확장된 경로 표현을 이용한 XML 문서의 저장 구조 설계 및 구현

*
백주현, 최윤철
연세대학교 컴퓨터과학과 멀티미디어/그래픽스 연구실

Design and Implementation of Storage Structure in XML using Extended Path Expression

Joo Hyun Baek, Yoon-Chul Choy
Department of Computer Science, Yonsei University
E-mail : bbaek@rainbow.yonsei.ac.kr
ycchoy@rainbow.yonsei.ac.kr

요 약

최근 인터넷상에서 정보 교환의 표준으로 XML이 자리잡고 있다. XML은 웹 문서뿐만 아니라 전자 도서관, 전자 상거래, EC/EDI를 포함한 다양한 분야에서 사용하기 위하여 폭 넓은 연구를 진행하고 있다. 따라서 이들 문서에 대한 저장 및 검색에 대한 연구가 활발히 진행되고 있다. 그러나 이들 기법들은 주로 관계형 데이터베이스 시스템만 지원하거나, 주로 검색만 다루어 문서 정보 갱신이 발생할 경우 인덱스 변환에 따른 비용이 증가하는 문제점이 있다. 본 연구에서는 기존의 기법과 달리 객체 관계 및 관계형 데이터베이스 환경 둘 다 지원할 수 있도록 하였으며, 검색 정보 표현과 정보 갱신에 따른 인덱스 갱신 비용을 최소화하기 위해 확장 경로 표현 기법을 제안하였다. 그리고 제안된 기법을 Window 2000에서 PHP와 SQL Server 2000 데이터베이스 시스템을 이용해서 구현하였다.

1. 서론

최근, 인터넷의 발전으로 많은 사용자들이 기존 정보를 공유하게 되었으며, 이로 인해 XML(eXtensible Markup Language)이 데이터와 문서를 위한 표준 포맷으로 자리잡고 있다. 그 결과 많은 종류의 데이터들이 XML 문서나 XML 데이터 형태로 바뀌어 가고 있다. 이로 인해 XML 문서를 저장하고, 저장된 문서로부터 정보를 검색하기 위한 기법들이 많이 개발되고 있다. 개발된 기법은 주로 관계형 데이터베이스 시스템에서 DTD에 의존적이거나 독립적인 방법으로 스키마를 설계한 뒤 저장을 하고, 저장된 문서에 대해 질의를 하는 질의 처리에 대해 주로 연구가 진행되어 왔다[1, 2]. 그러나 최근에는 전통적인 관계형 시스템에 제약 없는 데이터 형 선언이나 *set-value*등을 지원하기 위해 객체 관계 데이터베이스 환경에서 XML 문서를 저장하고 질의 처리를 하려는 연구가 진행되고 있다[3, 7].

본 논문에서는 이런 최근의 추세를 반영하기 위해 객체 관계형 시스템 및 관계형 데이터베이스 시스템을 지원할 수 있도록 확장 경로 표현 기법을 제안하고, 이를 위해 문서 저장을 위한 테이블을 제안하였으며, 질의 처리 시의 효율적인 검색을 위해 DTD에 독립적으로 스키마를 구성하고, 저장 스키마를 분할 저장 방식을 이용하여 저장하였다. 또한 제안된 기법을 실제 데이터베이스 시스템(SQL Server 2000)에서 설계 구현하였다.

2. 관련 연구

최근까지의 저장 구조에 대한 연구는 1) 각 엘리먼트 노드들이 자신의 고유번호를 갖고 상위 노드가 하위 노드에 대해 고유 번호를 리스트 형태로 유지하는 방법이다. 이 방법은 트리 형태의 특정 노드나 그 노드의 서브 트리에 대한 검색 시 트리 탐색을 해야하므로 많은 탐색 시간이 소요된다는 단점이 존재한다[4]. 2) 자신이 자식 엘리먼트 중 몇 번째 엘리먼트인가를 나타내는 엘리먼트 ID를 자신의 부모로부터 계승받아 사용하는 EID 기법은, 저장 정보 내에 순환 구조가 존재하는 경우에는 경로 엘리먼트 ID가 무한히 증가한다는 문제점이 발생한다[5]. 3) 파싱된 엘리먼트 트리의 루트 노드로부터 DFS(*Depth First Search*)방식으로 노드를 방문하여 처음 방문할 때의 순서와 자신의 자식 노드의 방문

이 끝난 뒤의 노드의 방문 순서의 한 쌍으로 구조 정보를 표현하는 방법이다. 이 기법의 경우 자식 노드의 수가 상당히 많은 경우에는 검색 시간이 증가하며, 문서 정보에 대한 내용을 추가하거나 삭제 할 경우는 다시 DFS Numbering을 해야하는 단점이 있다[4]. 4) 경로 정보와 위치 정보를 이용하는 방법으로 이 방법은 순환 구조가 여러 개의 루트를 가지고 있을 경우에는 경로 표현이 불가능하다는 문제점이 존재한다[6].

3. 저장 구조

본 연구에서 저장 구조 설계를 위해 고려 시 저장 공간 문제는 고려하지 않았으며 문서 정보의 저장, 추가/삭제, 검색시의 효율성만을 고려하였다. XML 문서 저장 구조는 다음과 같이 세 개의 모듈로 구성된다. 첫째는 저장 관리기로 XML 파서를 포함하고 있는 모듈로 XML 문서 저장 요구가 발생하면 해당 문서를 파서를 이용하여 파싱한 후 DOM 트리 형태로 변환하여 해당 객체를 데이터베이스 테이블에 매핑하는 모듈이다. 두 번째는 질의 생성기로 사용자가 인터페이스를 통해 질의를 하면 질의의 패턴을 분류한 뒤 SQL 형태의 질의어로 변환하는 모듈이다. 세 번째는 질의 변환기로 Xpath 형태의 질의가 들어오면 XML 저장 구조에 맞는 SQL로 변환하는 모듈이다. 이때 저장을 위한 기본적인 구조는 [3, 6]의 경로 표현 기법을 보완한 확장 경로 표현을 제안하였다. 확장 경로 표현은 다른 저장 기법들이 가지고 있던 문제점을 보완 개선한 것으로 참조 관계를 표현하기 위해서 Path Table에 Flag field(Type)을 제안하였으며, 문서 정보를 저장할 경우 Region 값을 할당하는데 [3]과는 달리 Element, Text에 모두 정수 값을 배정하고(DFS와 유사), Text 노드도 시작 값과 끝 값을 다르게 할당한다(관계형인 경우). Region Type은 내포관계를 유지하기 위해서 제안되었으며, 노드가 추가될 때 추가될 위치의 양 쪽 Region 값의 중간 값을 할당함으로써 별도의 오버헤드가 발생하지 않도록 하였다.

또한 XML 문서의 내용과 DTD 정보를 분리해서 DTD에 독립적으로 저장하는 방법을 선택한 이유는 의존적인 방법과는 다르게 테이블에 대한 속성과 테이블 수가 고정될 수 있으므로 질의 검색 형태에 따라 질의 변환 및 정형화가 쉽기 때문이다. 본 연구에서는 문서 저장을 위한 스키마의 구성을 크게 두 부분으로 나누었는데 저장 부분과 인스턴스 부분이며, 6개의 테이블로 나누어 구성하였다. 먼저 저장 부분은 DTD, Path, Document 테이블로 구성되며, 인스턴스 부분은 element, text, attribute 테이블로 구성된다. 스키마를 구성하는 테이블은 다음과 같다.

- DTD_Table

DTD			
DTD ID	DTD Name	DTD size	DTD Cont

- Path_Table

Path			
DTD ID	Path id	path	type

- Attribute_Table

Attribute			
path ID	Doc ID	value	pos

- Document_Table

Document		
DTD ID	Doc ID	Docname

- Element_Table

Element				
Path ID	Doc ID	order	reorder	pos

- Text_Table

Text			
Path ID	Doc ID	value	pos

본 연구에서 사용된 DTD와 XML 문서는 그림 1, 2와 같다.

```
<!ELEMENT books (book*)>
<!ELEMENT book (title, editor, author, summary)>
<!ATTLIST  book style CDATA "textbook">
<!ELEMENT title (#PCDATA)>
<!ELEMENT editor (family, given)*>
<!ELEMENT author (family, given)*>
<!ELEMENT summary (#PCDATA | keyword)*>
<!ELEMENT family (#PCDATA)>
<!ELEMENT given (#PCDATA)>
```

[그림 1] book DTD

위에 정의된 XML 문서의 저장은 다음과 같은 과정을 통해 저장한다. 먼저 XML 문서를 등록한 후 문서를 DOM 파서에 의해 DOM 트리 형태로 표현한다. 그 후 이 트리 형태를 본 연구에서 정의한 저장 관리기를 통해 테이블로 매핑시켜 저장한다.

저장 관리기는 확장 경로 표현을 이용하여 노드의 정보를 해당 테이블에 저장한다. 노드의 타입이 Element이면 Element 테이블에 path ID와 Doc ID, position 등의 정보가 저장되며, Attribute가 있는지

를 검사하여 Attribute 테이블에 Attribute 값과 path ID와 position 값을 저장한다.

```
<books>
<book style="textbook">
    <title>Designing XML applications</title>
    <editor>
        <family>Bob</family><given>Kraft</given>
    </editor>
    <author>
        <family>Nick</family><given>Marcus</given>
        <family>Bob</family><given>Pant</given>
    </author>
    <summary>
    This book is the guide to design <keyword>XML
        </keyword>applications.
    </summary>
</book>
</books>
```

[그림 2] XML 문서

이런 과정의 반복을 통해 모든 정보를 데이터베이스에 저장한다. 저장 관리기를 통해 생성된 테이블은 그림 3과 같다.

• DTD

DTD ID	DTD Name	DTD Size	DTD Cont
1	smaple.dtd	314	<!ELEMENT..>

• Path

DTD ID	Path id	path	type
1	1	/books	0
1	2	/books/book	0
1	3	/books/book/@style	0
1	4	/books/book/title	0
1	5	/books/book/editor	0
1	6	/books/book/editor/family	0
1	7	/books/book/editor/given	0
1	8	/books/book/author	0
1	9	/books/book/author/family	0
1	10	/books/book/author/given	0
1	11	/books/book/summary	0
1	12	/books/book/summary/keyword	0

• Text

Path ID	Doc ID	value	pos
4	1	Designing XML ...	4, 5
6	1	Bob	9, 10
7	1	Kraft	13, 14
9	1	Nick	19, 20
10	1	Marcus	23, 24
9	1	Bob	27, 28
10	1	Pant	31, 32
11	1	This book is ...	36, 37
12	1	XML	39, 40
11	1	applications.	42, 43

• Attribute

path ID	Doc ID	value	Pos
1	1	textbook	2, 2

• Document

DTD ID	Doc ID	Doc Name
1	1	sample.xml

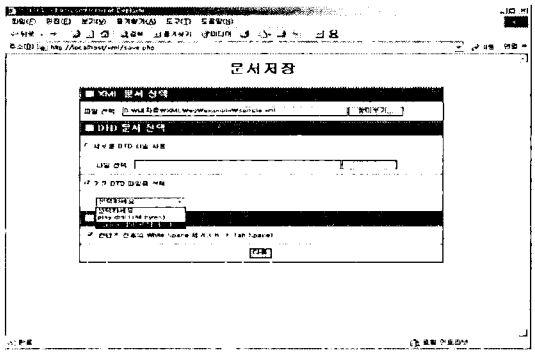
• Element

Path ID	Doc ID	order	reorder	pos
1	1	0	-1	1, 46
2	1	0	-1	2, 45
4	1	0	-1	3, 6
5	1	0	-1	7, 16
6	1	0	-1	8, 11
7	1	0	-1	12, 15
8	1	0	-1	17, 34
9	1	0	-2	18, 21
10	1	0	-2	22, 25
9	1	1	-1	26, 29
10	1	1	-1	30, 33
11	1	0	-1	35, 44
12	1	0	-1	38, 41

[그림 3] 저장된 테이블

4. 구현

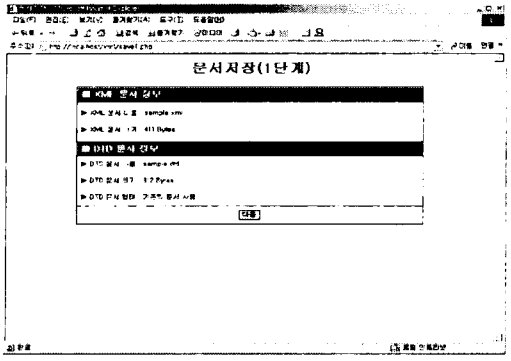
본 연구에서는 제안된 저장 구조를 실제 환경에서 구현하기 위해 데이터베이스 시스템으로는 SQL server2000을, 운영체제 환경은 Window2000으로, 그리고 PHP로 구현하였으며 DOM을 통해 파서를 구성하여 파싱하였다. 구현된 내용은 그림 4, 5, 6, 7에서 보여주고 있다. 구현 시에 사용한 예제는 그림 1, 2를 이용하였다.



[그림 4] 문서 선택 단계

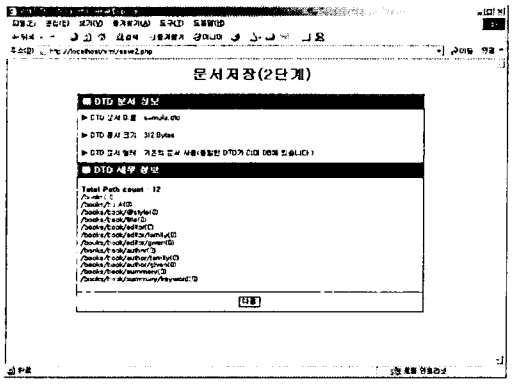
저장 단계를 살펴보면 그림 4는 문서 선택 단계로 저장할 문서와 DTD를 선택한다. 기존 DTD 파일 선택 부분은 사전에 DTD 정보가 저장되어 있으면 테이블에서 정보를 가져와서 보여준다. White

Space 제거 옵션은 Element Tag와 Contents 사이의 공백이나 개행 문자를 제거하며, "다음" 버튼을 누르면 선택한 파일을 서버에 업로드하게 된다.



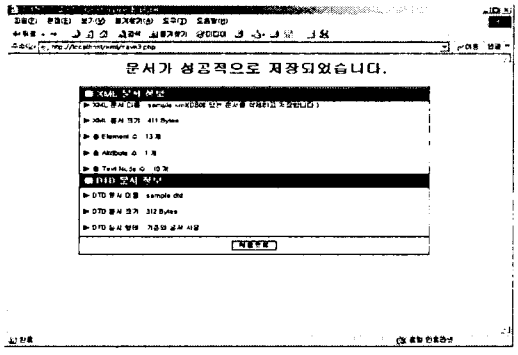
[그림 5] 문서저장(1단계)

그림 5는 파일 정보 표시 단계로서 XML 문서와 DTD 문서의 계략적인 정보를 표시한다.



[그림 6] 문서저장(2단계)

그림 6은 DTD 문서의 정보를 본 연구에서 정의한 6개의 테이블로 매핑하는 과정으로 여기서는 매핑을 통해 제시된 엘리먼트 테이블, 애트리뷰트, path 정보 등을 보여준다.



[그림 7] 문서저장 완료

그림 7은 매핑된 정보를 통해 변환된 DOM 객체를 깊이 우선 탐색으로 순환하면서 정의된 6개 테이블의 모든 정보를 나타낸다.

5. 결론 및 향후 계획

본 연구에서는 XML 문서를 데이터베이스 시스템에 저장하고 검색할 수 있도록 시스템을 설계 구현하였다. 이를 위해 문서 저장 시 분할 저장 방식을 이용하고, 기존 검색 표현 방법인 DFS Numbering이나 EID, 경로 표현 기법의 문제점을 보완하였다. 즉, DTD에 독립적으로 스키마를 구성하고 저장 구조를 위해 확장 경로 표현 기법을 제안하였으며, 순환 정보와 set-value의 표현으로 객체 관계 데이터베이스에서도 사용이 가능하도록 하였다.

앞으로는 실제 질의 변환에 필요한 질의 처리기의 설계가 필요하고 이에 따른 변환 과정 및 구현이 필요하다. 마지막으로 순환 정보를 표현 시 루트 노드가 여러 개일 경우에 대한 연구가 필요하다.

6. 참고문헌

[1] Stefano Ceri, Piero Franternali, "XML: Current Developments and Future Challenges for the Database Community", EBDT, p.2-7, 2000

[2] J. Shamugasundaram, et. all, "Relational Databases for Query XML Documents: Limitations and Opportunities", VLDB, p.302-314, 1999

[3] Takeyuki Shimura, et. all, "Storage and Retrieval of XML Documents Using Object-Relational Databases", DEXA99, p.206-217, 1999

[4] 이용석, 손기락, "XML 문서 저장 시스템의 설계 및 구현", 한국정보과학회, '98 가을 학술 발표논문집, 제25권 2호, p.347-349, 1998

[5] 연제원, 조정수, 이강찬, 이규철, "XML 문서 구조 검색을 위한 저장 시스템 설계", 한국정보과학회, '99 봄 학술 발표논문집, 제26권 1호, p.3-5, 1999

[6] 김정은, 신판섭, 이재호, 임해철, "DTD 문서를 위한 DTD 독립적인 데이터 모델 설계", 한국정보과학회, '00 봄 학술 발표논문집, 제27권 1호, p.69-71, 2000

[7] 김훈, 홍의경, "객체 관계 데이터베이스를 이용한 XML 문서 저장 모델 설계", 한국정보과학회, '00 가을 학술 발표논문집, 제27권 2호, p.225-277, 2000