

# 저장과 색인의 효율성을 고려한 정적 테이블 기반의 XML 문서 저장 시스템 설계

홍석건<sup>o</sup>, 김정화, 곽호영  
 제주대학교 컴퓨터공학과  
 {hancem<sup>o</sup>, carina, kwak}@cheju.ac.kr

The design of XML document repository based on Static Table  
 considering the efficiency of storing and indexing

Seok-Gun Hong<sup>o</sup>, Jeong-Hee Kim, Ho-Young Kwak  
 Dept. of Computer Engineering, Cheju National University

## 요약

본 논문에서는 XML 문서를 관계형 데이터베이스에 저장하는 XML 문서 저장 시스템을 제안한다. 제안 시스템은 XML 문서 구조인 DTD(Document Type Definition)를 보완한 XML Schema를 XML 문서의 기반 구조로 사용하며, 또한 XML 문서의 저장과 색인의 효율성을 위해 XML 문서에 대한 저장 구조는 정적 테이블과 필드들을 가지는 데이터베이스를 생성하여 처리하도록 하여, 제안 시스템을 이용한 애플리케이션에서 활용하기 용이하도록 하였다.

## 1. 서론

최근 인터넷 사용과 정보의 양이 급증하면서 인터넷상의 정보를 보다 효과적으로 사용하고자 하는 연구가 활발히 진행되고 있다. 한편 웹에서 가장 많이 사용되고 있는 HTML은 문서의 구조보다는 표현에 중점을 두고 있어 특정 응용분야의 구조를 표현하는 문서로는 기능이 부족하다는 단점이 있다[1,4]. 이에 웹 발전의 주도적인 역할을 하고 있는 W3C(World Wide Web Consortium)에서는 차세대 웹 문서의 표준으로 XML(Extensible Markup Language)이라는 전자문서 메타언어를 1996년 제안하였으며 현재까지 그 기능이 계속 확장되고 있는 상태이다[2].

XML 문서들의 구성요소를 제한하는 기술로는 DTD(Document Type Definitions)와 XML Schema 두 가지가 있다. DTD는 초기 XML 권고안에 포함된 기술로 선언 규칙을 통해 문서의 구조를 정의하는 방식이다. 반면 XML Schema는 요소 템플릿을 이용하여 문서의 구조를 정의하는 방식으로 XML Schema 자체도 XML 문법을 사용하며 DTD의 여러 단점을 보완하여 만들어졌으며 객체 지향 개념을 도입하여 강력하게 문서의 구조를 정의할 수 있게 되었다.[2,3,5,6]

본 논문에서는 효율적인 XML 문서의 저장과 색인을 위해 연구가 진행되는 상태에서 기존 연구의 기초인 DTD 문서 구조를 보완한 XML Schema 기반의 XML 인스턴스들을 관계형 데이터베이스에 저장하는 방법을 제안한다. 또한 저장과 색인의 효율성을 고려하여 관계형 데이터베이스내의 정적 테이블 구조는 XML 인스턴스를 저장하기 위한 인스턴스와는 별개로 일반적인 XML 기본 구성 요소에 따라 생성되도록 하였다. 이는 기존 DTD를 적용한 저장 시스템은 검색을 위한 문서의 구조를 중심으로 한 저장 구조를 가지고 있기 때문에 문서를 관리하는 측면에는 매우 취약하기 때문이다.

본 논문의 구성은 먼저, 2장에서는 제안한 시스템에

서 사용할 XML Schema와 XML 인스턴스를 저장할 수 있는 기본적인 저장 구조를 제시하고, 3장에서는 제시한 저장 구조를 적용한 저장 시스템을 기술한다. 4장에서는 기존 DTD를 적용한 시스템과 제안 시스템과의 비교분석하며 마지막으로 5장에서 결론과 향후 연구를 기술한다.

## 2. XML Schema와 XML 인스턴스의 저장 구조

XML Schema와 XML 인스턴스는 XML의 기본 문법이 적용되므로 기본적인 구조는 같다. 따라서 ID, SID, 계층 정보, Element, Attribute, Entity, Namespace를 구성요소로 하였다.

문서 저장구조는 그림1에 나타내었으며, ID는 엘리먼트들을 구분하기 위한 식별자이며, SID는 XML Schema를 저장할 때에는 스키마간 구분을 위한 구분자로 XML 인스턴스를 저장할 때에는 적용되는 스키마를 구분하기 위해 사용되며 계층정보는 상위 엘리먼트를 가르키는 ref, 엘리먼트의 등급을 저장하는 ref\_level, 그리고 엘리먼트의 순서를 저장하는 ref\_step으로 구성되며 Element는 엘리먼트명, Attribute는 엘리먼트의 속성과 그에 해당하는 값들을 가지며, Entity는 엘리먼트가 가지는 값을, 그리고 네임스페이스는 접두사를 갖는 nsName과 URI를 저장하는 URI로 구성되도록 하였다.

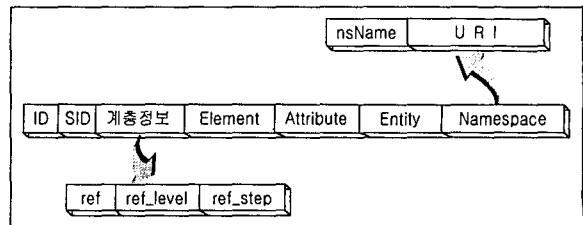


그림 1. XML Schema와 인스턴스 저장 구조

그림1의 저장 구조를 이용하면 XML Schema정보와 XML 인스턴스를 동시에 저장하는 것이 가능하다. 하지만 위의 구조를 적용하면 단일 XML Schema에 대한 XML 문서만 저장되기 때문에 이를 보완하기 위해 XML Schema 저장 테이블에는 저장된 각각의 Schema마다 일련번호를 부여하고 이에 따라 XML 인스턴스 저장 테이블에도 어떤 Schema가 적용되는지를 표시하는 필드를 추가하여 적용되는 XML Schema의 일련번호를 저장하도록 하였다.

제안한 구조를 모델링하기 위한 예제 XML Schema는 그림 2와 같으며, 이는 표 1과 같은 형태로 저장된다.

```
<?xml version="1.0" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Customer">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="FirstName" type="xs:string" />
        <xs:element name="MiddleInitial" type="xs:string" />
        <xs:element name="LastName" type="xs:string" />
      </xs:sequence>
      <xs:attribute name="customerID" type="integer" />
    </xs:complexType>
  </xs:element>
</xs:schema>
```

그림 2. 예제 XML Schema 문서

표 1 XML Schema의 저장내용

1/1	1/0/0	xml	version="1.0"		
2/1	2/0/0	schema		xs / http://.....	
3/1	2/1/1	element	name="Customer"	xs / http://.....	
4/1	2/2/2	complexType		xs / http://.....	
5/1	2/3/3	sequence		xs / http://.....	
6/1	2/4/4	element	name="FirstName" type="xs:string"	xs / http://.....	
7/1	2/5/4	element	name="MiddleInitial" type="xs:string"	xs / http://.....	
8/1	2/6/4	element	name="LastName" type="xs:string"	xs / http://.....	
9/1	2/7/3	attribute	name="customerID" type="xs:integer"	xs / http://.....	

그리고 XML 인스턴스도 XML Schema의 저장 구조와 같은 형태이므로 그림 2의 XML Schema가 적용된 예제 XML 인스턴스가 그림 3처럼 있고 이를 저장 구조에 저장을 하면 표 2와 같은 형태로 저장이 된다.

```
<?xml version="1.0" ?>
<Customer customerID="24332">
  <FirstName>Raymond</FirstName>
  <MiddleInitial>G</MiddleInitial>
  <LastName>Bayliss</LastName>
</Customer>
```

그림 3. 예제 XML 인스턴스

표 2 XML 문서의 저장 내용

1/1	1/0/0	xml	version="1.0"		
2/1	2/0/0	Customer	customerID="24332"		
3/1	2/1/1	FirstName		Raymond	
4/1	2/2/1	MiddleInitial		G	
5/1	2/3/1	LastName		Bayliss	

### 3. XML 문서 저장 시스템 구조

본 논문에서 제안하는 시스템은 그림 4와 같이 크게 Interface Class, Module Class, Analyzer Class, Storage Class 4부분으로 구성된다.

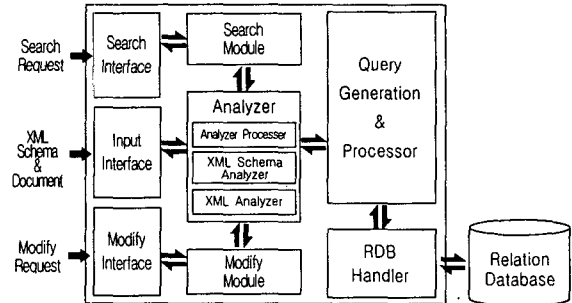


그림 4 제안 시스템 구성도

#### 3.1 Interface Class

Interface Class는 외부 요청에 따른 데이터를 입력 받고 그에 따른 결과를 보여주는 부분으로서 본 논문에서 제안한 시스템에서는 크게 Search, Input, Modify Interface로 구분하였다.

Search Interface는 저장된 XML 인스턴스 내에서 사용자가 필요로 하는 데이터를 검색 및 출력하기 위한 Interface로서 검색에 필요한 질의 정보를 사용자로부터 입력받아 Search Module로 전달하며, 전달된 정보에 따른 반환 결과를 사용자에게 보여준다.

Input Interface는 저장할 XML 인스턴스와 XML Schema를 사용자로부터 입력 받아서 관계형 데이터베이스에 저장하기 위한 Interface이다.

Modify Interface는 저장된 XML 인스턴스 또는 XML Schema를 수정 및 삭제하기 위한 정보를 사용자로부터 입력받아 Modify Module로 전달하여 명령의 수행 성공 여부를 사용자에게 보여준다.

#### 3.2 Analyzer Class

Analyzer Class는 Input Interface Class나 Module Class에서 들어온 요청에 따라 이를 분석하여 Query Generation & Processor에서 처리할 수 있는 정보를 생성하는 역할을 하게 되며 Analyzer Processor, XML Schema Analyzer, XML Analyzer로 구성된다.

작업 요청이 들어오면 Analyzer Processor에서는 XML Schema Analyzer와 XML Analyzer를 이용하여 작업을 수행하기 위한 정보를 생성하며 이를 Query Generation & Processor에 전달하여 작업을 수행한다.

XML Schema Analyzer와 XML Analyzer는 요청한 작업에 해당하는 XML Schema와 XML 인스턴스에서 작업에 필요한 정보를 얻어오는 역할을 한다.

#### 3.3 Module Class

Module Class는 Search와 Modify Module를 말하며 필요한 연산을 Analyzer부와 연계하여 작업을 수행한다. Analyzer부에서 처리할 수 없는 연산을 각 연산에

속하는 Module로 만들어서 Analyzer에서 처리할 수 있는 정보로 변환하여 처리하도록 하였다.

### 3.4 Storage Class

Storage Class는 Query Generation & Processor 과 RDB Handler로 구성되며, 관계형 데이터베이스에 저장하기 위한 Query를 생성하고 이를 처리하는 역할을 한다. Query Generation & Processor부분은 Analyzer부분에서 입력받은 정보를 이용해 관계형 데이터베이스를 이용하기 위한 표준 질의를 생성하는 부분이며 RDB Handler 부분은 Query Generation & Processor에서 생성된 Query를 관계형 데이터베이스에 적용하는 부분이다.

### 3.5. 시스템 처리과정

본 논문에서 제안한 저장 시스템에서 중심이 되는 부분은 Analyzer Class이다. Interface Class나 Module Class에서 요청이 들어오면 Analyzer Class에서는 요청을 분석하여 Query Generation & Processor에 전달한다. Query Generation & Processor에서는 이를 이용해 질의어 생성을 하며 RDB Handler에 질의를 넘겨주며 RDB Handler는 관계형 데이터베이스에 질의를 수행하게 된다. 질의를 수행한 후 결과는 역순으로 사용자에게 전달된다. 처리과정은 그림 5와 같다.

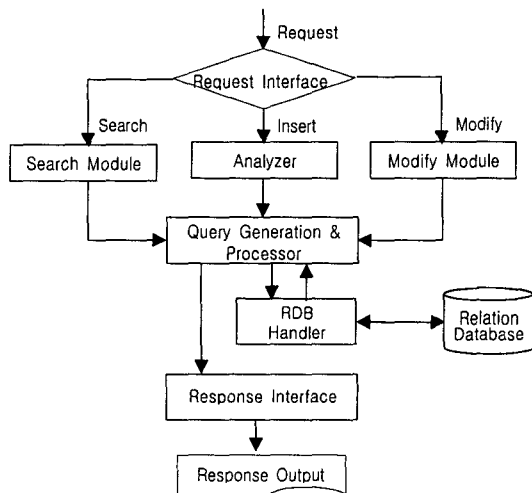


그림 5 시스템 처리과정

### 4. DTD를 적용한 시스템과의 비교

기본 문서 구조로 DTD를 적용한 저장 시스템은 검색 중심의 설계방식을 적용하였으며 별도의 공간에 색인 구조를 생성해야 한다. 따라서, 색인 구조를 저장하기 위한 추가 저장 공간이 필요하며 새로운 인스턴스를 추가로 저장할 때는 색인 구조를 재생성해야 한다. 또한 저장된 문서나 DTD를 변경하는 경우나 저장된 DTD를 적용한 문서의 유효성 검사를 하는 경우 추가연산에 많은 시간이 소요된다. 본 논문에서 제안한 저장 시스템과 DTD를

적용한 시스템과의 비교는 표 4와 같다.

표 4 시스템 비교

구분	DTD 적용 시스템	제안 시스템
주저장내용	문서 구조 중심	문서 내용 중심
색인구조여부	존재함	존재하지 않음
문서 저장 목적	검색	문서 관리
저장구조 복잡도	DTD와 문서의 저장 구조가 다르며 검색 위주의 저장방식이기 때문에 구조 정보의 저장비율이 높다.	동일 문법을 사용하는 XML Schema를 적용해서 문서와 동일 구조에 저장 가능하며 저장 시 문서의 손상을 최소화한다.
유효성 검사	문서의 구조와 간단한 값 검사만이 가능함	문서의 구조는 물론 값의 형태와 범위 등 강력한 검사가 가능함.

### 5. 결 론

본 논문에서는 DTD의 단점을 보완한 XML Schema를 적용하여 XML 인스턴스를 관계형 데이터베이스에 저장할 수 있는 구조와 이를 이용한 저장 시스템을 설계하였다.

제안한 저장 시스템은 정적 테이블의 사용으로 저장 공간 활용이 효율적이며 XML 인스턴스나 XML Schema의 손상을 최소한으로 유지하면서 원 문서를 최소한의 손상으로 저장하기 때문에 Schema를 적용한 유효성 검사나 인스턴스를 변환할 때 프로그램 연산시간을 최소한으로 하여 수행할 수 있도록 설계하였다. 또한 정적 테이블의 사용은 저장 구조를 최대한 간단하고 고정된 형태를 유지할 수 있도록 하게 함으로써 이를 이용한 애플리케이션을 개발 시 개발 기간을 단축시킬 수 있을 것이다.

하지만 제안한 저장구조를 적용하면 XML Schema의 모든 기능을 사용하기 어렵다는 단점이 발생하므로 향후 연구는 제안한 구조를 바탕으로 하여 XML Schema 저장 구조에 대한 추가 연구가 필요할 것으로 생각된다.

### 참 고 문 헌

- [1] 연제원, 조정수, 이강찬, 이규철, "XML 문서 구조검색을 위한 저장 시스템 설계", 한국정보과학회 학술발표 논문집(B), 제26권 제1호, pp.3-5 1999.
- [2] Extensible Markup Language (XML) 1.0 (Second Edition), "http://www.w3.org/TR/REC-xml"
- [3] XML Schema, "http://www.w3.org/XML/Schema"
- [4] D. W. Shin, ... "BUS:An Effective Indexing and Retrieval Schema in Structured Documents", in Proc. Digital Libraries, 1998
- [5] Brian Lowe, Justin Zobel, Ron Sacks-Davis, "A Formal Model for Databases of Structured Text", Proceedings of the Fourth International Conference on Database Systems for Advanced Applications (DASFAA '95), pp.449-456, 1995.
- [6] Toung Dao, "An Indexing Model for Structured Documents to Support Queries on Content, Structured and Attributes", Proceedings of ADL '98, pp.88-97, 1998.
- [7] 박종관 외, "XML 문서의 효율적인 구조 검색을 위한 색인 모델", 한국정보처리학회 논문지, 제8-D권, p451-460, 2001.