XML Schema 지원도구 설계 및 구현

나종연*, 오정진**, 최한석**
목포대학교 멀티미디어공학과
nakka@apollo.mokpo.ac.kr , jjoh@mail.chunnam-c.ac.kr,
chs@kware21.com

Design and Implementation of XML Schema Supporting Tool

Jong-Youn Na,Jeong-Jin Oh, Han-Suck Choi
Dept. of Multimedia Engineering, Mokpo University

요약

XML이 표준화 제정될 당시 XML문서의 구조를 정의하기 위해 DTD를 사용하였다. DTD는 XML의 전신인 SGML에서부터 사용되어 왔으며 여러 훌륭한 기능을 지원하고있다. 그러나 DTD는 그 고유의 형식으로 이루어져 있어 XML 개발자는 XML구문분석기 이외에 DTD 구문분석기를 별도로 개발하여야하고, DTD는 지원하는 데이터 형식에 제한이 있으며, 데이터 값의 범위 등을 설정할 수 없어 XML 문서의 유효성 검사에 제한이 많다. 이러한 문제를 해결하기 위하여 W3C에서는 XMLSchema를 제정하였다. DTD의 구문을 XML Schema로 모델링하였고, XML 문서에서 XML스키마를 적용하기 위한 도구와 DTD의 XMLSchema변환을 효과적으로 생성할 수 있는 GUI기반 XMLSchema Tool을 설계, 구현하였다.

1. 서론

XML Schema는 XML 객체의 속성과 요소간 상호 관계의 추상적 표현으로 문서에서 스키마를 나타내기 위해 구조 분석과 각 구조 요소를 정의한다. DTD는 XML문서의 구조를 정의할 수는 있지만, XML 문서를 구성하는 각 엘리먼트나 어트리뷰트들에 대한 데이터 타입을 정의할 수 있는 방법을 제공하지 않는다는 점이다. XML Schema의 경우 이러한 DTD의 한계를 극복할 수 있도록 XML 엘리먼트에 대한 데이터 타입을 정의하여 사용할 수 있도록 하고 있다. 현재 XML Schema Working Group에 의해서 XML Schema에 대한 표준 스펙이 정의되었고, 이미 몇몇 업체에서는 Schema 지원을 구체화시키고 있다. 문서형 정의(DTD)나 단순 객체 XML(SOX: Simple Object XML)과 같은 언어보다 다소 유리하고 XML로 되어 있기 때문에 파서(parser)에 의한 매개 처리 없이 직접 처리되며, 자기 기록, 자동 스키마 생성, XSL 변환(XSLT)을 통한 정의 능력 등 이점이 있다. 그러므로 본 논문에서는 XML스키마를 쉽게 작성할 수 있도록하고 XML문서에서에 읽기/쓰기를 WYSWYG하게 적용할 수 있도록 XML스키마의 분석과 XML문서의 적용을 설계하였다. 본 논문에서 XML스키마의 정규형을 규정하고 DTD와의 관계성을 분석하여 DTD를 XML스키마로 변환 표현하고자 한다.

2. 이론적 배경

2.1 DTD

DTD는 자체 고유의 문법 체계를 통해서 XML 문서의 구조를 정의하기 위한 일종의 언어(language)이다

1. 엘리먼트(Elements)

XML문서에서 요소를 선언할 수 있는 엘리먼트(Element)의 선언 형식은 <표 1>과 같다.

<!	ELEMENT	이름	내용	지시자	>
<!	ELEMENT	제품	(제품이름,제품단가)		>

<표 1> 엘리먼트 선언

2. 어트리뷰트(Attributes)

어트리뷰트는 엘리먼트가 가지고 있는 정보나 속성을 정의할 뿐만 아니라 엘리먼트가 가질 수 있는 어트리뷰트에 대한 제약 조건들도 명시해 줄 수 있다.

3. 엔터티(Entity, Entities)

문서내에서 그리스 문자, 수학 공식기호 등 다양한 문

<	ATTLIST	엘리먼트 이름	어트리뷰트 이름	타입 및 키워드	디폴트 값	>
<	ATTLIST	핸드폰	기종	CDATA	#REQUIRED	>

<표 2> 어트리뷰트를 선언하는 형식

자와 그래픽, 이미지, 표, 수식, 도표, 사진과 같은 정보를 포함할 수 있다.

<	ENTITY	name	SYSTEM 또는 PUBLIC 키워드	외부 식별자	>
<	ENTITY	1장	SYSTEM	"doc\chap01.xml"	>

<표 3> 엔티티 선언 방법

4. 노테이션(Notations, 표기법)

외부의 파싱되지 않는 엔터티들을 선언하기 위해서 사용된다. 노테이션 선언 방법은 엔터티를 선언하는 방법과 동일하다.

<	ENTITY	name	SYSTEM 또는 PUBLIC 키워드	외부 식별자	키워드	명칭	>
<	ENTITY	수량	SYSTEM	"doc\제품.xml"	NDATA	CHAP	>

<표 4> 노테이션을 포함하고 있는 엔티티 선언방식

2.2 DOM

플랫폼 과 언어에 중립적인 인터페이스로서, 프로그램과 스크립트가 동적으로 문서의 내용과 구조, 스타일에 접근하고 갱신할 수 있도록 해준다. 문서는 횡단하고 수정할 수 있는 트리구조로 재표현될 수 있다 DOM은 문서를 노드 객체들의 계층구조로서 나타낸다. 문서의 구조에 있어서, 어떤 타입의 노드는 여러 가지 종류의 자식노드들을 가질 수 있고, 다른 노드는 하위에 아무것도 가지고 있지 않은 단말 노드들이 될 수 있다. 자식 노드로서 가질 수 있는 노드타입은 <표 5>와 같다.

2.3 XML Schema

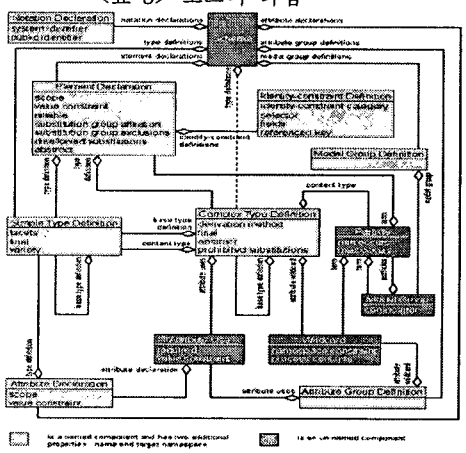
DTD는 시간이 지나면서 몇 가지 문제점을 가지게 되었는데, 첫 번째로 DTD가 고유한 문서구조 형식으로 이루어져 있어 XML 개발자는 XML 구문 분석기 이외에 DTD 구분 분석기를 별도로 개발해야 했다. 이러한 문제점을 보완하고 대체할 목적으로 스키마(schema)가 나오게 되었다.

가. XML 스키마 구조

XML 스키마를 구조적인 면에서 살펴보았을 때는 12개의 스키마 컴포넌트(schema component) 블록을 통해서 이루어진다.

노드타입	단말노드
문서	엘리먼트(maximum of one), 처리명령, 주석, 문서타입
문서조각	엘리먼트, 처리명령, 주석, 텍스트 CDATA섹션, 엔터티참조
문서타입	자식없음
엔터티참조	엘리먼트, 처리명령, 주석, 텍스트 CDATA섹션, 엔터티참조
엘리먼트	엘리먼트, 텍스트, 처리명령, CDATA섹션 엔터티참조
속성	텍스트, 엔터티참조
처리명령	자식없음
주석	자식없음
텍스트	자식없음
CDATA섹션	자식없음
엔터티	엘리먼트, 처리명령, 주석, 텍스트 CDATA섹션, 엔터티참조
• Notation(표기)	자식없음

<표 5> 노드의 타입



<그림 1> XMLSchema 컴포넌트 데이터 모델

나. 엘리먼트 선언

XML 스키마에서 엘리먼트는 <element>라는 엘리먼트를 이용해 선언된다. 또한 타입에 따라서 simpleType과 ComplexType으로 나뉜다. <표 6>은 minOccurs와 maxOccurs의 특징을 요약한 것이다.

	디폴트 값	값의 범위	의미
minOccurs	1	양수	0 : 0번이상 출현
maxOccurs	1	양수	Unbound : 무한대로 출현

<표 6> minOccurs와 maxOccurs의 특징

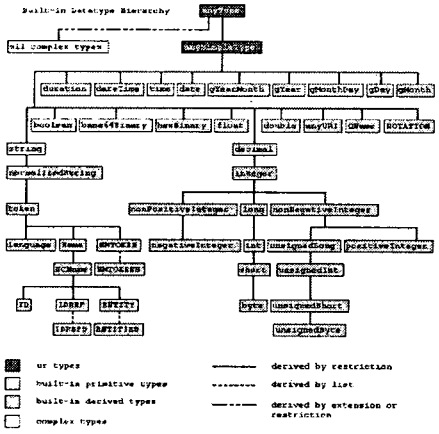
다. 어트리뷰트선언

어트리뷰트를 선언하는 방법은 엘리먼트를 선언할 때의 방법과 비슷하다. 그러나 어트리뷰트의 출현 횟수를

선언하는 방법은 약간 다른데, 그 이유는 어트리뷰트의 디폴트 출현 횟수는 선택적으로(optional)하게 명시되어야 하기 때문이다.

라. 타입 정의

XML 스키마는 Complex 타입과 Simple타입이라는 두 개의 타입 정의를 가지고 있다. 다음은 XML 스키마의 내장된 데이터타입 (Built-in datatypes)들을 그림으로 나타낸 것이다. 내장된 데이터 타입은 크게 <그림 2> 처럼 두 가지로 나누어 진다.

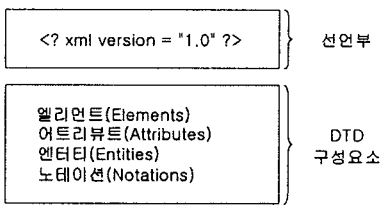


<그림 2> XML 스키마의 내장된 데이터타입

3. DTD XMLSchema 모델링

3.1 DTD 구조

DTD 문서는 크게 두 부분으로 구성되어 있다.



<그림 3> DTD 구조

3.2 DTD 구성요소 XMLSchema 모델링

XML 스키마는 모듈화(modularity)를 제공한다. XML 스키마에서는 구조를 그룹화해 줄 수 가 있는데, 이런 그룹화를 통해 어트리뷰트나 모델을 모델링 할 때 그룹으로 분류해서 모델링 할 수가 있게 된다.

가. PI 선언

PI(Processing Instructions)으로 응용들에 정보를 제공하기 위해서 사용한다. <그림4>는 DTD에서의 PI를 <그림5>와같이 XML 스키마 PI로 변환한것이다.

```
<?xml version="1.0" encoding="euc-kr"?>
```

<그림 4> DTD의 PI

```
<xml version="1.0" encoding="euc-kr"?>
<xsd:schema xmlns:xsd = "http://www.w3.org/2001/XMLSchema">
```

<그림 5> XML 스키마의 PI

나. 요소,속성 선언

DTD에서 요소(Element)와 속성(Attlist)를 사용할 경우 XMLSchema에서는 복잡타입(Complex Type)을 사용하여 표현한다.

```
<!ELEMENT Customer (#PCDATA)>
```

<그림 6> DTD의 ELEMENT 선언

```
<element name ="Customer" type="string" />
```

<그림 7> XML 스키마에서의 ELEMENT 선언

```
순서정의
<element name="Customer">
 <complexType name="CustomerType">
   <sequence>
     <element name="firstName" type="string"/>
     <element name="lastName" type="string" />
     <element name="emailAddress" type='string" />
 </sequence>
</complexType>
</element>
```

<그림 8> 자식 요소를 갖는 선언

요소	Tag	요소의 의미
선택	<choice>	DTD의 선택연산자 : (|)
순차 리스트	<sequence>	DTD의 순차리스트 연산자 : (,)
모든 그룹	<all>	무작위로 한번씩만

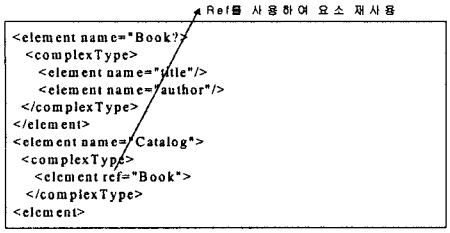
<표 8> 선택, 순차, 모든 그룹을 나타내는 요소

다. Any 선언

DTD의 ANY 내용 모델과 유사하고 항상 네임스페이스와 연계하여 사용하여야 한다. 이 요소는 모든 적법(well-formed) XML의 포함을 허용한다.

```
Catalog 스키마의 myCatalog 엘리먼트 사용
<element name="myCatalog">
  <complexType>
    <any
      namespace="http://www.ourSupplier.com/catalog"
      minOccurs="0" maxOccurs="unbounded"
      processContents="skip" />
  </complexType>
</element>
```

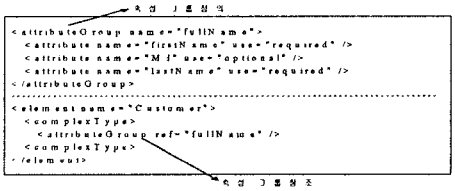
<그림 9> 다른 스키마로부터의 컨텐츠



<그림 10> Ref를 사용하여 요소 재사용

카디널리티연산자	minOccurs	maxOccurs	자식요소의 수
[None]	1	1	오직하나
?	0	1	없거나 하나
*	0	Unbounded	없거나 여러개
+	1	Unbounded	하나이거나 여러개

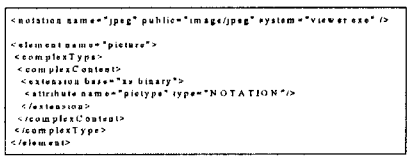
<표 9> DTD의 카디널리티와 XML스키마 카디널리티 비교



<그림 11> 속성 그룹

다. 노테이션 선언

하나의 노테이션에 대한 식별자와 이름을 결합하고, 적합성 검증과 직접적인 관련은 없지만, NOTATION의 단순 타입 정의에 해당하는 문자열의 적합성을 검증할 때는 참조 할 수 있다.

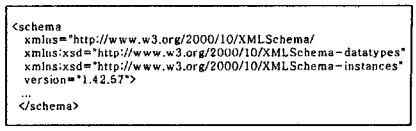


<그림 12> 노테이션(notation) 선언

4. XML Schema Tool 설계 및 구현

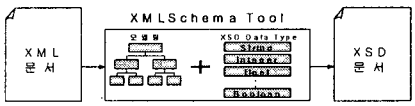
.4.1 시스템 설계

XML스키마는 XML문법을 사용하며, Valid 문서 이여야만 한다. <그림 13>는 <schema> 요소의 스키마와 그 안의 모든 자식 요소들에 의해 사용될 수 있는 모든 외부 스키마를 식별하기 위해 이 속성을 사용한다.



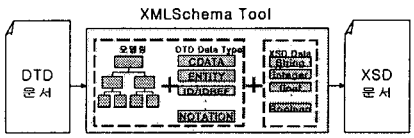
<그림 13> Schema 태그의 속성과 네임스페이스

아래 <그림 14>은 Well-Formed문서를 DOM Level2를 이용하여 XML스키마로 모델링하고. XML스키마의 데이터 타입을 추가하여 스키마 문서를 작성한다.



<그림 14> XML문서변환 XMLSchema Tool 시스템 설계

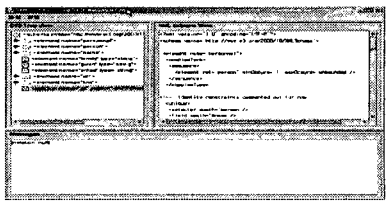
아래 <그림 14>는 DTD문서를 DOM Level2를 이용하여 XML스키마로 모델링하고. XML스키마의 데이터 타입을 추가하여 스키마 문서를 작성한다.



<그림 15> XML문서변환 XMLSchema Tool 시스템 설계

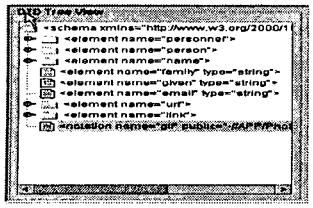
4.2 XMLSchema Tool 구현

XMLSchema Tool 실행화면이다. DTD Tree View와 XML Schema View, 그리고 Messages 부분으로 나눈다.



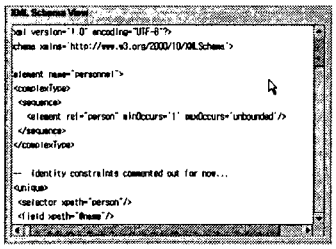
<그림 16> XML Schema Tool 실행화면

<그림17>은 DTD Tree View 부분으로 DTD문서를 읽어오면 DOM2.0을 사용하여 계층구조로 변환하고 한눈에 문서구조를 파악 할 수 있다.



<그림 17> DTD Tree View

<그림18>은 XML스키마로 변환된것이며 정교한 작업을 위하여 <그림19<를 두어 수정할 수있도록 하였다.



<그림 18> XML Schema View

<그림 19> Messages 편집

5. 결론 및 향후계획

본 논문은 DTD의 한계에 나타나는 문제점들을 극복하기 위하여 나온 XML스키마로 변환하는 도구를 설계 및 개발하였다. XML 스키마는 그 자체로도 XML 문서이며 이는 XML스키마는 XML 문서로도 취급할 수 있다는 것이다. 뿐만 아니라 XML 스키마는 데이터 타입을 지원한다. 즉, XML 스키마는 DTD에 정의된 데이터 타입을 지원할 뿐만 아니라 정수형, 실수형, date, times, string, URL 등의 데이터 타입에 대한 유효성 체크가 가능하다. DTD의 Element와 Attlist는 XML스키마에서 comolex type으로 모델링되고, 순서("|" , ",")는 <choice>, <sequence>, <all>로 표현된다. 외부 DTD는 외부스키마로 표현하고, ID, IDREF는 ID, IDREF, REF를 사용하여 객체 재사용를 하게한다. DTD의 커디널리티("none", "?", "*", "+")는 minOccurs, maxOccurs로 모델링하였다. DTD의 entity는 XML스키마의 속성그룹으로 모델링하고, notation은 스키마의 notation Element와 name 속성을 사용하여 모델링하였다.

본 논문에서는 XML스키마의 향후 관계형 데이터베이스와의 관계를 표현하지는 않았다. 전자상거래의 발전과 문서로서의 XML이 아니고 데이터로서의 XML이 되고자 한다면 XML스키마와 관계형 데이터베이스 스키마와의 모델링은 꼭 필요하다.

6. 참고문헌

[1] W3C, "Extensible Markup Language (XML) 1.0 : W3C Recommendation," http://www.w3.org/TR/1998/REC-xml-19980210.html, February 1998.

[2] OASIS, the Organization for the Advancement of Structured Information Standards, "XML Web Page," http://www.oasis-open.org/cover/xml.html, 1999.

[3] Charles F. Goldfarb and Paul Prescod, The XML Handbook, Prentice Hall PTR, 1998.

[4] Michael Leventhal, David Lewis and Matthew Fuchs, Designing XML Internet Applications, Prentice Hall PTR, 1998.

[5] Object Management Group, The Common Object Request Broker : Architecture and Specification Revision 2.2, February 1998.

[6] R. Orfali and D. Harkey, Client/Server Programming with Java and CORBA Second Edition, John Wiley & Sons Pub., 1998.

[7] Sing Li and Panos Economopoulos, Professional COM Applications with ATL, Wrox Press Ltd., 1998.

[8] Richard Grimes, Professional ATL COM Programming, Wrox Press Ltd., 1998.

[9] Richard Grimes, Professional DCOM Programming, Wrox Press Ltd., 1997.

[10] Cay S. Horstmann and Gary Cornell, Core Java 1.2 Volume 1, Prentice Hall PTR, 1999.