제 19 회 한국정보처리학회 춘계학술발표대회 논문집 제 10 권 제 1 호 (2003. 5)

UML 표기법을 이용한 XML Modeling의 확장

김항배*, 조민호*, 류성열*
*숭실대학교 컴퓨터학과
e-mail : hangbee@selab.ssu.ac.kr

XML Modeling's extension that use UML notation

Hang-Bae Kim*, Min-Ho Cho*, Sung-Yul Rhew *
*Dept. of Computer Science, Soongsil University

요 약

HTML 의 사용상의 한계로 인해 등장한 XML 은 현재 여러 분야에서 다양하게 사용되고 있다. XML 모델링 이란 어플리케이션과 사용자가 요구하는 정보를 가지고 미래에 요구되는 변화에도 충분히 대응할 수 있는 유연한 XML 문서를 제작하는 과정이다. 이에 따라 XML 문서를 모델링 하기 위한 방법들이 나오게 되었으며 좀더 정확하고 유연한 XML 문서의 제작을 위한 모델링기법이 요구되고 있다. 본 연구에서는 현재의 모델링언어 중 보편적으로 사용되고 있는 UML(Unified Modeling Language)표기법을 이용하여 XML 을 모델링 하는 기존의 모델링기법을 소개하고 기존의 방법이 가지고 있는 어트리뷰트와 엘리먼트의 식별, 네임스페이스표기 등 문제점을 파악한 후 문제점들을 개선한 XML 모델링기법을 제안하였다. 기존에 모델링기법에서 표현하지 못하였던 어트리뷰트(Attribute)와 엘리먼트(Element)를 구별하였고 네임스페이스(Namespace)의 표기를 나타내었다.

1. 서론

현재 웹(Web)에서 컨텐츠(Contents)표현 언어로 가장 많이 사용되고 있는 언어는 HTML 이다. 하지만, 특수 분야를 중심으로 XML 의 활용이 매우 증가하고 있으며, 대표적인 예로는 전자상거래, 인터넷 검색엔진, 문서관리, 전자도서관 등이 있다. XML 의 사용이 확산되는 이유는 XML 문서를 만드는 개발자가 구조화된 문서를 정의하고 자유롭게 태그를 정의할 수 있다는 SGML 의 장점과 인터넷 상에서 손쉽게 하이퍼미디어(Hypermedia)문서를 제공할 수 있는 HTML 의 장점을 그대로 보유하고 있기 때문이다. 이러한 특징을 가진 XML 을 제대로 활용하기 위해서는 주어진 상황을 XML 로 표현하는 것이 중요하다. 즉, XML 의 특징을 살려서 문서를 설계하기 위한 모델링은 다음과 같이 진행되어야 한다. 첫째, 빠르게 이루어져야 하며 둘째, 정해진 틀을 가져야 하고 셋째, 정확하게 이루어져야 한다. 올바르게 모델링이 되지 않은 XML 문서는 탐색이나, 확장성 등에서 문제가 된다. 이에 본 논문에서는 UML 을 이용하여 XML 을 모델링 하는 기법들을 소개하고 XML 이 가진 특성들을 고려한 모델링 기법을 제안함으로써 기존의 UML 을 이용한 XML 모델링기법을 확장하였다.

2. 관련연구

2.1 XML 모델링기술

XML 의 Modeling 은 어플리케이션과 사용자가 요구하는 모든 필요한 정보를 가지고, 또한 미래에 요구되는 어떠한 변화에도 충분히 대응할 수 있을 만큼 유연한 XML 문서를 제작하는 과정이라 할 수 있다. XML 데이터 모델링은 아래와 같은 과정을 거친다.

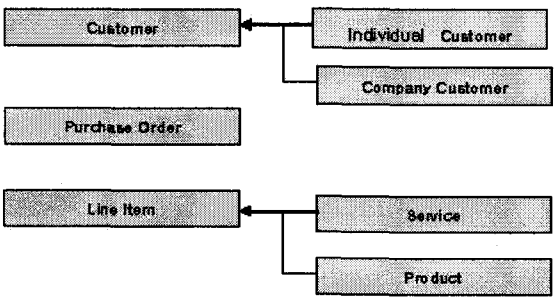
① 정보 모델링(Information Modeling)
② 문서설계(Designing XML documents)
② 스키마 작성(schema Languages and notations)

정보 모델링은 정보가 나타내는 의미를 알고 정보의 구조를 이해하는 것이고, 문서설계는 정보모델을 어떠한 룰이나 스키마를 가지고 표현하는 것이며 스키마 작성은 작성된 문서의 구조를 요약하여 다른 사람이 참고할 수 있도록 하는 것이다. 정보 모델링에서 정의하고 있는 모델은 시스템이 가질 수 있는 상태, 속성, 관계에 대한 모델인 정적정보모델(Static Information Model)이 있고 시스템의 업무 흐름이나 프로세스, 데이터 흐름에 대한 모델인 동적 정보 모

델(Dynamic Information Model)이 있다. 본 연구에서는 기존의 모델링을 개선하고자 하는 것을 목적으로 하고 있으며, 개선할 부분의 범위는 정적 모델링에 해당된다.

2.2 정적 정보 모델링

정적정보모델은 시스템이 가질 수 있는 상태, 속성, 관계에 대한 모델로서 4 가지 단계를 거쳐서 생성한다. [그림 1]은 일반적인 구매 시스템을 정적 모델링 하는 과정을 단계별로 나타낸 것이다. 모델링 할 자료의 요구사항은 다음과 같이 주어진다. 고객은 개인일 수도 있고, 회사일 수 도 있다. 고객은 구매 의뢰서를 낼 수도 있고, 안 낼 수도 있다. 내는 경우에 여러 개를 낼 수도 있다. 구매 의뢰서는 구매 품목에 대한 정보가 포함되지 않는 경우도 있고, 여러 개 포함될 수도 있다. 구매 의뢰서의 각 항목은 제품 구매에 대한 것이거나, 서비스 구매에 대한 것, 둘 중의 하나이다. 이와 같은 요구사항을 가지고서 행해지는 첫 번째 단계에서는 객체의 식별, 객체에 이름 부여, 객체의 정의를 해주어야 한다. 객체의 이름을 부여할 때는 구현하고자 하는 시스템과 관련된 객체의 리스트를 만드는 것으로 시작한다. [그림 1]에서 Customer, Individual, Company, Purchase Order, Line Item, Service, Product 등으로 객체를 만들고 이름을 부여하였다. 그 다음 객체를 식별하여 주어야 하는데 위에 이름을 부여한 객체들은 [그림 1]과 같이 식별 할 수 있다.



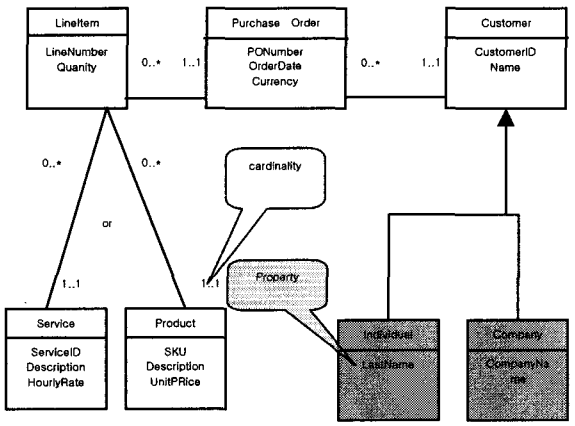
[그림 1] 객체식별

고객은 개인고객과 회사고객으로 식별 할 수 있고 구매의뢰서(Line Item)은 각각 서비스와 생산물로 나누어 식별 할 수 있다. 이렇게 이름을 붙이고 식별해준 객체에 마지막으로 객체를 정의 해준다. 객체의 정의는 [표 1]의 설명과 같다.

[표 1] 객체의 정의

객체이름	객체정의	기타
Customer	고객에 대한 것을 지칭한다	개인, 회사로 구분
Individual	고객 중에서 개인 고객을 지칭한다	
Company	고객 중에서 회사 고객을 지칭한다	
Purchase Order	고객이 주는 구매 의뢰서를 지칭한다	
Line Item	구매의뢰서의 각 항목을 지칭한다	Service,Product 로 구성
Service	구매의뢰서의 항목 중에서 서비스 구매 항목	
Product	구매의뢰서의 항목 중에서 제품 구매 항목	

처음 단계가 끝나고 나면 두 번째 단계는 클래스계층으로 구성 한다. 아래의 그림은 각 객체들을 클래스 계층으로 구성한 것이다. 세 번째 단계는 객체의 클래스 계층 구성을 끝낸 후 구성한 각각의 클래스들의 관계를 찾아서 표시 해주어야 한다. [그림 2]는 각 클래스의 관계를 나타내고 있으며 다중성(cardinality)도 나타내었다. 클래스간의 관계를 정의하고 나면 마지막 네 번째 단계에서 클래스 들의 구체적인 행동과 상태를 나타내 주는 속성(Property)을 정의하여 준다. [그림 2]는 클래스 다이어그램에 속성(Property)을 정의해준 그림이다.



[그림 2] Property 정의

위의 네 가지 단계를 거치고 나면 정적 클래스 다이어그램으로 만들어 진다. 이렇게 만들어진 클래스 다이어그램을 가지고 XML 문서를 생성할 수가 있는데 [그림 3]은 만들어진 데이터 모델을 가지고 XML 문서로 매핑 한 것이다.

```
<?xml version="1.0">
<PurchaseOrder PONumber="124442" OrderDate="2001-03-14"
                    Currency="US$">
    <LineItem LineNumber="1" Quantity="3">
        <Product SKU="14414" Description="Widget"
                    UnitPrice="2.5"/>
    </LineItem>

    <LineItem LineNumber="2" Quantity="5">
        <Product SKU="14413" Description="Gromets"
                    UnitPrice="2.5"/>
    </LineItem>
</PurchaseOrder>
```

[그림 3] 정보모델링을 통한 XML 문서의 메핑

3. XML 모델링 기술의 문제점

현재 XML 모델링은 몇 가지 문제점을 안고 있다. 그 문제점은 아래와 같다.

① 표현 방식이 일반적이지 않다.
② 어트리뷰트와 엘리먼트의 구분이 되지 않는다.
③ 네임스페이스의 개념이 들어가 있지 않다.

첫 번째 문제점은 그 동안 XML 을 모델링 하기 위한 모델링 언어가 정해지지 않았다는데 문제가 있다. 지금 까지는 각각 나름대로 모델링 언어를 사용하여 왔기 때문에 모델링 된 자료를 필요로 하는 사람들이 이해하기 힘든 면이 있었다.

두 번째는 XML 문서의 구조상 XML 태그는 어트리뷰트가 사용 될 수도 있고 엘리먼트가 사용 될 수도 있는데 어트리뷰트와 엘리먼트를 모델링 할 때 어떻게 구분해야 하는지에 대한 문제가 발생할 수 있다. 객체를 클래스로 표현할 때 클래스 안에 속성을 정의해 주게 되는데 그 안에 어트리뷰트와 엘리먼트 두 가지가 다 들어갈 수 있기 때문에 구분해주는 것이 필요하다.

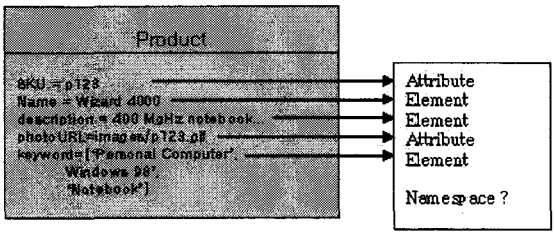
세 번째로는 네임스페이스의 문제를 들 수 있겠는데 XML 문서에서 네임 스페이스는 각기 다른 XML 문서 타입으로부터 엘리먼트들을 뽑아내어 다른 문서로 결합시킬 때, 또는 여러 개의 문서를 동시에 처리하고 있을 때 엘리먼트들을 구별할 수 있는 수단을 제공한다. 네임스페이스는 XML 태그의 중복 사용시 그것들을 구분시켜 주기 위해서 사용한다. 그러므로 모델링 시에 각 객체마다 네임스페이스에 대한 정의를 해주어야, 각 객체가 자체 생성인지, 외부 참조인지를 확인할 수 있고, 향후, 제작된 XML 문서를 활용할 때에도 모델링의 객체 단위로 분리하여 응용할 수 있다. 현재의 XML 의 모델링 기법에서는 이 네임스페이스를 나타내지 못하는 점이 발견되었다.

4. 개선된 XML 모델링 기법의 제안

본 절에서는 XML 모델링 기법의 문제점들에 대한 해결을 함으로써 좀더 향상된 XML 모델링기법을 제안하고자 한다. 처음 제기 되었던 일반적이지 않은 표현 방식의 문제는 UML(Unified Modeling Language)를 사용함으로써 해결할 수 있다. UML 은 1998 년 OMG 그룹에서 1.2 버전이 발표된 이후로 현재 1.4 나와 있으며 소프트웨어 중심 시스템의 산출물을 가시화 하고, 명세화 하며, 구축하고, 문서화 한다. UML 은 현재 객체지향 설계부분에서 많이 사용 하고 있으며 가장 널리 사용되는 모델링 언어이다. 이에 XML 에서도 UML 을 이용 모델링을 함으로써 모델링 언어를 일괄적으로 통일 할 수 있다.

두 번째 문제점은 XML 문서의 특성상 Attribute 와 Element 의 요소를 클래스에 표현 할 때 어떻게 구분하여줄 것인가 하는 문제이다. 현재 UML 을 가지고 모델링을 한다고 해도 이 부분은 구별되지 않는다. 또한 네임스페이스의 경우도 현재의 모델링 표기로써는 표현하지 못하고 있다 아래의 예는 간단한 XML 문서를 데이터모델링 한 것이다.

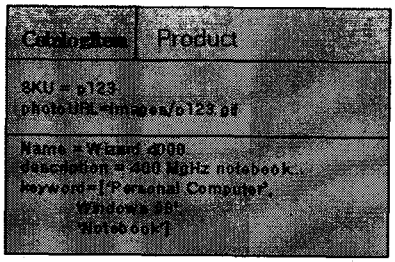
```
<Product sku=" p123"  photoURL=" images/p123.gif" >
   <CatalogItem:name>Wizard 4000</CatalogItem:name>
   <CatalogItem:description>
      400 MgHz notebook computer with Windows 98
   </CatalogItem:description>
   <CatalogItem:keyword>Personal
Computer</CatalogItem:keyword>
   <CatalogItem:keyword>Windows 98</CatalogItem:keyword>
   <CatalogItem:keyword>Notebook</CatalogItem:keyword>
</Product>
```



[그림 4] XML 문서의 UML 클래스 표기

[그림 4]와 같이 XML 문서를 클래스로 표현하였는데 현재의 모델링 방법에서는 어떤 것이 어트리뷰트 이고 엘리먼트 인지의 구분이 모호하며 위에서 사용한 네임스페이스 또한 처리하지 못하였다. 위의 그림을 가지고 XML 문서를 만든다면 정확한 문서가 되지 못할 것이다. 이에 어트리뷰트와 엘리먼트를 구분할 수 있고 네임스페이스 문제를 해결한 모델링 기법을 제안한다.

어트리뷰트와 엘리먼트의 경우 새로운 표기법을 제안한다. 기존의 XML 모델링에서는 클래스 안에 어트리뷰트와 엘리먼트를 전부 표기한 형태이나 이 부분을 둘로 나누어 상부에는 어트리뷰트를 하단에는 엘리먼트를 표시함으로써 어트리뷰트와 엘리먼트를 구분할 수 있다. 이 방법은 툴로 구현하였을 경우 구현하기 편리하게 해준다. 네임스페이스의 경우는 클래스에 표기할 때 클래스 다이어그램에 네임스페이스의 공간을 넣어줌으로써 해결하였다. [그림 5]는 네임스페이스 공간을 넣은 클래스이다.



[그림 5] XML 클래스의 제안

5. 비교평가

두 모델간의 비교평가를 위하여 아래와 같은 항목을 정하였다. 각 항목은 각각의 가중치를 두었다. 각각의 가중치는 다음과 같다.

- 네임스페이스 표현이 가능: 1 불가능: -1
- 어트리뷰트 표현이 가능: 1 불가능: -1
- 엘리먼트 표현이 가능: 1 불가능: -1
- 문서 -> 모델 변환이 가능: 0.5
- 모델 -> 문서 변환 정확성 = {가중치 합계(3.5) + 불가능요소 가중치} * 28.5 = % (결과값은 소수 첫째 자리에서 반올림한다)

평가는 기존의 XML 모델링기법과 개선된 모델에 각각 적용시켜 보았다. 여기서 유의할 점은 기존의 모델링기법을 XML 문서의 요소인 네임스페이스, 어트리뷰트, 엘리먼트의 3 요소를 전부 사용하는 경우, 네임스페이스, 어트리뷰트만 사용하는 경우 네임스페이스, 엘리먼트 만 사용하는 경우 엘리먼트만 사용하는 경우 이렇게 네 가지 조합으로 분류하였다. 이렇게 한 이유는 XML 문서를 제작할 때 네임스페이스, 어트리뷰트, 엘리먼트의 요소가 전부 쓰이지 않을 수 있기 때문이다. 아래 표는 두 가지 모델링을 비교평가 하였다. 위에 표들을 비교한 결과 개선된 모델링이 기존의 모델링에 비하여 문서로 표현 시 정확성이 높다는 것을 알 수 있다.

[표 2] 기존 모델링과 개선된 모델링의 비교

모델 / 평가 항목	기존모델링기법				확장된모델링기법
	Namespace Attribute Element	Namespace Attribute	Namespace Element	Element	
네임스페이스표기	불가능	불가능	불가능	-	가능
어트리뷰트 식별	불가능	가능	-	-	가능
엘리먼트 식별	불가능	-	가능	가능	가능
문서 -> 모델 변환가능	가능	가능	가능	가능	가능
모델 -> 문서 변환정확성(%)	14	71	71	100	100

6. 결론

현재 웹에서 컨텐츠 표현 언어로 가장 많이 사용되고 있는 HTML 의 기능적인 한계 때문에 등장한 XML 다양한 분야에 사용되고 있다. XML 모델링은 미래에 요구되는 어떠한 변화에도 충분히 대응할 수 있을 만큼 유연한 XML 문서를 제작하는 과정이다. XML 모델링은 현재 몇 가지의 문제점을 안고 있는데 XML 데이터 모델링을 위한 통일된 언어의 부재 그리고 XML 문서의 구조적인 특징인 어트리뷰트와 엘리먼트의 구별, 네임스페이스 문제가 그러하다. 이러한 문제점들을 해결하기 위해 현재 가장 보편적으로 많이 사용하고 있는 UML 을 이용하여 XML 의 구조적인 특징적인 부분을 보완 어트리뷰트와 엘리먼트를 구별하고 또한 네임스페이스의 개념을 클래스에 넣어서 보다 완전한 XML 문서를 구현할 수 있는 모델링 기법을 제안하였다. 이 방법은 클래스 안에 공간을 나누어 어트리뷰트와 엘리먼트를 표기하고 네임스페이스를 표현하였기에 툴로 구현하더라도 편리하며 네임스페이스의 개념을 나타내었기 때문에 XML 문서의 각 엘리먼트를 외부에서 가져오는 경우에 네임스페이스로 식별할 수 있고, 향후 다른 회사에서 만든 문서와의 병합 시 혼돈되지 않는다. 또한 어트리뷰트와 엘리먼트를 구별하여 줌으로써 사용자의 요구 사항에서 자료의 가치를 가지는 데이터와 보조데이터를 구분할 수 있다. 현재까지의 연구는 확장된 데이터 모델링을 제안하는데 까지 그쳤으나 앞으로 연구를 확장 발전시켜 제안된 데이터 모델링을 구현한 툴을 개발 하는 것이 향후 과제이다.

참고문헌

[1] Berners-Lee, Tim Berners-Lee, "Weaving the Web," New York: HarperCollins, 1999.

[2] Booch, Grady, james Rumbaugh, and Ivar Jacobson. "The Unified Modeling Language UserGuide," Reading, MA: Addison-Wesley, 1999.

[3] Bradley, Neil Bradley, "The XSL Companion," Boston: Addison-Wesley, 2000.

[4] Conallen, James Conallen, "Building Web Applications with UML," Boston: Addison-Wesley, 2000.

[5] Fowler, Martin Fowler, Kendall Scott, "UML Distilled: A Brief Guide to the Standard Object Modeling Language, Second Edition," Boston: Addison-Wesley, 2000.

[6] Jacobson, Ivar Jacobson, Grady Booch, and James Rumbaugh, "TheUnifiedSoftware Development Process," Reading, MA: Addison-Wesley, 1999.

[7] Kay, Michael Kay, "XSLT Programmer's Reference," Birmingham, UK: Wrox Press, 2000.

[8] Kobryn, Cris Kobryn, "UML 2001: A standardization Odyssey," Communications of the ACM. October 1999.