객체-관계형 데이터베이스 기반의 XML 응용설계 방법

김경수, 주경수
천안외국어대학 컴퓨터정보과
순천향대학교 공과대학 컴퓨터학부

A Design Methodology for XML Application based on ORDB

Kim Kyung-Soo, Joo Kyung-Soo
Dept. of Computer Information,
Cheonan College of Foreign Studies
Dept of Computer Science and Engineering
College of Engineering SoonChunHyung Univ
E-mail : kkskim@ccfs.ac.kr, gsoojoo@asan.sch.ac.kr

요 약

현재 B2B 전자상거래와 같은 XML을 이용한 정보교환이 확산되고 있으며, 이에 따라 상호 교환되는 정보에 대하여 체계적이며 안정적인 저장관리가 요구되고 있다. 이를 위해 XML 응용과 데이터베이스 연계를 위한 다양한 연구가 관계형 데이터베이스를 중심으로 수행되었다. 그러나 계층구조를 갖는 XML 데이터를 2차원 테이블의 집합인 관계형 정보로 표현하는 관계형 데이터베이스로의 저장에는 본질적인 한계가 있으므로, 해결책으로 계층구조를 지원하는 ORDB로의 저장이 요망되고 있다. 따라서 계층구조를 갖는 XML 데이터를 ORDB로 저장하기 위한 모델링 방안이 요구된다. 양질의 어플리케이션 시스템을 구축하기 위해서는 우선적으로 모델링이 중요하다. 본 논문에서는 UML을 이용하여, ORDB 기반의 XML 응용을 위한 통합 설계 방법론을 제안한다. 이를 위하여 먼저 UML을 이용하여 XML DTD를 설계하기 위한 XML 모델링 방안을 제시하고, 아울러 교환되는 XML 데이터를 효율적으로 저장하기 위하여 ORDB 스키마 설계를 위한 데이터 모델링 방법을 제안한다.

1. 서론

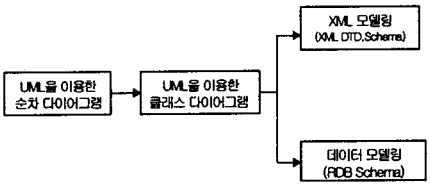
XML은 구조화된 정보를 포함하고 있는 문서들을 위한 마크업 언어이다. 구조화된 정보는 구체적인 내용과 그 내용이 수행해야할 역할을 포함하고 있다[3]. 한편 XML 어플리케이션과 데이터베이스 시스템 사이의 원활한 연계를 위해서, 그 동안 XML DTD를 관계형 데이터베이스 스키마로 변환하는 방법에 대해서는 많은 연구가 진행되었다. 본 논문에서는 UML을 기반으로 하여 XML DTD를 산출하고, 그 결과를 이용하여 ORDB 스키마로 저장하는 통합 모델링에 대한 연구를 하였다. 본 논문에서는 2장에서 UML를 이용한 모델링, 3장에서는 XML 모델링, 4장에서는 데이터 모델링 그리고 마지막 5장에서는 결론을 기술한다.

2. UML를 이용한 모델링

UML은 소프트웨어 시스템의 구조물을 명세화, 구조화, 시각화, 문서화뿐만 아니라 비-소프트웨어 시스템 그리고 비즈니스 모델링을 위한 하나의 언어이다[7]. 또한 XML 프로그램이나 XML를 이용한 B2B 시스템 구축과 같은 XML 프로젝트에서는 객체 지향적 설계 언어인 UML을 이용하여 개발의 효율을 높일 수 있다.

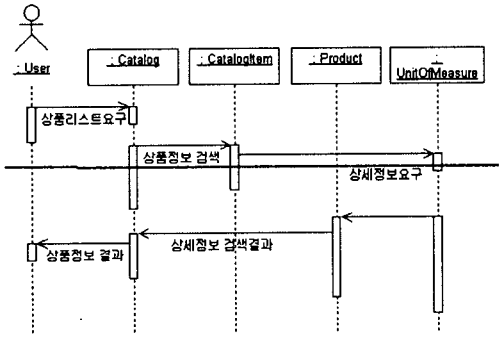
순차 다이어그램은 응용 프로그램에서 사건의 흐름을 분명하게 하는 역할을 하며 글로 풀어서 쓴 유스케이스(Use case)를 메시지로 표현한 그림으로 변환할 수 있다. 또한 그 그림은 항상 더 간결하게 이해되고, 특히 우리가 클래스 다이어그램을 만들 때 작업을 쉽게 해준다. 그러므로 UML을 이용해 유스케이스에

의한 순차 다이어그램을 도출하여 클래스 다이어그램을 만든 후 그 클래스 다이어그램에 의해서 그림 1과 같이 XML 모델링과 데이터 모델링으로 변환할 수 있다.



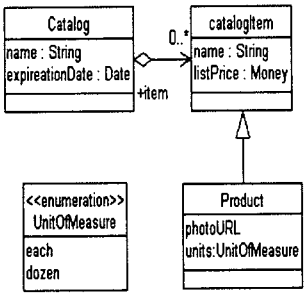
(그림 1) UML을 이용한 XML 모델링과 데이터 모델링

그림 2의 순차 다이어그램은 객체들간의 메시지과정을 설명한다. 이 다이어그램은 메시지 객체들과 진행과정에 관계된 액터들을 포함하고 있다[2].



(그림 2) 메시지 객체와 순차 다이어그램

그림 2에서 객체들과 액터들 이외에 수직의 축은 시간을 표현한다. 화살표는 행위 또는 객체와 액터들간에 발생하는 사건을 표현한다. 그림 3의 클래스 다이어그램은 4개의 클래스, 그리고 그들의 연관된 속성과 서브-객체를 예를 들어 설명한다.



(그림 3) 클래스 다이어그램

그림 3은 Catalog 와 CatalogItem간에 관계성에서 연관 관계를 갖는다. 왜냐하면 CatalogItem은 Catalog 객체에 의존하고 있기 때문이다. 또한 CatalogItem은 Product의 부모 클래스가 된다. 이것은 Product 클래스에서 CatalogItem의 일부 속성을 상속받기 위해서이다. 속성의 상속은 하위 클래스에서 유용하게 사용되기 때문이다. 표 1은 각 클래스의 객체의 특성을 서술한 것이다. 또한 각각의 객체에 대한 속성, 사상수, 관계성은 표 2와 같다.

(표 1) 객체 정의

클래스	기술
Catalog	상품목록의 이름과 그 정보에 대한 만기일자를 가지고 있다. 사용자가 특정 카다로그의 목록을 원하면, CatalogItem 객체를 통해서 정보를 가져온다.
Catalog Item	특정한 목록의 이름과 가격정보를 가지고 있다. 자식 클래스인 Product를 이용하여 각 상품에 대한 정보를 표현할 수 있다.
Product	CatalogItem에서 상속받은 속성과 자신이 가지고 있는 속성을 이용하여 하나의 제품에 대한 여러 가지 정보를 표현한다.

(표 2) 객체들의 속성, 사상수, 관계성

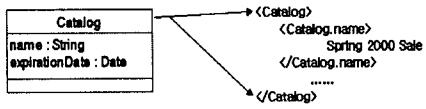
클래스	속성	사상수	관계	기술
Catalog	name	1. .1	복합	상품의 목록을 기술
	expiration Date	1. .1	복합	만기일자를 기술
Catalog Item	name	1. .1	복합	세부 상품이름을 기술
	ListPrice	1. .1	복합	상품의 가격을 기술
Product	PhotoURL	1. .1	복합	상품의 사진이 있는 URL을 기술
	Units	1. .1	복합	단위 수량을 기술

3. XML 모델링

XML 어플리케이션에서 기본이 되는 XML DTD 설계 과정에서 최적의 DTD가 되도록 하기 위해서 물리적 모형이나 도해를 만드는 것을 말한다.

3.1 UML를 XML로의 변환 방법

① UML 클래스를 XML 엘리멘트로 변환

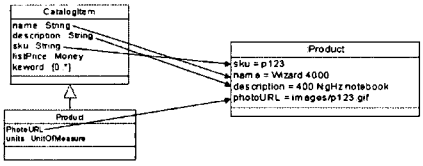


(그림 4) UML 클래스의 변환

UML 클래스의 경우 XML로의 변환은 그림 4와 같이 하나의 XML 요소로 완전히 변환된다. XML 태그 이름은 UML 클래스 이름과 같으며, 태그 이름은 공백을 포함하지 않고, 알파벳, 숫자만으로 제한된다. 특수문자인 '.','-','_'를 포함할 수 있으며, 태그 이름은 각각의 글자나 '_'로 시작한다. UML 클래스나 패키지 이름은 XML 문자로 사용되지 않는다. 그림 4는 UML 클래스를 XML 엘리멘트로 변환한 것이다[1,6].

② 상속(Inheritance)

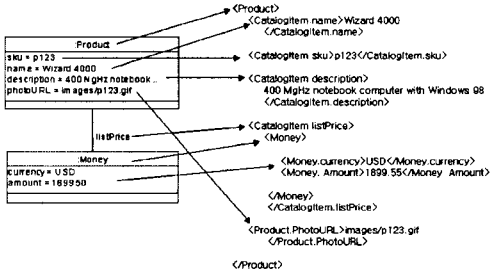
상속을 사용하는 것은 객체 지향 모델링에서 기본적으로 적용되는 특성이다. 그림 5는 상속을 UML 다이어그램으로 표현한 것이며, 상속받은 클래스는 부모 클래스의 어트리뷰트를 사용할 수 있다. 이것은 XML DTD로의 변환에서도 중요하게 사용된다.



(그림 5) 상속을 UML 다이어그램으로 표현

③ UML 어트리뷰트를 XML 엘리멘트로 변환

UML 클래스의 인스턴스가 주어졌을 때, XML에 어트리뷰트들이 각각 매칭하게 된다. 이것은 XML 변환기법들을 디자인할 때, 자식 엘리멘트를 분리하는 것과 같이 각 데이터 값을 정의하는 일반적인 관례이다.



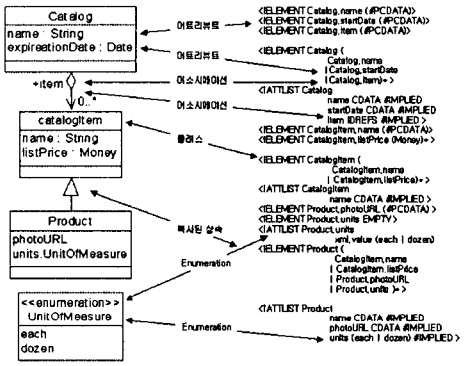
(그림 6) 어트리뷰트 값을 가진 하나의 객체와 변환된 XML 문서

④ UML 어트리뷰트를 XML 어트리뷰트로 변환

UML 어트리뷰트는 XML 어트리뷰트로 표현이 가능하다. 그림 6의 클래스 다이어그램을 XML로의 변환하는 것을 기초로 한다면, product 엘리멘트의 sku와 photoURL UML 어트리뷰트는 XML 문서의 어트리뷰트로 변환된다. currency와 amount는 money 엘리멘트의 XML 어트리뷰트로 변환된다. 만약 UML 어트리뷰트의 값이 string 값을 포함한다면, XMI(XML Metadata Interchange)는 각각의 엘리멘트나 어트리뷰트를 허용한다. 반면에 XML 어트리뷰트의 값이 항상 공백이 된다면, XML 파서는 모든 공백 문자(탭, 개행 문자, 리턴, 여러 가지 공백)를 삭제한다[2].

3.2 UML을 XML DTD로 변환

XMI는 UML 문서로부터 XML DTD 생성시 유연하게 처리하거나 강제적으로 느슨하게 변환한다. 이 방법은 DTD의 표현에 제한을 두었기 때문에 부분적인 모델의 변환을 가능하게 만들었다. 반면에, UML에서 엄격한 DTD의 생성을 가능하게 한다. UML 스테레오 타입은 디자이너의 컨트롤을 지원하는데, 이러한 엄격한 DTD는 표준화된 느슨한 DTD의 서브셋이기 때문이다[2]. 그림 7은 UML 클래스 다이어그램을 XML DTD로 변환한 결과이다



(그림 7) UML을 XML DTD로 변환

4. 데이터 모델링

데이터베이스 스키마 변환 과정에서 정보 구조로부터 논리적 개념을 이용하여 어떤 논리적(데이터) 구조로 표현하는 것이 필요한데, 데이터 모델링이란 이 변환과정을 말한다[5].

4.1 변환 방법

그림 3의 클래스 다이어그램을 ORDB 스키마로 변환하는 방법은 다음과 같다[4].

① 객체 클래스는 구조화된 객체 타입을 생성하며, 테이블과 연결하여 타입을 1개 또는 여러 개의 테이블에 객체의 원소가 되도록 한다.

② 객체 타입은 사용자 정의 타입이거나, 속성을 가진 객체 타입이다.

③ 클래스에서의 객체 속성은 객체 타입에서의 속성이다.

④ 타입이 테이블이나 객체 또는 특정 타입으로 변환할 때, 클래스의 객체 속성 타입은 객체 타입의 속성 타입이다.

⑤ 객체 속성이 NULL제약조건을 가지면 NOT NULL 제약조건도 가진다.

⑥ 객체 속성이 초기 값을 가지며 컬럼에 DEFAULT 값을 더한다.

⑦ 독립적인 클래스나 함축적인 성질을 가진 클래스들을 위해 기본키로 정수를 생성하고 {oid}를 위해 태그된 컬럼에 기본키 제약조건에 따라 {oid}를 더한다.

⑧ 부 클래스를 위해 기본키 제약조건과 외래키 제약조건에 따라 각각의 부모 클래스의 키를 더한다.

⑨ 클래스들의 결합을 위해 객체 타입을 생성하고 기본키 제약조건과 외래키 제약조건에 따른 역할 테이블로부터 기본키를 추가한다.

⑩ 교환 oid가 <n>인 경우 UNIQUE 제약조건에 따른 컬럼을 추가한다.

⑪ 각각의 확실한 제약조건을 위해 CHECK를 실시한다.

⑫ 결합하는데 있어서 각각의 0..1, 1..1 역할을 위한 참조 테이블에서 외래키 컬럼을 생성한다. 교대로 단일 객체들이나 컬렉션을 위한 그 자신 안에서 속성으로 객체를 선언하기 위하여 객체 타입에 참조를 사용하거나 또는 다중관계 객체를 위한 참조의 배열로 객체를 선언하기 위하여 객체 타입에 참조를 사용한다.

⑬ 집합 테이블에 외래키를 가진 다중 집합을 위해 기본키를 생성하고, 기본키를 위한 컬럼을 추가하며, 테이블 안에서 그 집합을 저장하기 위한 객체 타입을 사용한다. 또한 단일 객체를 위한 객체 속성 또는 컬렉션을 통하여 배열이나 중첩 테이블을 사용한다.

⑭ 너무 많이 이동하게 되는 이진 결합 클래스들을 최적화한다.

⑮ 외래키를 사용하여 결합이 안된 클래스와 함께 N : N 결합의 관계 테이블을 생성한다.

⑯ N : N 결합에서 역할 테이블의 키로부터 기본키, 외래키의 제약조건을 생성한다.

⑰ 객체 클래스의 전달 작용을 위한 객체 타입에 메소드를 생성한다.

4.2 ORDB 스키마 변환 예

그림 3과 표 1 및 표 2에 의한 UnitOfMeasure 객체는 ORDB 변환 방법 ③, ⑤번의 성질에 따라 UnitOfMeasure_t 객체타입 속성을 저장하는 UnitOf Measure 테이블로 그림 8와 같이 변환된다.

```
SQL> CREATE TYPE UnitOfMeasure_t AS OBJECT
(
  each    string,
  dozen   string
);
SQL> CREATE TABLE UnitOfMeasure OF
       UnitOfMeasure_t
(
  each    NOT NULL,
  dozen   NOT NULL
);
```

(그림 8) UnitOfMeasure 테이블

그림 3과 표 1 및 표 2에 의한 Product 객체는 ORDB 변환 방법 ③, ④, ⑤의 성질에 따라 Product _t 객체타입 속성을 저장하는 Product 테이블로 그림 9과 같이 변환된다.

```
SQL> CREATE TYPE Product_t AS OBJECT
(
  photoURL        string,
  units           UnitOfMeasure_t
);
SQL> CREATE TABLE Product OF Product_t
(
  photoURL        NOT NULL,
  units           NOT NULL
);
```

(그림 9) Product 테이블

그림 3과 표 1 및 표 2에 의한 CatalogItem 객체는 ORDB 변환 방법 ③, ④, ⑤, ⑫의 성질에 따라 CatalogItem_t 객체타입 속성을 저장하며 속성 CatalogItem의 객체타입은 그림 10과 같이 테이블로 변환한다. 배열을 이용한 것은 0..* 관계에서 *는 무한정의 매핑이지만, 실제로 데이터베이스 테이블에 무한정으로 매핑을 할 수가 없다. 그래서, 사용자 정의에 따라 한정된 배열을 지정해 준다.

```
SQL> CREATE TYPE CatalogItem_t AS VARYING ARRAY
(40) OF CatalogItem
(
  name            string,
  listPrice       integer
  );
SQL > CREATE TABLE CatalogItem OF CatalogItem_t
(
  name            NOT NULL,
  listPrice       NOT NULL
);
```

(그림 10) CatalogItem 테이블

그림 3과 표 1 및 표 2에 의한 Catalog 객체는 ORDB 변환 방법 ③, ④, ⑤, ⑧의 성질에 따라 Catalog_t 객체타입 속성을 저장하며 CatalogItem의 객체타입은 그림 10에서 정의한 CatalogItem_t를 적용하고 그림 11과 같이 테이블로 변환한다.

```
SQL> CREATE TYPE Catalog_t AS OBJECT
(
  CatalogID          integer,
  CatalogItemID      integer,
  name               string,
  expireationDate    date
);
SQL> CREATE TABLE Order OF Order_t
(
  PRIMARY KEY        (CatalogID, name),
  CatalogID          REFERENCES Catalog,
  CatalogItem        REFERENCES CatalogItem,
  name               NOT NULL,
  expireationDate    NOT NULL
);
```

(그림 11) Catalog 테이블

5. 결론

XML을 이용하여 상호 교환되는 정보를 체계적이고 안정적으로 저장·관리하기 위해서 XML 응용과 데이터베이스 연계를 위한 다양한 연구가 지금까지 관계형 데이터베이스를 중심으로 수행되었다. 그러나 XML 응용에서 DTD가 다양한 데이터타입을 정의할 수 없는 한계 때문에 데이터베이스에 매끄럽게 연계시키기 어려운 점이 있으며, 또한 계층구조를 갖는 XML 데이터를 2차원 테이블의 집합인 관계형 데이터베이스로의 저장에는 본질적인 한계가 있다.

따라서 본 논문에서는 계층구조를 갖는 XML 데이터를 계층구조를 지원하는 ORDB로 저장이 가능하도록 하는 통합 모델링 방법론을 제안하였다. 이를 위하여 우선적으로 객체지향 설계언어인 UML를 이용해서 클래스 다이어그램을 도출한 후, 클래스 다이어그램에 의해서 XML DTD 생성을 위한 XML 모델링을 제안하였다. 또한, 이들의 모델링으로 교환되는 XML 데이터의 효율적인 저장을 위하여 ORDB 스키마 설계를 위한 데이터 모델링 방법을 제안하였다.

[참고문헌]

[1] Andrew Davidson, Matthew Fuchs, Mette Hedin, Mudia Jain, Jari Koistinen, Chris Lloyd, Murray Maloney, Kelly Schwarzhof, "Schema for Object-Oriented XML 2.0", July, 1999. See http://www.w3.org/TR/NOTE-SOX

[2] Hiroshi Matuyama, Kent Tamura, Naohiko Uramoto, XML and Java Developing Web Applications, Addison Wesley, 1999, pp International Symposium on Industrial Electronics Proceedings, Pusan, Korea, June 2001

[3] http://www.xml.com/lpt/a/98/10/guidel.html.

[4] R.G.G.Cattell, Douglas K. Barry, The Object Database Standard : ODMG 2.0, Morgan Kaufmann Publishers, Inc, 1997.

[5] Robert J. Muller, Database Design for Smarties: Using UML for Data Modeling, Morgan Kaufmann Publishers, Inc, 1999.

[6] Shanmugasundaram, J., Shekita, E. J., Bar, R., Carey, M. J., Lindsay, B. G., Pirahesh, H., Reinwald, B.,: Efficiency Publishing Relational Data as XML Documents. In Abbadi, A. E., Brodie, M. L., Chakravarthy, S., Dayal, U., Kamel, N., Schlageter, G., Whang(eds.), K. Y., VLDB2000-Proceedings of the 26th International Conference on Very Large Data Bases, September 10-14, 2000, Cairo, Egypt, Morgan Kaufmann Publishers 2000.

[7] 임춘봉; 신인철; 심재철 공저, UML사용자 지침서, 도서출판 인터비전, 1999.