예

그림 4은 문서의 일부분을 통합하는 과정으로 SORD 값으로 정렬을 한 후에 CONTENT 부분의 시작 엘리먼트를 버퍼에 기록하고 micedContentCheck 값과 Ccount 값을 검사하여 앞에서 설명한 대로 문서를 통합하는 과정을 보여주고 있다.

4. XML 저장관리시스템 구현 및 평가

구현한 XML 저장관리시스템은 자바를 사용하여 유닉스 환경에 RDBMS로 오라클 8.0.1을 사용하였으며, 웹 환경에서 지원하도록 아파치 웹서버 1.3.9와 서블릿과 JSP를 지원하는 톰캣 3.0을 사용하였다. 그림 5는 구현한 XML 저장관리시스템의 클래스 다이어그램이다.

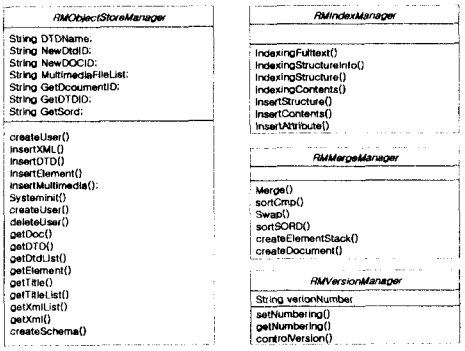

그림 5 클래스다이어그램

본 논문에서 구현한 XML 저장관리시스템을 검증하기 위해 테스트 문서로서 학위논문 XML 문서를 사용하였고, 내용변경과 구조변경을 통해 버저닝된 문서를 검색해 보았다. 저장된 문서에 대해 구조변경 및 내용변경을 수십회 수행한 후 버전기반의 구조검색과 내용검색을 수행한 결과 100% 원하는 결과를 얻을 수 있었다.

5. 결론

본 논문에서는 문서의 수정을 효율적으로 지원하는 분할모델을 이용하여 문서 수정에 따른 버전닝을 지원하는 데이터 모델을 제안하고, 버전닝을 지원하는 XML 저장관리시스템을 설계하고 구현하였다. 지원되는 버전관리는 문서의 구조는 변경되지 않고 단지 내용만이 수정되는 경우와 문서의 구조가 바뀌면서 내용이 수정되는 경우로 나누어진다. 내용만 변경되는 경우에는 변경된 부분에 대해서만 버전을 유지하는 엘리먼트 단위버전을 제공하고, 구조 변경시에는 구조 정보가 달라지기 때문에 문서 단위 버전을 제공한다.

향후 연구 방향으로는 현재 구조 변경시에는 문서 전체에 대해서 버전을 유지하는데, 구조 변경된 부분에 대한 구조 검색을 지원할 수 있도록 동적으로 구조정보를 표현하는 연구가 필요하다.

참고문헌

[1] Hyung-Il Kang, Jae Soo Yoo, ByoungYup Lee, "Design and Implementation of a XML Repository System Using DBMS and IRS" XML Asia Pacific, 2000.

[2] Walter F. Tichy, "RCS-A SYstem for Version Control," Software-Practice & Experience 15,7, July 1985, pp.637-654

[3] Shu-Yao Chien,T sotras V.J., Zaniolo. C, "Copy-based versus edit-based version management schemes for structured documents,"Research Issues in Data Engineering, 2001. Proceedings. pp.95-102, 2001.