XML 문서관리를 위한 데이터 모델 설계 및 성능평가

Design and Performance Evaluation of Data Models for the XML Document Management

유 재 수* 손 충 범** 조 혜 영***
Jae-Soo Yoo Chung-beom Son Hye-Young Cho

요 약

최근 다양한 분야에서 XML이 인터넷상에서 정보교환의 표준으로 자리잡아 가고 있다. 따라서 그동안 XML 문서를 데이터베이스에 저장하고 검색하기 위한 데이터 모델링에 관한 연구가 활발히 진행되어 왔다. 그러나 기존의 연구들은 동적 환경에서 문서 변경시 버전을 지원하면서 빠른 문서 검색을 효율적으로 지원하지 못하는 단점이 있다. 본 논문에서는 XML 저장관리시스템에서 사용되고 있는 비분할 모델과 분할 모델의 장점들을 수용하여 동적 환경에 적합한 혼합 모델링 방법 4가지를 제안한다. 또한 시뮬레이션을 통하여 제안한 혼합형 모델링 기법의 우수성을 보인다. 이를 통해 다양한 동적 응용에 적합한 데이터 모델링을 제시하고자 한다.

Abstract

Recently, in various fields XML has been become a standard for information exchange in internet. Therefore, the researches on data modeling for storing and fetching the XML documents have been made actively. However, existing researches have weak points that they can support neither versioning nor fast fetching of documents while changing documents in dynamic environments. In this paper, we propose four kinds of hybrid data modeling schemes that combine fragmentation model and nonfragmentation model. Our data modeling schemes are suitable to dynamic environments. We also present guidelines that our hybrid data modeling schemes can be used for various applications. We show through various experiments that our hybrid schemes partially outperforms the existing modeling schemes in terms of insertion time, storage space and retrieval time.

1. 서 론

최근 인터넷을 기반으로 하는 웹의 급속한 성장으로 인해서 다양한 분야의 정보를 인터넷을 통해서 얻을 수 있게 되었다. 웹과 관련한 표준으로서 지금까지 HTML이 가장 많이 이용되어 왔으나, HTML은 문서에 대한 구조를 표현하지 못하고 자신이 인식할 수 있는 태그만 표현하는 등 여러 제약사항이 있다. 이와 같은 불편함을 해소하고 사용자의 다양한 요구를 수용하기 위해 W3C (World Wide Web Consortium)에서는 새로운 표준으로서 XML(eXtensible Markup Language)을 제안하였다. XML은 웹상에서 구조화된 문서를 전송 가능하도록 설계된 표준화된 텍스트 형식으로써, 문서 내용을 기술하는 것보다는 문서의 내용과 연관된 태그를 사용자가 직접 정의할 수 있도록 하였다. 이러한 XML은 현재 인터넷상에서 뿐만 아니라 전자출판, 의학, 경영, 법률, 판매 자동화, B2B(Business-to-Business : 사업자와 사업자 간의 거래), EDI(Electronic Data Interchange : 전자 문서 교환), EC(Electronic Commerce : 전자 상거래), KMS (Knowledge Management System : 지식 관리 시스템), MathML(수학), MusicML(음악), HDML(Handheld Device Markup Language), VoiceML(음성인식 시스템), 디지털 전자도서관, 국가 공문서 표준 등 매우 광

* 정회원 : 충북대학교 정보통신공학과 부교수
yjs@cbucc.chungbuk.ac.kr
** 준회원 : 충북대학교 대학원 정보통신공학과 박사과정
cbson@trut.chungbuk.ac.kr
*** 비회원 : 충북대학교 대학원 정보통신공학과 석사과정 졸업
sunita@korea.com

범위하게 이용되고 있기 때문에, 대용량의 XML 문서에 대한 저장과 검색을 지원하는 효율적인 관리 시스템의 필요성이 대두되었다.

구조화된 XML 문서를 효율적으로 데이터베이스에 저장하고 검색하기 위한 데이터 모델링에 관한 연구가 그동안 활발히 진행되어 왔다. 모델링 방법으로서 XML 문서 전체를 저장하는 비분할 저장 모델과, XML 문서를 엘리먼트 단위로 분할하여 저장하는 분할 저장 모델이 있다. 비분할 모델은 문서 전체를 한번에 저장하기 때문에, 문서 변경시 버전을 지원하기 위해서 문서 전체를 다시 저장하여야 하는 단점이 있다. 분할 모델은 엘리먼트 단위로 쪼개져 저장되므로, 문서 변경시 변경된 엘리먼트만을 반영할 수 있어 동적으로 버전 기능을 효율적으로 제공하지만, 전체 문서 검색시 나뉘어진 엘리먼트들을 병합하기 위해서 많은 시간이 요구된다.

이에 본 논문에서는 비분할 모델과 분할 모델의 장점들을 수용하여 동적 환경에 적합한 혼합 모델링 방법 4가지를 제안한다. 제안 1은 XML 문서를 분할하여 저장한 후 검색이 자주 발생하게 되는 최근문서에 대해서 미리 병합하여 전체문서를 유지하는 분할 우선 혼합 모델 방법이다. 제안 2는 XML 문서를 분할하여 저장한 후 중요문서에 대해서 미리 병합하여 전체문서를 유지하는 분할 우선 혼합 모델 방법이다. 제안 3은 처음에는 XML 문서 전체를 저장한 후에 변경이 일어난 경우에는 동적으로 엘리먼트 단위로 변경된 부분만을 저장하는 분할 모델 방법을 따르는 비분할 우선 혼합 모델이다. 제안 4는 제안 3의 방법과 동일한데, 변경이 많이 이루어질 경우 중간문서에 대해서도 처음문서처럼 비분할로 저장하여 문서 병합시 중간문서부터 병합을 할 수 있어 검색 시간을 줄이기 위한 비분할 우선 혼합 모델이다. 이 혼합 모델링 방법 4가지 각각에 대해서 특징들과 장단점을 살펴 봄으로써 여러 가지 동적 응용에 적합한 데이터 모델링을 선택할 수 있는 가이드라인을 제시하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서 기존 모델 방법인 비분할 저장 방식과 분할 저장 방식에 대하여 살펴본다. 3장에서는 구조정보 표현 및 전체적인 시스템 구성도에 대해 알아본 다음, 제안하는 4가지 혼합 모델링에 대해서 각각의 특징과 장단점을 살펴보고 순서도와 알고리즘을 기술한다. 4장에서는 문서 저장 시간과 저장 공간, 검색 시간에 대해서 성능평가를 기술하고 각 모델들에 적합한 응용분야를 살펴본다. 마지막으로 5장에서는 본 논문의 결론 및 향후 연구 방향을 제시한다.

2. 관련 연구

XML 저장관리 시스템을 개발하기 위해서 제일 먼저 선행되어야 하며 가장 중요한 역할을 차지하는 것은 데이터 모델링 부분이다. 데이터 모델링이란 데이터와 데이터들간의 관계를 기술하는 개념적 도구로서, 데이터베이스의 논리적 구조를 명시하는 과정이다. 데이터 모델링은 우선 DBMS의 활용에 따라 관계형 모델과 객체 지향 모델로 나누어진다.

관계형 모델은 현재 가장 많이 사용하고 있는 관계형 데이터베이스 관리시스템(RDBMS)을 기반으로 하고 있으며 데이터를 테이블 형태로 관리하기 때문에 사용자의 쉬운 접근이 가능하다. 그러나 RDBMS에서는, 테이블과 튜플 수가 기하급수적으로 늘게 되는 경우 조인 연산으로 인한 시스템의 성능이 저하되는 단점이 있다. 객체 지향 모델은 객체 지향 데이터베이스 관리시스템(ODBMS)에서 지원하는 객체지향 개념을 이용할 수 있기 때문에 상속과 같은 객체지향 특성을 이용할 수 있으며, 엘리먼트 간의 전후종속 관계를 클래스에 기반한 객체들간의 링크로 나타낼 수 있다는 장점이 있다.

저장방식에 따른 분류에는 XML 문서 전체를

한꺼번에 저장하는 분할 저장 모델과 문서를 엘리먼트 단위로 나누어 저장하는 비분할 저장 모델로 나누어진다.

2.1 비분할 모델

XML 문서 전체를 저장하는 방식이다. XML 저장 관리 시스템에서는 전체문서뿐만 아니라, 엘리먼트 단위의 검색까지 지원해야 한다. 이때 엘리먼트들은 문서 안에서 계층상의 구조 정보를 가지고 있고 이로 인해서 모든 레벨에서 수평적 관계 및 수직적인 관계의 표현이 가능하다. 그림 1과 같이 각 엘리먼트에 대한 구조 정보 및 위치 정보를 유지하기 위해서 별도의 테이블을 만들고, 엘리먼트가 문서 안에서 나타난 시작 오프셋과 끝 오프셋에 관한 정보를 엘리먼트 구분자와 함께 기록한다. 문서 전체는 그림 2와 같이 한꺼번에 저장하므로, 전체 문서에 대해서는 빠른 검색이 가능하며, 각 엘리먼트에 대한 정보를 유지하므로 엘리먼트 단위의 검색도 지원한다. 그러나

DID	SORD	VERSION	START_OFFSET	END_OFFSET	ETID	Scount	Ccount
1	/	1	1	125	00	0	3
1	/10	1	10	30	0001	0	3
1	/20	1	31	115	0002	0	3
1	/20/10	1	40	60	000203	0	0
1	/20/20	1	61	75	000204	0	0
1	/20/30	1	76	95	000205	0	2
1	/20/40	1	96	105	000206	0	0

(그림 1) 비분할 모델의 엘리먼트 테이블

DID	VERSION	XMLNAME	FULLTEXT
1	1	조혜영.xml	<?xml version="1.0" encoding="EUC-KR" standalone="no"?> <!DOCTYPE paper SYSTEM "paper.dtd"> <paper> <title> 제목 </title> <abstract> 요약 </abstract> <chapter> 1.서론 ...
1	2	조혜영.xml	
2	1	정소영.xml	
2	2	정소영.xml	
2	3	정소영.xml	
3	1	민지영.xml	

(그림 2) 비분할 모델의 문서 테이블

문서의 일부분이라도 변경되었을 경우, 변경된 부분만을 효율적으로 반영하지 못하고 문서 전체를 다시 저장해야 하는 단점이 있다.

2.2 분할 모델

그림 3에서처럼, XML 문서에 대해서 엘리먼트 단위로 나누어서 저장하는 방식으로 문서의 실제 내용을 가지고 있는 각각의 단말 엘리먼트 안에 문서의 내용이 나뉘어 저장된다. 따라서 문서의 구조적 정보나 일부 내용들이 수정되었을 때 관계 있는 엘리먼트만 수정하면 되므로 문서의 버전 관리가 쉽고, 동일한 내용을 갖는 노드들을 공유할 수 있다는 장점이 있다. 그러나 XML 문서 전체를 추출하고자 하는 경우 각 단말 노드들을 순회하며 통합하는 과정에서 많은 시간이 소요하게 된다는 단점이 있다.

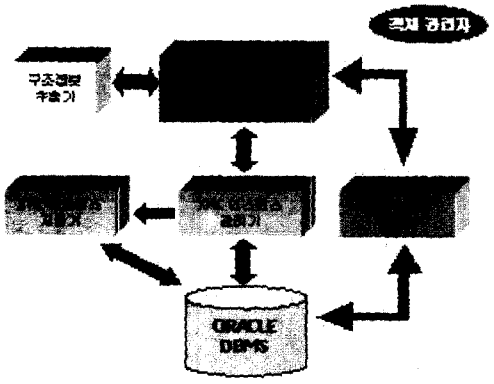
DID	SORD	VERSION	CONTENT	mixedCheck	Ccount
1	/	1	<?xml version="1.0"encoding="euc-kr"?> <!DOCTYPE paper SYSTEM "paper.dtd"> <paper status="public"> </paper>	0	3
1	/1	1	<head></head>	0	3
1	/2	1	<body> </body>	0	3
1	/3	1	<reference>참고문헌1</reference>	0	0
1	/1/1	1	<title>효율적인 구조검색을 위한 ... </title>	0	0
1	/1/2	1	<author></author>	0	2
1	/1/3	1	<abstract>최근 인터넷이 ....</abstract>	0	0
1	/2/2	1	<chapter>2. 색인구조 본 장에서는 ...</2/2/1></2/2/2></chapter>	1	2
1	/1/3	2	<abstract>XML 문서는 논리적인 구조...</abstract>	0	0

(그림 3) 분할 모델의 엘리먼트 테이블

3. 제안하는 데이터 모델링

3.1 시스템 구성도

본 논문에서 가장 핵심적인 역할을 담당하는 XML 객체 관리자의 구성도는 다음 그림 4와 같다. XML 객체 저장 관리자는 새로운 DTD(Document Type Definition)가 들어오면 XML 스키마 생성기를 통해서 DBMS에 스키마를 생성한다. 새 XML



(그림 4) 객체 관리자의 구성도

문서가 들어오면 구조정보 추출기를 호출하여 구조정보를 추출하고, XML 인스턴스 저장기를 통해서 문서를 저장하며, XML 인스턴스 관리기를 통해서 문서를 검색한다. 하부 저장 시스템으로 RDBMS의 하나인 ORACLE8을 사용하였다.

XML 문서의 구조정보를 표현하기 위해 본 논문에서 사용하는 구조정보는 엘리먼트 이름을 식별할 수 있는 ID, 부모와 자식 엘리먼트간의 계층정보, 동일한 부모 엘리먼트를 갖는 자식 엘리먼트들의 순서정보, 그리고 동일한 부모 엘리먼트를 갖는 자식들 중 동일한 타입의 엘리먼트들에 대한 순서정보와 버전정보로 구성된다.

엘리먼트를 식별할 수 있는 EID(Element ID)는 DTD에서 정의된 각 엘리먼트에 대하여 유일한 ID를 의미한다. EID는 2바이트를 사용하여 표현하는데 각 바이트는 '0' → '9' → 'A' → 'Z' → 'a' → 'z' 순으로 된 62개의 문자를 사용하며 ASCII 코드의 순서를 따르고 있다. 이를 통해 EID는 총 3844개의 엘리먼트를 표현할 수 있다.

부모와 자식 엘리먼트간의 계층정보를 표현하기 위해 ETID(Element Type ID)를 사용하는데 이 ETID는 앞에서 생성된 EID를 이용하여 만들게 된다. 즉, 부모의 ETID에 자신의 EID를 붙여서 자신의 ETID를 생성하게 되는데 부모가 없는 루트 엘리먼트인 경우는 자신의 EID를 ETID로 사용한다. 이 ETID는 문서의 논리적 구조상의 각 엘리먼트 타입에 부여되는 유일한 값이다.

엘리먼트의 순서정보를 표현하기 위해 형제 엘리먼트들에 대한 순서정보를 나타내는 SORD (Sibling ORDer)와 동일한 타입의 형제 엘리먼트들 간의 순서정보를 나타내는 SSORD(Same Sibling ORDer)를 사용한다. 또한 특정 엘리먼트의 자식 검색을 위해 엘리먼트가 갖는 자식의 수를 Ccount를 사용하여 유지한다.

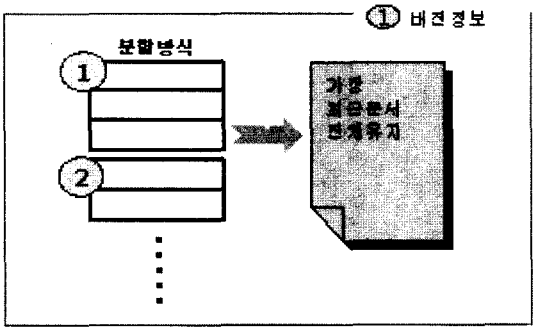
XML 문서의 내용이 변경되었을때 XML 저장관리 시스템에서는 가장 최근 문서뿐만 아니라 중간의 모든 문서들에 대해서도 정보를 유지해야 한다. 그러므로 문서 구분자로서 버전정보가 DID와 함께 사용된다. 어떤 한 문서가 변경되었을 때, DID는 동일하며 버전정보만 다르게 된다.

3.2 제안하는 기법

본 논문에서는 비분할 모델과 분할 모델의 장점만을 수용하는 혼합 모델링 방법 4가지를 제안한다. 비분할 모델이 가지는 전체 문서에 대해 빠른 검색 시간을 제공한다는 장점과 분할 모델이 가지는 장점인 문서 변경시 변경된 부분만을 동적으로 반영할 수 있다는 점을 최대한 수용하였다.

3.2.1 HMFM1 : 분할 모델 우선 중심의 혼합형 모델 기법(Hybrid Model based on Fragmentation Model 1)

분할 저장 모델과 비분할 저장 모델을 혼용하여 사용하는 첫 번째 혼합 모델링 방법으로 분할 저장 모델이 중심을 이룬다. 이 혼합 모델링 방법에서는 문서를 저장할 때 분할 저장 방식을 따른다. 처음 문서는 엘리먼트 단위로 쪼개져서 버전정보 1로 엘리먼트 테이블에 저장되고, 문서가 변경되면 변경된 엘리먼트에 대해서만 버전 정보가 하나 증가하여 엘리먼트 테이블에 저장된다. 변경될 때마다 가장 최근 문서에 대해서 미리 병합하여 전체문서를 유지한다. 항상 가장 최근 문서



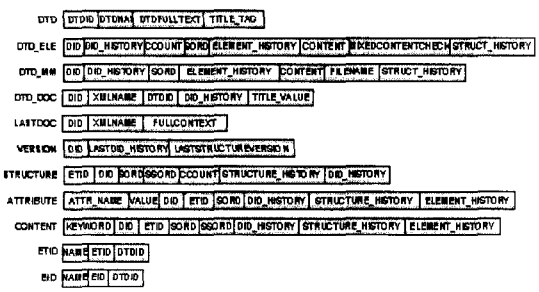
(그림 5) HMFM1 데이터 모델

하나만 최근 문서 테이블에 저장된다.

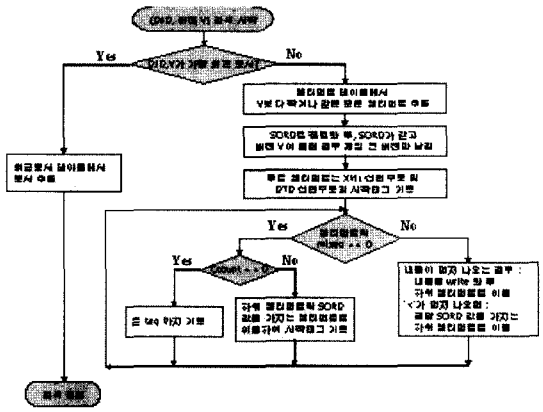
본 모델의 스키마 구조는 그림 6과 같다. DTD 테이블은 DTD를 저장하고 관리하며, DTD_ELE, DTD_MM, DTD_DOC 테이블은 각각 엘리먼트와 멀티미디어 그리고 XML 문서에 대한 파일명 및 가장 나중 버전 정보를 가진다. LASTDOC 테이블에는 가장 최근문서가 저장되고, VERSION 테이블에는 버전 정보를 유지한다. 구조 검색, 애트리뷰트 검색, 내용검색을 위한 인덱싱 정보를 저장하기 위해 STRUCTURE, ATTRIBUTE, CONTENT 테이블을 각각 사용한다. ETID 테이블은 엘리먼트에 대한 계층정보를 가지고, EID 테이블에는 엘리먼트 식별자에 대한 정보를 가진다.

분할 모델에서의 문서 저장 과정은 다음과 같다. 새로운 XML 문서가 들어오면 구조정보 추출기를 호출하고 그 결과물인 각각의 엘리먼트에 대한 정보들을 엘리먼트 테이블에 저장한다.

최근 문서에 대한 병합 과정은 분할 저장 방식



(그림 6) HMFM1 데이터 모델의 스키마 구조



(그림 7) HMFM1 데이터 모델의 검색 순서도

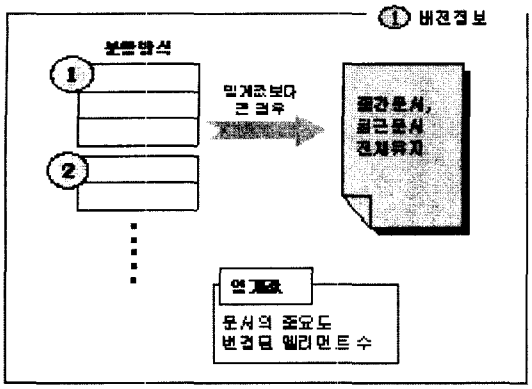
의 전체 문서 병합과 같다. 엘리먼트 테이블에서 전체 엘리먼트를 추출한 후, 동일한 SORD에 대해서는 VERSION이 제일 큰 엘리먼트만 남기고 제거한다. 그런 다음 SORD로 정렬한 후 병합한다. 병합이 이루어지면 최근문서 테이블에 저장된다.

이 모델링 기법의 검색 순서도는 그림 7과 같다.

이 모델링 기법은 XML 문서를 분할 저장 방식으로 저장하며, 최근 문서에 대해서만 전체 문서를 유지하기 때문에 문서 변경이 일어날 때마다 문서 병합을 하여야 하는 오버헤드가 있다. 그러나, 주로 최근 문서에 대한 검색이 이루어질 경우 빠른 응답시간을 제공하므로 분할 저장 방식의 단점을 보완할 수 있으며, 문서 변경시 변경된 부분만의 반영이 가능하며 이로 인해서 저장공간을 줄일 수 있는 분할 저장 방식의 장점을 그대로 가진다.

3.2.2 HMFM2 : 분할 모델 우선 중심의 혼합형 모델 기법(Hybrid Model based on Fragmentation Model 2)

분할 저장 모델과 비분할 저장 모델을 혼용하여 사용하는 두 번째 혼합 모델링 방법으로 분할 저장 모델이 중심을 이룬다. 이 혼합 모델링 방법에서도 첫 번째 모델링 방법과 마찬가지로 문서를 저장할 때 분할 저장 방식을 따른다. 첫 번째



(그림 8) HMFM2 데이터 모델

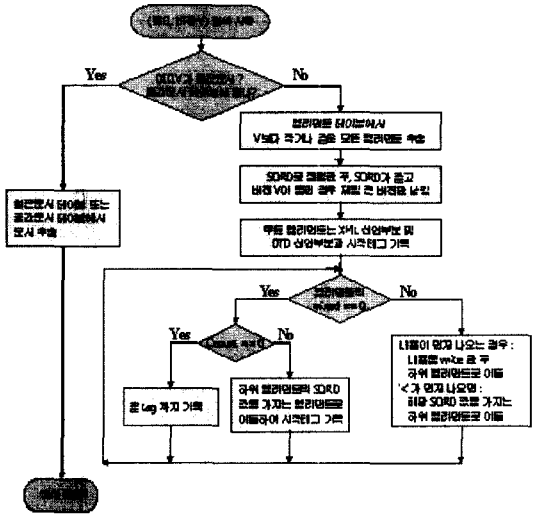
모델링 기법에서는 주로 검색이 이루어지는 최근 문서에 대해서만 전체 문서를 유지하는데 반해, 두 번째 모델링 방법은 최근 문서뿐만 아니라 중간 문서에 대해서도 전체 문서를 유지한다. 여기에서 중간 문서란 검색이 자주 이루어질 만한 중요문서가 될 수도 있으며(사용자 지정시), 문서 변경이 많이 이루어져 엘리먼트 변경 횟수가 특정 임계값을 초과하는 경우가 될 수 있다.

스키마 구조는 최근 문서와 중간 문서가 FULLDOC 테이블에 저장된다는 점만 제외하면 첫 번째 모델링과 동일하다.

문서 병합 과정은 분할 저장 방식을 따르므로 첫 번째 모델링 기법과 동일하다. 문서병합이 이루어지면 최근문서는 FULLDOC 테이블에 저장되며, 이 문서가 중요 문서이거나 변경이 많이 이루어진 경우 중간 문서로 판단되어 없어지지 않고 계속 테이블에 유지된다.

검색 순서도는 다음 그림 9와 같다.

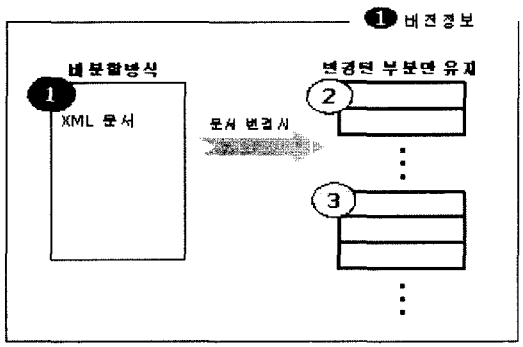
첫 번째 모델링 기법과 동일하게 이 모델링 기법도 XML 문서를 분할 저장 방식으로 저장하며, 최근 문서에 대한 전체 문서를 유지하기 위해서 문서 변경이 일어날 때마다 문서 병합이 필요하고, 중간 문서에 대해서도 전체 문서를 유지하므로 중복 저장으로 인한 저장 공간의 낭비가 있다. 그러나, 문서 검색이 최근 문서에 대해서나 중간 문서에 대해서 이루어질 경우에 빠른 검색시간을 제공할 수 있다는 장점이 있다.



(그림 9) HMFM2 데이터 모델의 검색 순서도

3.2.3 HMNM1 : 비분할 모델 우선 중심의 혼합형 모델 기법(Hybrid Model based on Nonfragmentation Model 1)

분할 저장 모델과 비분할 저장 모델을 혼용하여 사용하는 세 번째 혼합 모델링 방법으로 비분할 저장 모델이 중심을 이룬다. 이 혼합 모델링 방법에서는 맨 처음 XML 문서가 들어오면 비분할 방식으로 저장한다. 문서 전체는 DTD_DOC 테이블에 저장하며, 엘리먼트의 정보를 유지하기 위해서 별도의 DTD_ELE 테이블을 만들어 엘리먼트에 대한 모든 정보를 유지한다. 문서 변경이 일어나면 전체 문서를 다시 저장하는 대신, 변경

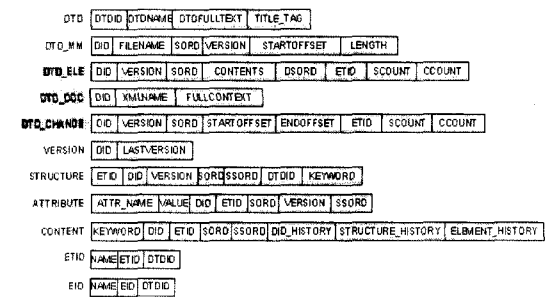


(그림 10) HMNM1 데이터 모델

된 부분만을 DTD_CHANGE 테이블에 저장한다. 즉, 처음 문서는 비분할 방식으로 저장하되, 문서 변경이 일어난 경우 분할 방식으로 저장하는 기법이다.

이 모델링 기법에서 사용하는 엘리먼트 테이블은 분할 저장 모델 방식의 엘리먼트 테이블과 조금 다르다. 분할 저장 방식에서는 엘리먼트의 내용변경이 일어났을 경우에는 동적으로 엘리먼트 테이블에 저장할 수 있었지만, 구조변경이 일어난 경우엔 버저닝을 위해 문서전체를 저장해야 한다. 그러나 이 모델링 기법에서는 구조변경이 일어나더라도 엘리먼트 테이블에 그 정보를 동적으로 반영할 수 있는데, 이를 위해서 새로운 엘리먼트가 삽입되었을 경우 새로운 엘리먼트의 구분자인 SORD를 기존의 SORD의 변경 없이 동적으로 할당한다. 단순한 내용 변경인지 구조적인 변경사항인지 구분하기 위해서 DSORD라는 애트리뷰트가 추가된다. 이 DSORD의 값이 엘리먼트의 SORD 값과 같으면 엘리먼트의 내용이 변경되었거나 삭제된 경우이며, DSORD가 SORD의 형제나 첫 번째 자식의 값이라면 엘리먼트가 추가된 경우이다. DSORD의 값이 NULL인 경우는 엘리먼트와 PCDATA가 혼합된 경우를 처리한다.

본 모델의 스키마 구조는 다음과 같다. DTD_CHANGE 테이블에는 변경된 엘리먼트에 대한 정보가 저장되고, DTD_DOC 테이블에는 처음 버전 문서 전체가 저장되며, DTD_ELE 테이블에는 처음문서에 대한 엘리먼트 정보가 기록된다.

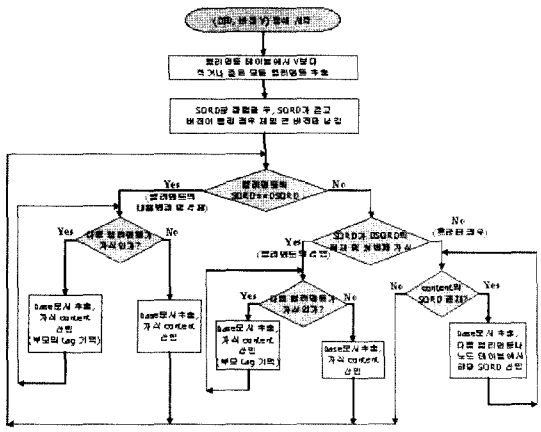


(그림 11) HMNM1 데이터 모델의 스키마 구조

이 혼합형 모델에서의 문서 저장 과정은 다음과 같다. 새로운 XML 문서가 들어오면 구조정보 추출기를 호출하고 그 결과물인 각각의 엘리먼트에 대한 정보들을 노드 테이블에 저장한다. 그리고 실제 XML 문서는 DTD_DOC 테이블에 저장한다. 문서의 변경이 이루어지면, 변경된 엘리먼트만을 DTD_CHANGE 테이블에 저장한다.

문서 전체에 대한 병합 과정은 다음과 같다. 변경이 일어난 엘리먼트는 엘리먼트 테이블을 참조하여 반영시켜야 한다. 이때, 여러번 변경된 엘리먼트는 맨 마지막에 수정된 사항만을 반영한다. 변경이 일어나지 않은 엘리먼트들에 대해서는 노드 테이블을 참조하여 그대로 추출한다. 전체 문서에 대한 검색 순서도는 그림 12와 같다.

이 모델링 기법은 처음 XML 문서에 대해서만 비분할 저장 방식으로 저장하고, 변경이 일어난 경우 분할 저장 방식을 따른다. 따라서, 문서의 변경이 이루어진 경우, 전체 문서를 다시 저장할 필요 없이 변경된 엘리먼트만을 효율적으로 반영할 수 있다. 저장공간 측면에서 비분할 저장 방식의 단점을 보완할 수 있으며, 문서 변경이 적은 경우 빠른 문서 검색을 할 수 있지만, 문서의 변경이 아주 많이 이루어질 경우에는 변경된 엘리먼트들을 모두 반영하여야 하므로 전체 문서를 병합하는데 시간이 많이 걸리게 되어 분할 저장 방식의 단점을 가지게 된다.



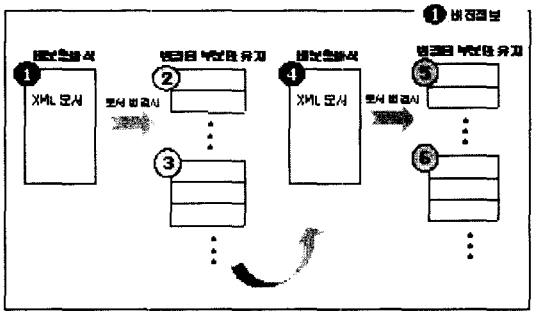
(그림 12) HMNM1 데이터 모델의 검색 순서도

3.2.4 HMNM2 : 비분할 모델 우선 중심의 혼합형 모델 기법(Hybrid Model based on Nonfragmentation Model 2)

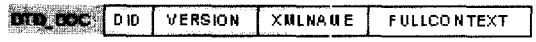
분할 저장 모델과 비분할 저장 모델을 혼용하여 사용하는 네 번째 혼합 모델링 방법으로 비분할 저장 모델이 중심을 이룬다. 이 혼합 모델링 방법은 세 번째 모델링 방법과 대부분 같지만 문서 변경이 이루어질 때마다 계속해서 엘리먼트 테이블에 정보를 유지하는게 아니라, 임계값을 넘도록 문서가 많이 변경될 경우 문서를 병합하여 다시 비분할 문서로 저장 관리한다.

세 번째 모델링 기법에서는 문서 변경이 많이 이루어진 경우 맨 마지막 버전의 전체 문서를 만들기 위해서 많은 시간이 요구되어진다. 네 번째 혼합 모델링 기법에서는 이 단점을 보완하기 위하여 문서의 변경이 많이 이루어지면 다시 비분할 문서인 중간문서로 유지한다. 이러한 중간문서는 DTD_DOC 테이블에 버전 정보와 함께 저장되며, 처음 문서와 마찬가지로 중간 문서 안의 엘리먼트에 대한 정보를 노드 테이블에 유지한다. 따라서 문서 변경이 많이 이루어졌을 경우에 세 번째 모델링 기법에서는 처음 문서부터 차례대로 반영해 주어야 하는데 반해, 본 모델링 기법은 중간 문서에서부터 반영시켜 주면 된다.

스키마 구조는 DTD_DOC 테이블에 처음 버전의 전체 문서뿐아니라 버전 정보를 가지는 중간 문서에 대해서도 저장된다는 점만 제외하면 세



(그림 13) HMNM2 데이터 모델



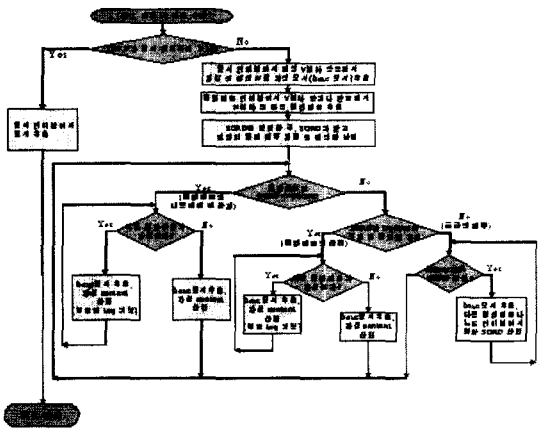
(그림 14) HMNM2 데이터 모델의 스키마 구조

번째 모델링과 동일하다.

이 혼합형 모델에서의 문서 저장 과정은 세 번째 모델링 방법과 동일하며, 단지 엘리먼트 변경 횟수가 특정 임계값을 초과하는 경우 문서를 병합하여 DTD_DOC 테이블에 버전 정보와 함께 저장하고, 노드 테이블에도 그 중간 문서에 대한 엘리먼트 정보를 유지한다는 점만 다르다.

전체 문서에 대한 검색 순서도는 그림 15와 같다.

세 번째 모델링 기법과 동일하게 이 모델링 기법도 처음 XML 문서에 대해서 비분할 방식으로 저장하고, 변경이 일어난 경우에만 분할 저장 방식을 따르므로, 문서 변경에 대해서 효율적으로 반영할 수 있다. 또한 문서 변경이 여러 번 이루어진 경우에는 중간 문서에 대해서 비분할 방식으로 저장하여 정보를 유지하므로, 맨 마지막 버전에 대한 문서 검색도 빠르다. 분할 저장 방법이나 비분할 저장 방법보다 부가적으로 저장해야 하는 정보는 많으나, 문서 변경시 동적으로 반영할 수 있고 어느 문서이든지 비교적 빠른 문서 검색시간을 가지는 장점을 가진다.



(그림 15) HMNM2 데이터 모델의 검색 순서도

4. 성능 평가

본 장에서는 효율적인 XML 문서관리를 위해 제안한 4가지 혼합 데이터 모델들의 성능을 기존의 분할 저장 방식, 비분할 저장 방식과 비교하여 평가한다. 비교를 위한 항목으로서 XML 문서의 저장 시간, 저장 공간, 검색 시간을 두었다.

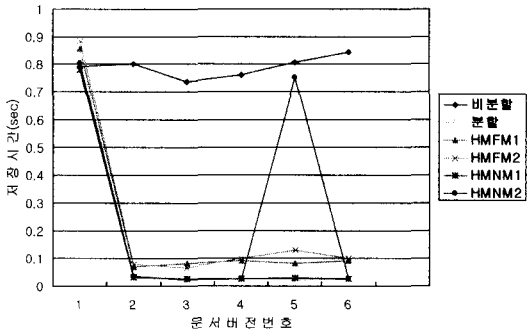
구현 언어로는 Java를 사용하였고, 구현환경으로 Enterprise JavaBeans를 지원하는 BEA사의 Weblogic 5.1.0 서버를 사용하였다. 또 개발 툴로서 Webgain 사의 VisualCafe 4.5를 이용하였다.

본 논문에서 제안한 혼합 모델링 기법들은 문서가 변경되었을 때 효율적이다. 문서가 변경되지 않는다면 모든 측면에서 비분할 모델이 최적이며, 문서 변경이 빈번히 이루어진 경우 저장 공간 측면에서는 분할 모델이 단연코 우수하지만 문서 검색 시간이 오래 걸리는 단점이 있다. 이에 본 실험에서는 문서변경 회수가 증가함에 따라서 저장 시간, 저장 공간, 검색 시간에 대해 살펴본다.

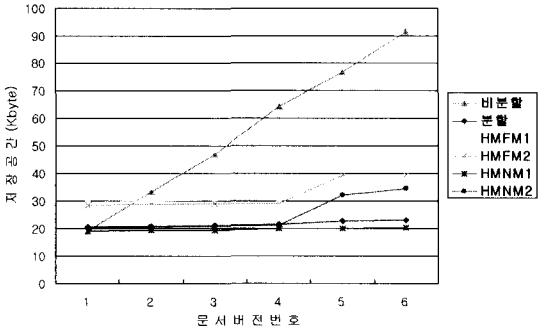
그림 16은 문서 저장 시간에 대한 성능 평가 결과이다. 성능 평가를 위한 데이터는 XML 문서 10,000개를 이용하였고, 각 문서가 6번까지 버저닝되는 경우를 측정하였다. X축 변수는 문서의 버저닝 번호로서, 1은 변경되지 않은 처음 문서이며, 2는 변경이 한번 발생한 경우를 나타낸다.

비분할 모델은 문서가 변경될 때마다 문서 전체를 저장하기 위한 시간이 매번 소요되며, 분할 모델은 처음 문서만 전체 문서를 저장하고 버저닝 2번부터는 변경된 부분만 저장하므로 저장 시간이 급격히 줄어드는 것을 볼 수 있다. 제안한 HMFM1, HMFM2는 분할 모델보다 저장 시간이 조금 더 걸리는데, 이는 마지막 버전의 전체 문서를 한번 더 저장하기 때문이다. HMNM1, HMNM2는 분할 모델과 비슷한 성능을 보이며, HMNM2에서 특히 5번 버전에서 저장 시간이 현저하게 증가하는데, 이는 5번째 버전에서 다시 비분할 저장 방식으로 문서를 저장하기 때문이다.

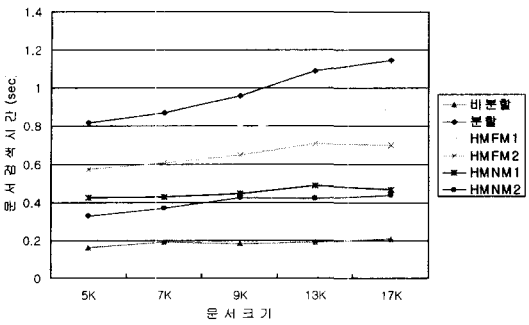
그림 17은 문서 저장 공간에 대한 성능 평가 결과이다. 성능 평가를 위한 데이터의 평균 사이즈는 10K이다. 비분할 모델에서의 저장 공간은 버저닝에 비례하여 증가하는데 비해 분할 모델에서는 증가량이 아주 적음을 볼 수 있다. HMFM1, HMFM2는 분할 모델처럼 저장 공간의 증가가 적지만, 처음문서 사이즈가 더 크게 나타나는 것을 볼 수 있는데 이는 최근 문서에 대해서 문서 전체를 따로 저장하고 있기 때문이다. 특히 HMFM2의 5번 버전에서의 증가는 5번 버전 문서를 중간 문서로서 저장하고 있기 때문이다. HMNM1, HMNM2 또한 버저닝이 되어도 저장 공간은 거의 변경이 없음을 볼 수 있는데 이는 문서 안에서 변경된 부분만을 반영하기 때문이다. 또, HMNM2에서도 5번 버전에서 저장 공간이 갑자기 늘어난 이유는 5번 버전에서 문서 전체를 비분할로 다시 저장하고 있기 때문이다.



(그림 16) 저장 시간



(그림 17) 저장 공간



(그림 18) 검색 시간

그림 18은 문서 검색 시간에 대한 성능 평가 결과이다. 문서당 평균 엘리먼트 개수는 124개이며, 버저닝때마다 변경된 평균 엘리먼트 개수는 7개로 비교적 변경이 적은 데이터를 사용하였다. 각 문서들은 버저닝이 6번까지 되어 있으며, 버전에 관계없이 임의 문서를 검색하는 방법을 사용하였다. 또한 문서 크기에 따라 검색 시간의 차가 크므로 문서를 크기별로 구분하였다. 그림에서 볼 수 있듯이 비분할 방법이 가장 검색 시간이 빠르며, 분할 방법은 가장 성능이 나쁘다. HMFM1, HMFM2 는 분할 모델 우선 중심 모델이지만 분할 모델보다 검색 시간이 적은데, 이는 최근 문서에 대해서 검색한 경우 검색 시간이 빠르기 때문이며, 특히 HMFM2에서는 중간 문서에 대해서도 검색 시간이 단축되고 있기 때문이다. HMNM1, HMNM2 는 비분할 모델과 같이 검색 시간이 적게 나타나는 것을 볼 수 있는데, 비분할 방법을 취하면서 변경된 엘리먼트에 대해서만 반영을 해주면 되기 때문이다. HMNM2가 HMNM1보다 검색시간이 빠른 이유는 5번 버전 문서에서 문서 전체를 저장하고 있기 때문이다.

지금까지 살펴본 결과 기존의 분할 방법과 비분할 방법은 한쪽 기능은 우수하지만, 다른 한쪽 성능은 현저히 떨어진다. 그러나 본 논문에서 제안한 혼합형 모델링 기법들은 저장 시간과 저장 공간이 분할 모델과 비슷한 우수한 성능을 보이면서 검색 시간 또한 우수한 성능을 보인다.

혼합형 모델링 기법들은 각기 특징을 가지고 있으므로 그 특성에 맞는 응용에 적합하다. 우선 HMFM1 데이터 모델은 항상 최근 문서를 유지하기 때문에, 주로 최근 문서에 대한 검색이 이루어지는 응용에 아주 적합하고, HMFM2 데이터 모델은 중요 문서에 대해 문서 전체를 유지한다. 그러므로 중요한 버전의 문서가 주로 검색되어지는 응용에 적합하다. HMNM1 데이터 모델은 문서의 변경이 적은 응용에 아주 적합하며, HMNM2 데이터 모델은 문서 변경이 빈번히 일어나는 응용에 적합하다.

(표 1) 성능평가 결과

	비분할모델	분할모델	HMFM1	HMFM2	HMNM1	HMNM2
저장시간 (sec)	4.737	0.972	1.305	1.34	0.923	1.668
저장공간 (K)	91K	23K	32K	40K	22K	34K
검색시간 (sec)	0.187	0.977	0.784	0.647	0.45	0.395

5. 결 론

본 논문에서는 분할 저장 방식과 비분할 저장 방식의 특성을 결합한 혼합 모델링 방법 4가지를 제안하였다. 분할 저장 방식은 문서 변경에 대한 효율적인 버저닝을 지원하지만 문서 검색 시간이 오래 걸리고, 비분할 저장 방식이 빠른 문서 검색을 지원하지만 버저닝을 위한 오버헤드가 크다는 단점을 보완하였다.

HMFM1 데이터 모델은 분할 모델의 스키마 구조를 따르고, 문서 변경시 최근문서에 대한 정보를 유지하기 위한 최근 문서 테이블이 하나 추가된다. 최근 문서 테이블에는 가장 최근 문서 하나만 저장되며, 최근 문서에 대한 검색이 이루어지면 이 테이블에서 문서를 검색한다. HMFM2 데이터 모델은 분할 모델의 스키마 구조를 따르며, 최근 문서와 중간 문서는 FULLDOC 테이블에 저장한다. 검색되어지는 문서가 최근 문서나 중간 문서인 경우 이 테이블에서 직접 검색이 가능하다.

HMNM1 데이터 모델은 비분할 모델의 스키마 구조를 가지며, 문서 변경이 이루어질 경우 엘리먼트 단위의 동적인 반영을 위해 DTD_CHANGE 테이블이 추가된다. 특정 버전에 대한 문서 검색시, 제일 처음 문서의 엘리먼트 정보가 기술된 DTD_ELE 테이블과 DTD_CHANGE 테이블을 동시에 반영하여야 한다. HMNM2 데이터 모델은 HMNM1을 따르며, 문서 변경이 여러번 이루어질 경우 다시 비분할 문서로 정보를 유지하므로 어느 버전이라도 대체로 빠른 문서 검색이 가능하다.

제안한 혼합형 모델 4가지에 대해서 각각 검색 순서도를 기술하였고, 기존의 분할 모델, 비분할 모델과 비교하여 각 모델들의 성능 평가를 실시하였다. 성능 평가 결과 저장 공간 및 저장 시간이 비분할 모델보다 상당히 우수한 성능을 보였으며, 검색 시간 또한 분할 모델보다 월등히 나은 성능을 보였다.

향후 연구로써, 버저닝시 문서 변경이 많이 이루어진 경우에 대해 추가적인 성능 평가를 실시할 필요가 있고, 버저닝 회수가 많이 발생하는 경우에 대해서도 성능 평가를 실시하는 것이다.

Acknowledgement

이 논문은 2001년도 한국학술진흥재단의 지원에 의하여 연구되었음(KRF-2001-041-E00233)

참 고 문 헌

[1] 민영수, 강승헌, 강형일, 유재수, 이하욱, 최한석, "XML 문서를 위한 구조정보 추출기의 설계 및 구현", 한국정보과학회 '99 가을 학술발표 논문집(Ⅰ), 한국정보과학회, pp, 81-83, 1999.

[2] 연제원, 조정수, 이강찬, 이규철, "XML 문서 구조검색을 위한 저장 시스템 설계", 한국정보과학회 학술 발표 논문집(B), 제26권 제1호, pp. 3-5 1999.

[3] 강형일, "효율적인 구조검색을 위한 XML 저장관리 시스템 설계", 박사학위논문, 2001. 2.

[4] Alin Deutsch, Mary Fernandez, and Dan Suciu, "Storing Semistructured Data with STORED", SIGMOD, 1999.

[5] Arijit Sengupta. "Toward the union of databases and document management : The design of DocBase," Conference on Management of Data (COMAD'98), Hyderabad, India, 1998.

[6] Ricardo Baeza-Yates and Gonzalo Navarro, "Integrating Contents and Structure in Text Retrieval", ACM SIGMOD, pp. 67-79, 1996.

[7] K. Bohm, A. Muller, and E. Neuhold, "Structured Document Handling - a Case for Integration Databases and Information Retrieval", CIKM'94, pp. 147-154, 1994.

[8] Daniela Florescu and Donald Kossmann, "Storing and Querying XML Data using an RDBMS", IEEE Data Engineering Bulletin, 22(3), pp. 27-34, 1999.

[9] Tuong Dao, Ron Sacks-Davis, and James A. Thom, "An Indexing Scheme for Structured Documents and its Implementation", Proceedings of the Fifth International Conference on Database Systems for Advanced Applications(DASFAA '97), pp. 125-134, 1997.

◐ 저 자 소 개 ◑

유 재 수

1989년 전북대학교 컴퓨터공학과 졸업(공학사)
1991년 한국과학기술원 전산학과 졸업(공학석사)
1995년 한국과학기술원 전산학과 졸업(공학박사)
1995년~1996년 8월 목포대학교 전산통계학과 전임강사
1996년 8월~현재 : 충북대학교 정보통신공학과 부교수
관심분야 : 데이터베이스 시스템, XML, 멀티미디어 데이터베이스, 분산객체 컴퓨팅, etc.
E-mail : yjs@cbucc.chungbuk.ac.kr

손 충 범

1997년 충북대학교 정보통신공학과 졸업(공학사)
1999년 충북대학교 대학원 정보통신공학과 졸업(공학석사)
1999년~현재 : 충북대학교 대학원 정보통신공학과 박사과정
관심분야 : XML, 데이터베이스 시스템, 정보검색 프로토콜, 분산 객체 컴퓨팅 분야 etc.
E-mail : cbson@trut.chungbuk.ac.kr

조 혜 영

2000년 충북대학교 컴퓨터공학과 졸업(공학사)
2002년 충북대학교 대학원 정보통신공학과 졸업(공학석사)
관심분야 : 데이터베이스 시스템, XML, EJB etc.
E-mail : sunita@korea.com