18-1-R03

DOM 기반 XML-스키마 생성도구 설계

(Design of XML Schema Generation Tools based on DOM)

이 돈 양, 송 영 재
경희대학교 전자계산공학과
dylee62@empal.com

요 약

XML기반으로 데이터를 이용한 어플리케이션 개발에서 적절한 클래스의 생성을 위해 DOM 트리를 이용하면 매우 유연한 방법을 가질 수 있다. 또한 DOM은 XML Document의 데이터 구조를 표현하는데 매우 적당하다. 본 논문에서는 이런 DOM의 기능을 이용하여 XMI 메타모델에서 표기한 패턴의 클래스에 대한 메타데이터를 연구하였다. 특히 일반적인 메타데이터나 마크업언어 생성보다는 메타모델에 적합한 도구의 설계에 중점을 두었고, DOM 트리의 파서 생성방법을 이용하여 XML-스키마 마크업언어 생성도구를 설계하였다. 그 결과 XML-스키마 마크업언어를 인터페이스를 이용하여 생성할 수 있었으며, 아울러 복잡하고 까다로운 XML-스키마언어를 쉽게 작성할 수 있도록 하였다

1. 서론

소프트웨어설계에서 디자인 패턴 및 클래스에 대한 부분을 정형화하고 표준화하기 위한 방법으로 OMG의 표준방식으로 제안된 XMI를 이용하고 있다. 본 연구에서는 메타모델생성과 XML 메타데이터 그리고 DOM 트리의 형식을 이용한 마크업언어의 정의 및 설계에 중점을 두고 있다. XML에서 데이터를 이용한 어플리케이션 개발은 적어도 두 가지의 기능으로 접근 할 수 있다. 첫 번째는 마크업언어의 설계에 관한 것이고, 두 번째는 적절한 클래스의 생성에 대한 것이다. 여기서 DOM[1] 트리를 이용하면 클래스생성에 관한 매우 유연한 방법을 찾을 수 있다. 그리고 DOM은 XML Document[2]의 데이터 구조를 표현하는데 적당하다[3]. 본 논문에서는 이런 DOM의 기능을 이용하여 XMI[8] 메타모델에서 표기한 패턴의 클래스에 대한 메타데이터를 생성하였다. 일반적으로 마크업언어 표기에서는 XML이 SGML의 DTD[6]를 기본적인 XML 문서의 구조를 정의하는 스키마로 채택하고 있으나 데이터표현방법과 정확한 문서정의에는 많은 문제점들이 있다. 이를 보완하기 위한방법으로 본 연구에서는 XML-스키마[7]를 이용한 마크업언어의 생성을 제안하였다. 덧붙여서 일반적인 메타데이터나 마크업언어의 생성보다는 메타모델[5]에 적합한 도구의 설계에 중점을 두고 DOM 트리의 파서 생성방법으로 XML-스키마를 이용한 마크업언어 생성도구를 설계하였다. 파서의 형태로는 SAX[4]와 DOM이 많이 사용되고 있으

나 여기서는 XML 파일의 구성을 쉽게 트리를 이용하여 변환할 수 있어 DOM의 형식을 이용하였다. 그 결과 XML-스키마를 이용한 여러 형태의 데이터표현이 가능한 마크업언어를 설계할 수 있었다.

2. 관련연구

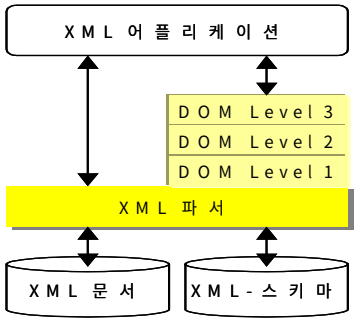
2.1 SAX

SAX 인터페이스는 매우 단순한 형태를 구성되었으며, XML 문서가 소프트웨어나 컴퓨터 자체에서 생성된 데이터정보를 읽어서 이용하는 어플리케이션 프로그램을 작성하는데 유용하다. 장점으로는 어떠한 크기의 파일도 파싱이 가능하고, 엘리먼트나 어트리뷰트를 사용하지 않는 own data structure를 만들 때 유리하다. 또한 필요 없는 데이터를 무시 할 수 있는 작은 단위의 정보를 파싱할 때도 매우 유용하다. 단점으로는 ID나 IDREF와 같은 internal cross-reference가 많으면 사용이 힘들고, 구조적으로 복잡한 경우 부모 등 인허리턴스(inheritance)를 찾기가 어렵다.

2.2 DOM

DOM은 DOM tree라는 하나의 트리를 구성하여 파서를 생성한다. 그리고 모든 엘리먼트와 어트리뷰트가 하나의 vertex로 표현이 되며, 트리의 root는 document의 root 엘리먼트가 되고, edge는 상호 종속관계로 표현이 된다. 장점으로는 문서의 모든 노드들이 parent와 child의 관계로 표현이 되고, DOM tree를 이용하여 XML 파일의 contents를 논리적인 표현으로 할 수 있다. 또한 DOM을 이용하여 이전의 파서형식보다 쉽게 XML 파일의 구성을 변환할 수 있어 개발자들이 쉽게 응용할 수 있다.

또한 DOM은 XML 문서구조와 내용을 다루기 위한 표준 응용 프로그래밍 인터페이스(API)로 XML 문서정보에 동적으로 접근하고, 갱신 및 수정, 탐색, 변경 등의 프로그램이 가능하도록 한다. DOM은 XML 파서의 기반으로 구성이 되었고, XML 파서는 XML 문서를 읽고 DOM 트리를 생성한다. 그리고 API를 통하여 DOM 트리에 접근할 수 있다.



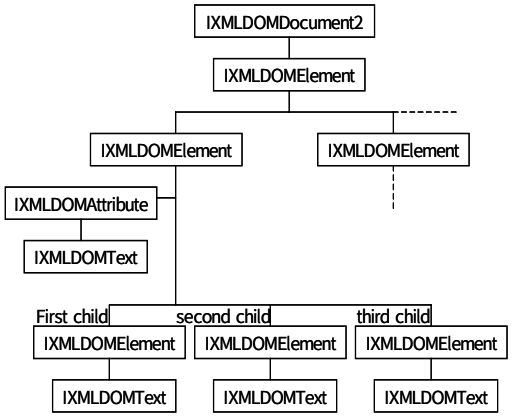
(그림 1) DOM 구조

W3C에서는 DOM의 표준으로 (그림 1)과 같이 세 단계의 레벨을 지원한다. 이 중 Level 1은 현재 대부분의 XML파서들이 지원을 하고 있다. 그리고 DOM은 XML 문서를 노드(node) 객체로 표현하고 이들에 대한 객체 인터페이스를 제공하고 있다. DOM에서 중요하게 제공하는 노드객체에 대한 타입으로는 Document, Element, Attribute, Text Node, Document Type, Namespace 등이 있다. XML 파서를 지원하는 라이브러리는 Microsoft의 MSXML, IBM의 XML Parser for Java, SUN의 Xerces 가 대표적이다. 본 연구에 적용한 MSXML 파서는 DOM Level 1과 XML-스키마언어와 SOM, XSLT, XPath, 그리고 XML에 관련된 표준을 지원하고 있다. 또한 VB, Java Script, JAVA, VC++ 등의 언어를 사용하여 DOM 인터페이스를 구현할 수 있다.

3. XML-스키마 문서형식 작성도구 설계

XML-스키마를 가지고 마크업언어를 개발하는 방법에서 XML 문서를 태그와 속성 및 내용으로 분리해서 어플리케이션을 해석하는

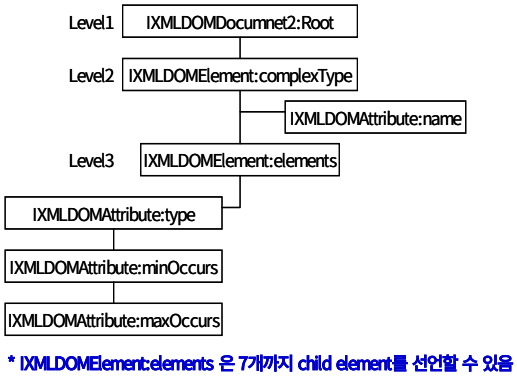
XML 파서(parser)가 요구된다. 본 연구에서는 이 파서의 기능을 가지고 마크업언어를 생성하기위해 Microsoft사의 MSXML 라이브러리를 이용하였다. 그리고 마크업언어는 XML 파일의 형태가 아닌 XML-스키마인 확장자가 "xsd" 파일로 생성하였으며, 메모리 상에 엘리먼트의 구조들을 트리(tree)형태로 형성하였다. XML-스키마의 DOM 트리 구조의 생성은 (그림 2)와 같은 XML 문서의 엘리먼트, 속성, 내용들을 하나씩 분류하는 형태로 하였다.



(그림 2) 엘리먼트 트리구조

4. 사용자 정의 콤플렉스타입(선택적 자식 엘리먼트사용)을 이용한 XML-스키마 DOM 트리 설계

MSXML 파서의 DOM 트리를 이용한 XML-스키마의 생성방법에는 기본적인 사용형태인 사용자정의 심플타입과 엘리먼트가 속성을 가지거나 자식엘리먼트를 가질 수 있는 사용자정의 콤플렉스타입이 있다. 그리고 본 연구에서는 다시 콤플렉스타입을 선택적 자식 엘리먼트를 사용하는 콤플렉스타입과 순차적 자식 엘리먼트를 사용하는 콤플렉스타입으로 두 개를 분류하여 XML-스키마를 위한 DOM 트리를 설계하였다. 여기서 선택적 자식 엘리먼트 사용 사용자 정의 콤플렉스 타입을 sequence 엘리먼트와 choice 엘리먼트를 사용하는 방법에 따라 4가지의 타입으로 분류하여 각각의 엘리먼트 트리와 DOM 트리를 설계하였으며 본 논문에서는 이중 첫 번째 타입을 적용하였다. 첫 번째 타입에 해당하는 1 Type은 가장 기본적인 콤플렉스 타입의 형태로 ROOT 아래에 콤플렉스 타입노드를 생성하여 하위부분에 여러 개의 엘리먼트를 생성할 수 있도록 한 것이다. 여기서는 최하위부분의 child 엘리먼트 개수를 7개로 하여 설계하였다.

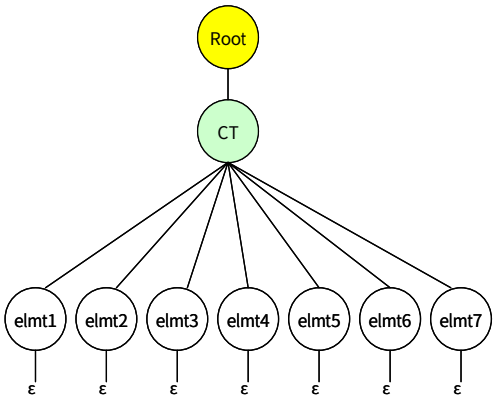


(그림 3) 사용자 정의 콤플렉스 타입 엘리먼트 1 Type 트리

(그림 4)와 같이 Document 트리를 생성하기 위한 조건으로 ∑를 XML 엘리먼트 타입의 유한집합으로, *D*는 데이터 타입의 유한집합으로 하여 아래를 정의하였다.

* Document 트리정의
1. ε is a tree
2. α(ω), where α∈∑ and ω∈D, is a tree
3. α(t1,t2, ..., tκ), κ≥1, where α∈∑ and ti is a tree, i = 1,2, ..., κ, is a tree
4. nothing else is a tree

그리고 Document 트리생성을 위해 ∑={Root, complexType, elmt1, elmt2, elmt3, elmt4, elmt5, elmt6, elmt7}로 두고 Root(complexType(elmt1,elmt2,elmt3,elmt4,elmt5,elmt6,elmt7))로 정의하여 트리를 도식하였다.



(그림 4) 사용자 정의 콤플렉스 타입 DOM 트리1

(그림 4)의 최하위에 있는 엘리먼트는 타입과 최소선택횟수로 minOccurs를 최대선택횟수로 maxOccur를 속성 값으로 가질 수 있도록 하였다. 그리고 사용자 정의 콤플렉스 타입에서 DOM 트리를 XML 문서의 문법형태로 정의를 아래와 같이 할 수 있다. 이것은 가장 단순한 형태로서 <complexType> 엘리먼트 아래에 <element1> ..... <elementn>까지의 엘리먼트를 가질 수 있도록 한 것이다.

```
<Root>
  <complexType>
    <element1>
      .....
    <elementn>
  </complexType>
</Root>
```

사용자 정의 콤플렉스의 두 번째 타입은 콤플렉스타입의 자식 엘리먼트로 sequence 엘리먼트를 사용하여 순차적으로 자식 엘리먼트를 기술할 수 있도록 한 것이고, 세 번째 타입은 콤플렉스타입의 자식 엘리먼트로 choice 엘리먼트를 이용하여도록 한 것이다. 마지막으로 네 번째 타입은 콤플렉스타입의 자식으로 엘리먼트 선언과 sequence 엘리먼트를 사용할 수 있으며 sequence 엘리먼트에서 다시 자식으로 엘리먼트 선언과 choice 엘리먼트 선언을 할 수 있도록 트리를 설계한 것이다.

사용자 정의 콤플렉스타입 XML-스키마언어를 생성하기 위한 방법으로 (그림 5)에서 각 엘리먼트들을 단계적으로 생성할 수 있도록 한 과정을 볼 수 있다. 기본 적인 생성방법은 심플타입 XML-스키마언어의 생성규칙과 같으며, sequence 엘리먼트를 선택하는 경우에 따라서 생성되는 자식트리의 형태를 알 수 있다.

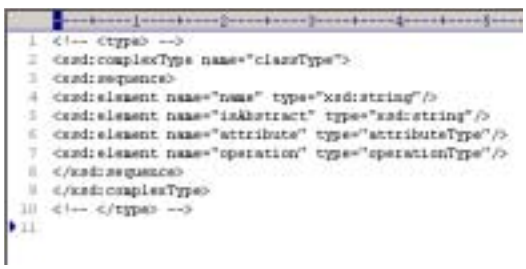
```
  set xmlDoc = New MSXML2.DOMDocument30
  set RootType = xmlDoc.documentElement
  set complexType = RootType.firstChild
  set complexType = xmlDoc.createElement
  call complexType.setAttribute("name")
  set complexTypeElement = xmlDoc.createElement
  select case complexTypeElementBase
     case 0 : "String"
     case 1 : "boolean"
     case 2 : "decimal"
          ...
     case 18 : "byte"
  end select
  call complexTypeElementBase.setAttribute
  if complexSequence <> " " then
    select case sequence
     case 0 : "sequence"
    end select
   if compexElement1 <> " " then
    set complexElement1 = xmlDoc.createElement
    call sequence.appendChild(complexElement1)
   end if
  else
   if complexElement1 <> " " then
    set complexElement1 = xmlDoc.createElement
    call sequence.appendChild(complexElement1)
   end if
  end if
  if complexElement1 <> " " then
   set complexElement2 = xmlDoc.createElement
  call complexType.appendChild(complexElement2)
  end if
  call complexType.setAttribute("type")
```

```
call complexType.setAttribute("minOccurs")
call complexType.setAttribute("maxOccurs")
if complexSequence <> " " then
    call complexType.appendChild(complexChild)
end if
call RootType.appendChild
```

(그림 5) 사용자정의 콤플렉스타입 XML-스키마생성 코드

(그림 6)의 사용자정의 콤플렉스타입(선택적 자식엘리먼트) XML-스키마 언어 인터페이스는 (그림 5)의 코드 작성규칙에 의해 설계된 GUI 폼이다. 여기서도 앞에서 소개된 심플타입 XML-스키마 언어의 생성방법과 마찬가지로 DOM 트리의 형식에 따라 마크업언어를 생성하였다. 그리고 엘리먼트의 속성으로 DTD에서 지원하지 못하는 광범위한 데이터 타입을 정의 가능하게 하고, 최소발생횟수와 최대발생횟수에 대한 속성을 추가하고 있다.

(그림 6) 사용자정의 콤플렉스타입 XML-스키마 인터페이스



(그림 7) 생성된 Class 엘리먼트

5. 결론

XML 메타데이터의 마크업언어의 생성에서는 DTD와 XML-스키마가 주로 쓰이고 있으나 본 연구에서는 엘리먼트에서 다양한 타입의 데이터형식 지원이 가능한 XML-스키마 형식을 선택하였다. 그리고 일반적인 메타데이터나 마크업언어 생성보다는 메타모델에 적합한 도구의 설계에 중점을 두고 DOM 트리의 파서 생성방법을 이용하여 XML-스키마 마크업언어 생성도구를 설계하였다. 그 결과 XML-스키마 마크업언어를 인터페이스를 이용하여 생성할 수 있었으며, 아울러 복잡하고 까다로운 XML-스키마언어를 쉽게 작성할 수 있었다. 마지막으로 향후 연구에 대해서는 현재 대부분의 스키마언어 생성 툴 들이 DTD기반에서 Low-level 문서작성의 형식으로 구현되고 있어 데이터 형식의 구조에 대한 정확하고 다양한 표현에 한계가 있다. 이를 보완하기 위한 방법으로 UML로 설계된 패턴모델형식을 문서작성 중심의 물리적 데이터구조(physical data structures)로 스키마를 생성하도록 하는 것이다. 그리고 본 연구에서 제시한 것 이외의 다른 타입의 스키마 형식에 대한 연구도 계속해서 이어질 것이다.

참고문헌

[1] Lauren Wood+. W3, Document Object Model(DOM) Level 1 Specification, http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001, October 1998.

[2] Chien, S.Y., Tsotras, V.J., Zaniolo, C, "XMLDocument Versioning", SIGMOD Record 30, pp. 46-53, 2001.

[3] Norihiro Ishikawa, Takeshi Kato, Hidetoshi Ueno, "Automatic Generation of a Differential XSL Stylesheet From Two XML Documents", Hiromitsu Sumino and

Hideharu Suzuki Multimedia Laboratories, NTT DoCoMo 3-5, Hikari-no-oka, Yokosuka, Kanagawa, 239-8536, Japan 2002.
[4] W3C, "Simple API for XML Version 2.0", WC3 Recommendation, November 1999.
[5] Jean-Guy Schneider and Markus Lumpe, "A Metamodel for Concurrent, Object-based Programming", Proceedings of LMO'00, Qubec, pp. 149-165, January 2000
[6] R. Behrens, "A grammar based model for XMLschema integration. In British National Conference on Databases (BNCOD)", pages 172-190, 2000
[7] David C. Fallside, editor. XML schema Part 0: Primer. World Wide Web Consortium, 2000
[8] "XMI Gets the Capability to convey information",http://www.3.ibm.com/software/ad/ standards/xmiwhite0399.pdf, 1999

이 돈 양
1987년 대구대학교 통계학과(이학사)
1993년 경희대학교 전자계산학과(공학석사)
2002년 경희대학교 전자계산공학과(박사수료)
1995년 ~ 2002년 대한상공회의소
2003년 ~ 현재 경인여대 전산정보 겸임교수
관심분야 : EJB, 디자인패턴, XML, OOD, 소프트웨어 재사용

송 영 재
1969년 인하대학교 전자공학과(공학사)
1976년 일본 keio 대학교 전산학과(공학석사)
1980년 명지대학교 전산학과(공학박사)
1976년 ~ 현재 경희대학교 컴퓨터공학과 교수
관심분야 : 소프트웨어재사용, OOP, CBD