

UML을 이용한 XML Editor의 설계 및 구현

신명섭, 김진한
충주대학교 전자계산학과

The Design and Implementation of XML Editor with UML

Myung-Sub Shin, Jin-Han Kim.
Dept. of Computer Science, Chung-Ju Nat' l University
E-mail : freemsshin@hanmir.com, jhkim@gukwon.chungju.ac.kr

요 약

웹 문서의 표현 언어로 XML이 대두되면서부터 신속히 애플리케이션을 위한 문서를 제작하는 것이 중요해졌다. 본 논문에서는 UML 쓰임새 중심 방법을 적용하여 XML 문서 편집기를 제작하였다. 쓰임새 방법의 단계는 해당 문제의 범위에서 객체를 찾아내는 도메인 모델링 단계, 시스템의 행동을 식별해내는 쓰임새 모델링 단계, 쓰임새의 정제와 새로운 객체를 찾아내는 계층 모델링 단계, 그리고 실제로 어떻게 구현할 것인가를 설계하는 교류 모델링으로 나누어 DTD 편집기와 XML 편집기를 설계하고 구현하였다.

1. 서론

현재 인터넷에서 사용되는 전자 문서는 대부분 HTML(Hyper Text Markup Language)로 표현된다. 그런데 HTML은 하이퍼텍스트 형태의 웹 문서를 표현하는데는 적합하지만, 개발자들이 임의로 태그를 확장할 수 없어 기술되는 문서의 구조가 단순해지기 때문에 다양한 문서 구조를 갖는 웹 응용에는 사용하기가 쉽지 않다.[8]

이러한 HTML의 한계를 극복하기 위해 새로운 대안으로 1996년 W3C(World Wide Web Consortium)에서는 웹 전자 문서 기술에 대한 표준 언어로 XML(Extensible Markup Language)을 제정하였다.[1] XML은 웹 환경에 맞도록 많이 사용되지 않는 SGML의 일부 기능을 제거하면서도 SGML이 가지고 있는 확장성(Extensibility), 구조(Structure), 검증(Validation)의 특성을 그대로 계승하였다.[8] 따라서 XML로 웹 문서를 표현하면 문서 자체에 대해서 사용자가 임의의 태그를 사용하여 문서를 확장하기 쉽고 문서에 대한 검증까지 할 수 있어서 앞으로의 웹 환경에서 표준으로 대두되고 있다. 이러한 환경에서 XML Application-전자 상거래, EDI 등-을 작성하고 업무를 수행하는데 있어서 사용자가 XML 문서를 얼마나 빨리 만들고, 얼마나 편리하게 작성할 수 있는가

가 중요해진다. 따라서 본 논문에서는 XML 문서 작성기를 설계하고 구현하여 XML 문서를 빠르고 간편하게 작성할 수 있게 했다. 또한 XML 문서는 작성하기가 어려워서 전문가만이 작성할 수 있는데, 이러한 문제를 해결하기 위해 간편한 인터페이스와 설명으로 개발자뿐만 아니라 일반 사용자에게도 쉽게 적용될 수 있도록 했고 개발 과정에 UML(Unified Modeling Language) 개념을 도입하여 시스템에 대해 완전성을 추구하고자 했다. 본 논문은 다음과 같이 구성된다. 2장에서는 XML에 대해 간단히 소개하고 3장에서는 UML 개발 방법론 중 쓰임새 방법에 대해서 간단히 소개하고 4장에서는 XML 문서작성기의 설계 및 구현을 보이고 마지막으로 5장에서는 결론 및 향후 계획을 기술한다.

2. XML

XML의 문서구조는 크게 7가지로 나누어지는데 첫 번째, XML은 문서가 만들어지기 위한 기초 스펙으로 문서가 따라야 할 문법을 정의하며 두 번째, DTD(Document Type Definition)와 Schema는 문서 형태를 구성하게 하는 방법을 제공하고, 세 번째, Namespaces는 XML 어휘집을 다른 어휘집과 구별할 수 있도록 하고, 네 번째, XPath는 문서의 주소 지정

을 위한 질의어를 제공하고, 다섯 번째, XSL은 문서를 다른 문서로 변환하는 방법을 제공하며, 여섯 번째, XLink와 XPointer는 문서와 문서 또는 문서와 문서의 일부분에 대한 링크를 제공하고, 일곱 번째, DOM(Document Object Model)은 애플리케이션 코드와의 연결을 제공한다.[3] 본 논문에서는 XML과 DTD 부분만을 다루었다.

2.1. XML의 구성

XML의 구성은 크게 문서 선언부, 문서 구조 정의부, 실제 문서부 이렇게 3가지로 구성된다.

2.1.1. 문서 선언부

문서 선언부는 “<?xml version=“1.0” encoding=“EUC-KR” standalone=“yes”?”과 같이 표현되며 반드시 문서의 첫 번째 줄에 위치하여야 한다. 문서 선언부에는 XML Spec의 버전과 문자를 표현하는 encoding 방식과 이 문서가 특정 DTD에 의존하는가, 하지 않는가를 나타내는 standalone 등으로 구성된다.[7][1]

2.2.2. 문서 정의부

문서 구조 정의부는 DTD로서 실제로 문서가 구성어지는 구조에 관한 규칙들을 가지고 있다. DTD는 외부 DTD를 참조하는 방식과 내부 DTD를 참조하는 방식이 있으며, 그 구성으로는 문서 구조를 나타내는 Element 선언부, Element에 대한 속성을 선언하는 Attribute 선언부, 문서 안에서 특정 문자나 자주 사용하는 문장 등을 선언하는 Entity 선언부, 그리고 비 XML 데이터를 처리할 때 XML Application을 위한 정보로 사용되는 Notation 선언부로 구성된다. 실제 문서부는 DTD에서 정의된 문서 구조대로 작성된 문서 내용으로 구성되어 있다.[7][1]

2.2.3. 실제 문서부

실질적으로 마크업을 가지고 문서를 표현하는 부분이다. 모든 태그는 시작태그와 종료태그가 있어야 하고 ‘<’와 ‘>’ 안에 이름이 삽입된다. 종료태그는 ‘>’문자 앞에 ‘/’문자를 붙인다. 실제 문서부는 실제 태그인 Element, XML 문서 안에 특정 문자나 자주 사용하는 문장 등을 사용할 수 있게 해주는 Entity, 특정 애플리케이션에게 데이터를 전달해주는 PI(Processing Instruction), 그리고 XML 문서 안에서 파싱되는 문

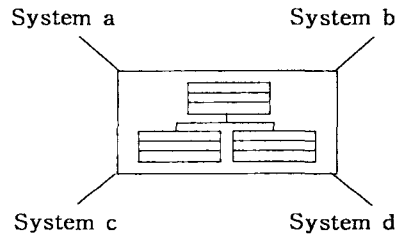
자까지 표현할 수 있도록 해주는 CDATA로 구성된다. 실제 문서부는 DTD의 정의에 따라 기술되어진 Valid 문서와 DTD 없이 XML 구분 규칙만을 적용하여 작성된 Well-formed XML 문서가 있다.[7][1]

3. 쓰임새 개발 방법론

시스템을 모델링하는데 UML(Unified Modeling Language)을 적용하여 개발하는 경우는 시스템을 개발하는 경우 복잡할 뿐만 아니라 비용과 시간, 인적 자원 등도 많이 소모되어지기 쉽다. 그래서 RUP(Rational Unified Processing)에서 꼭 필요한 프로세스만을 적용하여 시스템을 모델링하는 것을 쓰임새 개발 방법론이라 한다. 쓰임새 개발 방법론의 단계는 크게 도메인 모델링, 쓰임새 모델링, 계층 분석, 교류 모델링으로 나누어진다.[2]

3.1. 도메인 모델링

시스템이 해결해야 할 범위를 문제 도메인(Problem Domain)이라 하는데 도메인 모델링은 이러한 문제 도메인에서 처리되어야 할 객체 즉, 클래스들을 찾아내는 것을 말하고 쓰임새 방법에서 도메인 모델링은 자료 요구사항으로부터 시작하여 밖으로 진행하면서 제안 시스템에 대한 문제 영역의 정적 모델을 완성해나가는 작업을 말한다. 작업을 진행하면서 클래스들을 찾아내고 모든 클래스를 찾아냈으면 클래스에 대한 일반화와 집합관계 등을 식별한다.[2][4][5][6]



[그림1] Inside-Out

3.2. 쓰임새 모델링

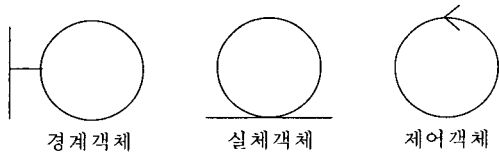
쓰임새 모델링은 새로운 시스템에 대한 사용자 요구 사항 즉, 시스템 행동을 파악하는 작업이다. 쓰임새를 식별하는 방법은 GUI에서 작업하여 쓰임새를 식별하는 방법과 사용자 매뉴얼에서 작업하여 식별하는 방법이 있다. 본 논문에서는 시스템을 새로 작성하므로 GUI에서 작업하여 쓰임새들을 식별하였다. 쓰임새를 식별한 후 쓰임새별로 정상 조건에서 수행

되는 동작을 나타내는 기본흐름과 오류가 발생했을 경우를 나타내는 대안흐름을 작성한다. 쓰임새에는 발생할 수 있는 모든 동작들을 쓰임새에 표현해야 하고 쓰임새 본문과 계층 분석에서 다룰 계층도와 일치할 때까지 계속 반복 정제한다. 그런 다음 공통 쓰임새는 포함(Include), 확장(Extend), 그리고 일반화(Generalization)를 적용하여 분할한 다음 쓰임새별로 패키징화 한다.[2][4][5]

3.3. 계층 분석(Robustness Analysis)

계층 분석은 쓰임새의 본문을 분석하여 쓰임새에 참여하는 객체들을 행위자와 교류하는 경계 객체(Boundary Object), 시스템의 특정 행동 이후에도 계속 정보를 유지하는 실체 객체(Entity Object), 그리고 쓰임새에서 필요한 기능과 시스템의 행동을 나타내는 제어 객체(Control Object)로 구분하는 작업이다. 계층 분석의 장점은 분석 단계와 설계 단계 사이의 틈새를 없게 해준다.[2][5]

계층 분석에서는 [그림2]와 같은 3가지 객체를 사용하며 계층도를 그릴 때는 [그림3]과 같은 규칙을 사용한다.

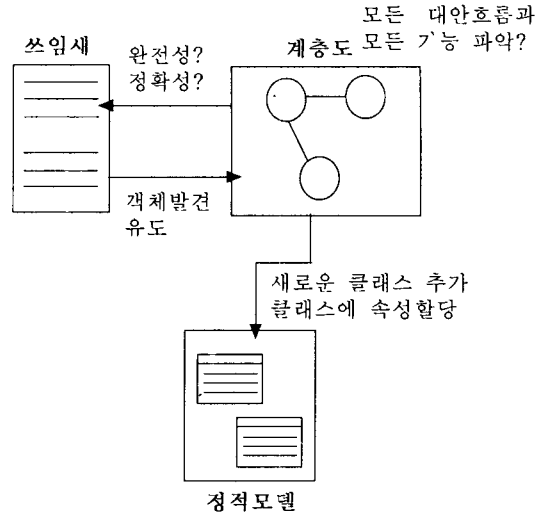


[그림2] 계층도에서의 3가지 객체

- ① 행위자는 경계 객체에게만 말할 수 있다.
- ② 경계 객체는 제어객체와 행위자에게만 말할 수 있다.
- ③ 제어객체는 경계객체와 실체객체, 그리고 다른 제어객체에게 말할 수 있으나, 행위자에게는 말할 수 없다.
- ④ 실체 객체는 제어객체에게만 말할 수 있다.

[그림3] 계층도 작성 규칙

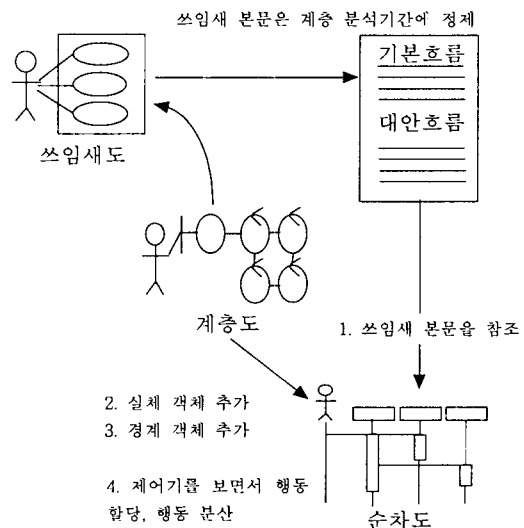
계층 분석이 끝나면 새로운 클래스를 도메인 모델에 추가하고, 모든 대안 흐름과 기능들이 파악되었는지 확인한다. 만약에 이것들이 모두 파악되지 않았다면 다시 쓰임새로 돌아가서 새로운 객체를 유도한다.[그림3][2]



[그림4] 계층 분석 피드백

3.4. 교류 모델링

교류 모델링은 실제로 시스템을 어떻게 구현할 건지에 대한 설계를 하는 단계이다.[2] 객체들에게 행동을 할당하고 객체 사이에서 일어나는 교류들을 나타내고, 클래스간의 연산을 분산시키는 작업을 한다. 교류 모델링에서 사용되는 것은 순차도와 협력도가 있는데 본 논문에서는 순차도만을 다루어 교류 모델링을 다루었다. 과정은 [그림5]와 같다.



[그림5] 순차도 구축하기

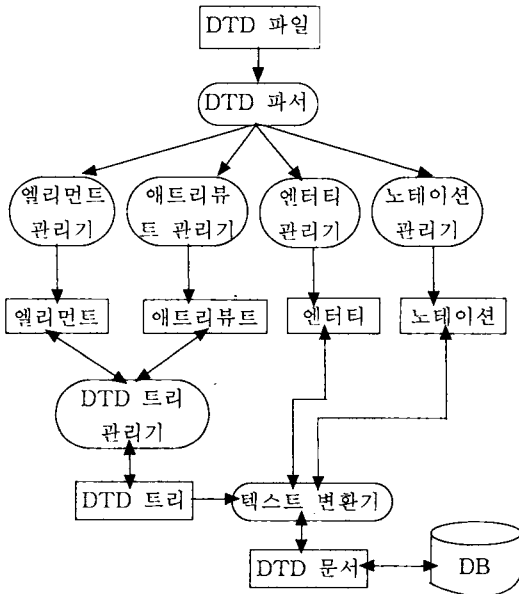
[2][4][5]

4. XML 문서 편집기의 설계 구현

본 논문에서의 XML 문서 편집기는 DTD 편집기와 XML 편집기로 나누어진다. 설계에서 사용한 도구는 Rational Rose 2000이며 구현은 비주얼 베이직을 사용하였다.

4.1. DTD 편집기의 설계

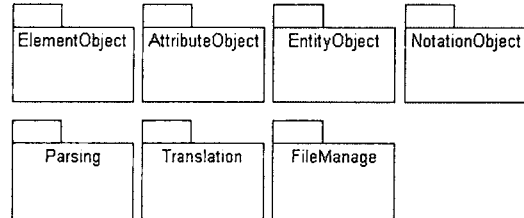
다음은 DTD 편집기의 전체 구성 및 처리 과정이다.



[그림6] DTD Editor 구성 및 처리과정

먼저 DTD 파일을 DTD 파서가 파싱을 하면 각 관리기들이 해당 객체를 추출해서 객체를 생성하고 엘리먼트와 에트리뷰트는 DTD 트리를 구성하여 구조 편집기에 표시되고 DTD 트리의 내용과 함께 엔터티, 노테이션 객체를 이용하여 텍스트 편집기에 DTD 문서를 표시한다.

본 논문에서는 [그림6]에 대한 처리과정을 쓰임새 방법으로 구성하였다. 먼저 도메인 모델링에서는 [그림6]의 처리과정에 참여하거나 관계 있는 객체들을 찾아내어 분석 클래스도를 구성하였다.



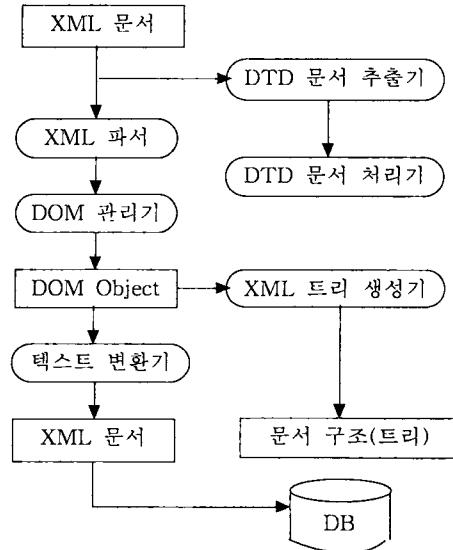
[그림7] DTD 편집기의 쓰임새 패키지

쓰임새 모델링에서는 시스템의 행동들을 찾아내어 그 행동에 대해 기술하였고 [그림7]과 같이 7개의 패키지로 나누어서 관리한다.

계층 분석에서는 작성되어진 쓰임새들을 유도 분석하여 새로운 클래스들을 찾아내어 클래스도에 추가시켰고 관련 쓰임새가 더욱 정확하고 완전하도록 하였다. 마지막으로 교류 모델링에서는 계층분석의 결과물인 계층도와 클래스도를 가지고 순차도를 이용하여 어떤 객체가 어떤 행동을 할 것인가에 대한 상세한 설계를 하였다.

4.2. XML 편집기의 설계

다음은 XML 편집기의 전체 구성 및 처리과정이다.

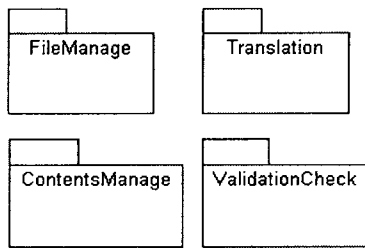


[그림8] XML Editor 구성 및 처리과정

XML 문서가 들어오면 먼저 DTD 문서 추출기가 해당 문서에서 DTD에 대한 내용을 추출하여 처리하고 DTD 문서의 처리과정은 중복되므로 여기서는 생략하였다. XML 파서가 문서에 대한 유효성 검사를 한 후

XML 트리 생성기가 DOM 객체를 이용하여 문서의 구조를 만들고 텍스트 변환기는 DOM 객체를 이용하여 XML 문서로 변환하여 준다.

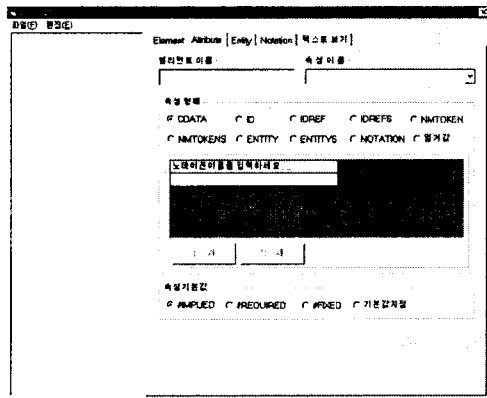
[그림8]에 대한 처리과정에 도메인 모델링을 적용하여 처리과정에 참여하거나 관계 있는 개체들을 찾아내어 분석 클래스도들을 구성하였다. 쓰임새 모델링에서는 시스템 행동을 찾아내어 그 행동에 대해 기술하였고 [그림8]에 대한 쓰임새를 4개의 패키지로 나누어서 관리한다.[그림9]



[그림9] XML 편집기의 쓰임새 패키지

계층 분석에서는 작성되어진 쓰임새들을 유도 분석하여 새로운 클래스들을 찾아내어 클래스도에 추가시켰고 관련 쓰임새가 더욱 정확하고 완전하도록 하였다. 마지막으로 교류 모델링에서는 계층분석의 결과물인 계층도와 클래스도를 가지고 순차도를 이용하여 어떤 객체가 어떤 행동을 할 것인가에 대한 상세한 설계를 하였다.

4.3. DTD 편집기의 구현



[그림10] DTD 편집기 메인 화면

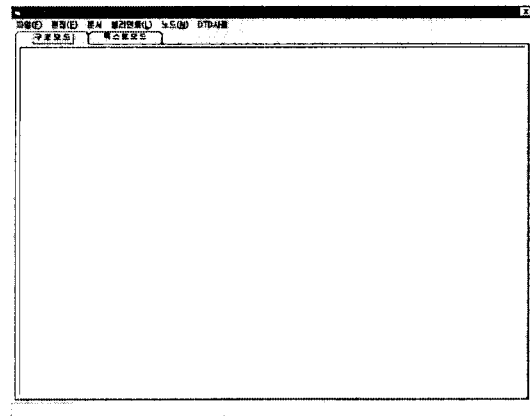
DTD 편집기는 여섯 부분으로 구성된다. 화면 왼쪽은

DTD의 구조를 나타내는 창이고 오른쪽은 각각 엘리먼트 창, 애트리뷰트 창, 엔터티 창, 노테이션 창, 그리고 텍스트보기 창으로 구성되어진다.

엘리먼트 창에서는 엘리먼트 이름을 직접 입력하여 지정하게 하였고 content model을 직접 창에 표시하므로 사용자가 쉽게 구조를 알 수 있도록 하였다.

애트리뷰트 창에서는 먼저 사용자가 엘리먼트 이름을 선택하고 그 다음 속성을 입력하도록 하였다. 속성 형태는 열거값, NOTATION 형태도 입력할 수 있도록 하였다. 엔터티와 노테이션 창에서는 엔터티와 노테이션을 테이블형태로 관리한다. 엔터티 테이블은 엔터티 종류, 이름, 대체 텍스트, PublicID, SystemID, Location1,2, NDATA, Notation 형태, 조건부 선택 등으로 구성되어 있으며 엔터티의 종류에 따라서 나머지 항목들의 입력 여부가 결정된다. 노테이션 테이블은 노테이션이름, PublicID, SystemID, Location1,2로 구성되어진다. 텍스트보기 창에서는 사용자가 구조 편집기에서 입력한 내용을 텍스트 형태로 보여주고 편집도 가능하고 텍스트 모드와 구조 모드를 상호연동하여 각각의 수정내용을 적용할 수 있도록 하였다.

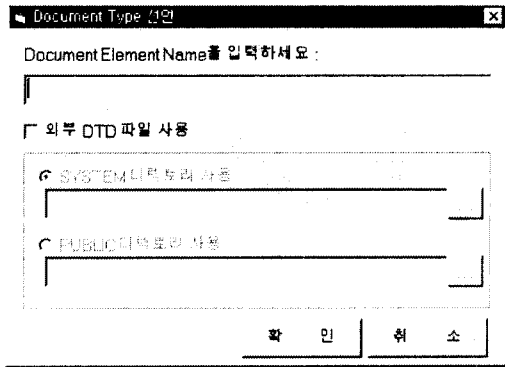
4.4. XML 편집기의 구현



[그림11] XML 편집기의 메인 화면

XML 편집기는 구조편집기와 텍스트 편집기로 구성되어진다. 텍스트 모드에서는 XML 문서를 텍스트로 표시하였고, 구조편집기에서는 XML 문서를 트리형태로 표시하고 구조편집기내에서도 요소나 속성들을 편집할 수 있도록 하였다. 또한 사용자가 DTD 문서를 사용함에 있어서 내부 DTD를 사용할 것인가 외부 DTD를 사용할 것인가를 지정할 수 있게 하였고

XML 편집기 내에 DTD 문서를 편집할 수 있는 기능들을 포함시켰다. [그림12][그림13]



[그림12] Document Type Declaration

문서	엘리먼트(E)	노드(N)	DTD사용
Document Type 선언			
	엘리먼트 입력		
	애트리뷰트 입력		
	엔터티 입력		
	노태이션 입력		
	엘리먼트 수정		
	애트리뷰트 수정		
	엔터티 수정		
	노태이션 수정		
	엘리먼트 삭제		
	애트리뷰트 삭제		
	엔터티 삭제		
	노태이션 삭제		

[그림13] 엘리먼트 메뉴

엘리먼트(E)	노드(N)	DTD사용
스트림모드		
	노드 입력	
	속성 입력	
	텍스트 입력	
	CDATA 입력	
	Comment 입력	
	PI 입력	
	노드 수정	
	속성 수정	
	텍스트 수정	
	CDATA 수정	
	Comment 수정	
	PI 수정	
	노드 삭제	
	속성 삭제	
	텍스트 삭제	
	CDATA 삭제	
	Comment 삭제	
	PI 삭제	

[그림14] 노드 메뉴

또한 XML 문서의 각각의 요소들을 분리하여 관리하고 유효성검사에서는 일반적인 검사뿐만 아니라 ID 참조, Entity 참조 등을 검사 할 수 있도록 했다.

5. 결론 및 향후 연구 방향

본 논문에서는 XML 편집기 중 DTD 편집기와 XML 편집기를 설계·구현하였다. XML 지원 편집기는 많이 제시되었지만 어떻게 개발되었는지 그 방법을 제시하는 편집기는 없다. 본 논문에서는 소프트웨어 개발 방법론 중 쓰임새 방법론을 적용하여 그 방법을 제시하였고 설계 및 구현하였다. 본 논문에서 개발한 DTD, XML 편집기는 GUI가 너무 단순하다는 것이 단점이다. 그러나 DTD 편집기에서는 DTD 문서에서 표현할 수 있는 모든 것을 다루었다. 좀 더 확장하여 XML 편집기와 연동을 한다면 XML 편집기에서는 DTD 처리 부분을 구현하지 않아도 된다. XML 편집기는 쓰임새 방법론을 적용한 결과 XML 문서의 각 부분들을 독립적인 객체로 분리하여 개발 시에도 그대로 적용했기 때문에 좀 불편할 수도 있다. 그러나 이렇게 각 객체들을 분리하여 구현해서 XML에 대한 좋은 교육 도구가 될 수도 있다. XML 편집기를 확장하여 XSLT와 XPath도 지원한다면 더 유용한 도구가 될 것이다.

[참고문헌]

- [1] <http://www.w3.org/TR/2000/REC-xml-20001006>
- [2] Doug Rosenberg with Kendall Scott, XML, Object Modeling - Centered Use case, 인터비전, 2000.3.
- [3] David Hunter, Beginning XML, 정보문화사, 2000.10
- [4] Grady Booch, James Rumbaugh, Ivar Jacobson, The Unified Modeling Language User Guide, 인터비전, 2000.10
- [5] Object-Oriented Software Engineering : A Use Case Driven Approach, Addison-Wesley, 1992
- [6] Object-Oriented Analysis and Design with Application, Grady Booch, 1994
- [7] 최지태, 전영훈, 황대훈, DTD 생성 기능을 지원하는 SGML/XML 문서 병용 편집기의 설계 및 구현, 434, 1999
- [8] 김은옥, 황종욱, 박인호, 강현석, XODT : XML DTD 설계를 위한 객체 다이어그램 기법, 460, 1999