Java 프로그램의 정보공유를 위한 XML DTD 설계

장근실*, 유철중**, 장옥배**
* 광양대학 전자계산과, ** 전북대학교 컴퓨터과학과
bigjang@chollian.net

The Design of XML DTD for Information Sharing of Java Program

Gun-Sil Jang*, Cheol-Jung Yoo**, Ok-Bae Chang**
* Dept. of Computer Science, Kwangyang College
** Dept. of Computer Science, Chonbuk National Univerity

요 약

개발환경이 고정된 장소에서 분산된 장소로 점차 변함에 따라서 프로젝트나 프로그래밍에 관련된 주변인들 사이에 발생할 수 있는 정보 공유와 교환이 어려워지고 있다. 인터넷과 인트라넷의 급격한 증가로 인해 개발자들은 분산된 환경에서 작업을 수행하는 일이 많아지면서 기존의 HTML을 이용한 문서의 공유나 교환은 HTML의 제한점들로 인해 많은 부담이 된다.

본 논문에서는 분산환경에서의 정보 공유와 교환을 위해 HTML의 제한점들을 보안하여 확장성이 뛰어나고, 사용자 정의 태그를 지원하며, 문서의 논리적인 정보를 제공할 수 있는 XML을 이용하여 급격한 사용자 증가를 얻고 있는 Java 프로그램을 대상으로 하는 정보 공유와 교환에 필요한 정보들을 각 모듈별로 제안하고, 제안된 정보에 대한 XML DTD를 설계한다.

1. 서론

소프트웨어 시스템이 점차 복잡해지고 대형화되어 가고 있다 하드웨어의 급속한 발달 추세를 따라가지 못하는 소프트웨이의 진보는 소프트웨어 개발자나 관리자 및 공학자 등을 비롯한 관련인들 모두에게 커다란 부담이 되고 있다[1]. 또한 개발환경의 변화는 기존의 사무실 환경에서 네트워크와 인터넷의 급속한 증가에 힘입어 점차 분산화되어 가는 추세이고, 이와 더불어 발생하는 다양한 문제점들 역시 유발되고 있다. 개발팀 원들이나 시스템 분석가나 유지보수자 등의 측면에서 정보 공유 역시 커다란 문제로 나타나고 있다

분산된 개발 환경은 인트라넷이나 인터넷을 기반으로 하여 서로의 정보를 HTML(Hyper Text Markup Language)을 이용하여 표현하고 공유하고 있다[2]. 개인적인 사용자 측면에서 HTML의 사용이나 학습의 용이성은 충분히 매력적이고, 이와 같은 용이성들이 인터넷의 발달과 보급에 커다란 영향을 미쳤다 하지만, 위에서 언급한 개발자들이나 유지보수자 등의 측면에서는 HTML을 이용한 소프트웨어 시스템의 정보 공유는 그리 큰 매력이 되지 못한다. 그 이유는 HTML이 갖는 제한사항들 때문인데, 사용자 정의형 태그를 지원하지 못한다는 것과 자료의 구조적인 정보를 제시할 수 없다는데 있다

본 논문에서는 1990년대 중반부터 엄청난 이용자 층을 확보하고 있는 Java를 기반으로 하는 소프트웨어 시스템에 있어서 효과적이고 효율적인 정보 공유를 위한 방법을 XML(eXtended Markup Language)을 중심으로 알아본다. XML은 HTML이 갖는 장점들은 수용하고 문제점과 SGML(Standard Generalized Markup Language)의 문제점인 학습 및 이용의 어려움을 해결한 마크업 언어로 사용자 정의 태그와 구조적인 정보의 표현 및 확장성이 용이하다[2,3,4].

2장에서는 XML에 대한 개괄적인 내용과 HTML 및 SGML과의 비교분석을 제시하고, 본 연구와 비슷한 관련연구들을 기술한다. Java는 C++와는 다리 최소한 하나의 클래스가 필요한 순수 객체지향 언어이다. 그러므로 Java로 구현하는 소프트웨어 시스템의 구조적인 구성은 클래스와 클래스를 구성하는 메소드 및 어트리뷰트들과 이들 사이의 관계성이라 할 수 있다. 그래서 3장과 4장에서는 Java 원시 코드의 각 모듈에 대해서 필요한 정보들을 제안하고, 이 정보들을 대상으로 XML DTD(Document Type Definition)를 설계한다. 5장에서는 결론 및 앞으로의 연구 분야에 대해서 언급을 한다.

2. 관련 연구

2.1 XML과 HTML 및 SGML의 관계

1996년 W3C(World Wide Web Consortium)에서 제안한 XML은 많은 관심을 끌고 있다 그 이유는 기존의 마크업 언어인 HTML을 이용한 인터넷 기반의 정보 표현과 공유에 대하여 사용자들이 느끼는 여러 가지 제약사항들로부터 기인한다 HTML은 인터넷의 보편화에 지대한 영향을 미쳤지만, 사용상에 있어서의 한계점들(사용자 정의형 태그를 지원하지 못하고, 제한된 태그만을 이용하기 때문에 구조적인 정보를 표현할 수 없다)로 인해 여러 가지 다양한 주변기술들의 개발을 유도하게 했다 XML 역시 그 영향을 받은 부산물 중의 하나이다. 1986

년 ISO8879에 정의된 SGML은 XML이나 HTML의 개발과 발전에 뿌리가 되는 마크업 언어로 실제로 이들 사이에는 수직적인 관계가 존재한다. SGML은 XML과 HTML 생성에 있어 이론적이고, 실제적인 측면에서 슈퍼셋(superset)의 역할을 한다 하지만, 좀 더 자세한 구분에 의하면 HTML은 SGML의 어플리케이션이고, XML은 SGML의 정밀한 부분집합이라고 할 수 있다 이 말은 HTML은 SGML의 확장성인 사용자 정의 태그의 지원을 허용할 수 없지만, XML은 SGML의 확장성을 지원한다는 것이다[4,6]. 그러므로 XML을 다른 마크업 언어를 정의할 수 있는 메타언어 (meta-language)라고 할 수 있다[5].

SGML은 커다란 문제점인 사용상의 어려움과 복잡성으로 인해 많은 사용자층을 확보하는데 큰 성과를 얻지 못했다. 반면에 XML은 확장성으로 인해 많은 사용자층을 확보할 수 있게 되었다. XML은 정보 구조와 내용의 기술을 표현과는 다르게 구분하는데, 데이터 구조와 구문을 정의할 때는 DTD를 이용하고, 표현에 관련된 내용들은 XSL(XML Style Language)를 이용하여 기술한다[7].

2.2 관련 연구

XML을 이용한 정보 제공 및 공유 방법에 대한 연구는 다양한 응용 분야를 중심으로 연구가 되어 왔고, 그 중에는 실제로 이용되고 있는 분야도 있다.

[2]의 UXF(UML eXchange Format)는 UML(Unified Modeling Language)[8]을 표현할 수 있도록 개발한 XML의 서브셋으로 분산된 장소에서의 정보 공유 및 UML 문서에 포함되어 있는 논리적인 정보를 표현 할 수 있도록 개발되었다. 현재 버전은 UML의 의미정보만을 표현할 수 있고, 다이어그램의 위치 정보나 도형 정보는 표현할 수 없다

이 외에도 화학식을 표현하도록 개발된 CML(Chemical Markup Language), 수학식을 지원할 수 있도록 개발된 MathML(Mathematical Markup Language)[9], 푸쉬 기술에 적용되는 CDF(Channel Data Format), 경영 및 재정과 관련된 정보를 표현할 수 있도록 개발된 OFX(Open Financial eXchange) [10] 등이 있다.

3. Java 원시 프로그램 모듈 정보

Java는 객체지향 언어이지만, C++와는 다른 점들이 많다. 그 중에서 객체지향 측면에서 볼 때 가장 큰 차이점은 클래스의 존재 여부이다. C++는 기존의 언어인 C에다 객체지향 특성을 추가시킨 언어이므로 프로그램이나 프로젝트의 상황에 따라 클래스가 존재하지 않을 수도 있다. 이에 반해 Java는 반드시 하나 이상의 클래스를 포함한다.

본 연구에서는 Java를 구성하는 주요 구성요소를 위와 같은 특징으로 인해 클래스와 클래스의 어트리뷰트 및 메소드로 구분한다 그리고 여기에 클래스들의 관계성을 추가한다. 기존의 Javadoc[11]와 같은 문서화 도구는 Java 원시 코드 내부에 필요한 정보를 시스템에서 지원하는 특정 플래그를 지정하여 정의하였다 그리고, 이렇게 플래그화된 정보를 필요에 따라서 관련 HTML 파일로 생성해준다. 더불어 근래에 제작되거나 배포되는 대부분의 소프트웨어 시스템들은 사용자 지침서나 관련 정보를 HTML 파일의 형식을 빌어 제공하고 있다.

하지만, 이와 같은 HTML 파일들의 경우는 제공자가 제공하는 단 하나의 형식으로만 정보를 공유할 수 있다. 표현하는 정보의 내용은 동일하지만, 레이아웃이나 디스플레이와 같이 형식을 달리하는 경우에는 전혀 새로운 문서를 만들어야만 하는 상황이 된다.

일반적으로 원시 코드 내에 기술되는 정보들은 주석의 형식을 빌어 제공되며, 주석으로 제공되는 정보들은 원시 코드에서 추출 가능한 정보(comment)와 추출 불가능한 정보(annotation information)로 나눌 수 있다. 추출불가능한 정보는 실제로 원시코드를 읽고 이해할 때 많은 정보를 제공한다.

본 논문에서 제안하는 각 모듈의 정보들은 [12]에서 제안한 정보들을 중심으로 한다.

3.1 클래스 관련 정보

클래스에 대한 정보의 경우 <표 3-1>의 좌측과 같다. 표에서 알 수 있듯이 왼쪽 부분인 Comment Information 부분은 원시 프로그램에서 추출 가능한 정보이다. 클래스의 "종류"는 추상 클래스인지 가상 클래스인지 등의 정보를 의미하고, 클래스의 "엑세스 모드"는 public인지 private인지 등의 정보를 의미한다. 실제 중요한 정보들은 표의 오른쪽 부분인 Annotation Information에 나타나는데, 클래스가 어떠한 목적으로 기능을 수행하는지를 기술하는 정보나 클래스의 전제조건이나 제한사항, 예외처리 등의 정보가 실제 원시 프로그램을 이해할 때 많은 시간이 소비되는 부분이기 때문이다.

Java API에 존재하는 인터페이스나 클래스와는 달리 개발자가 작성한 클래스의 경우 위에 나타나는 다양한 정보들은 해당 클래스의 이해와 재사용에서 이용자에게 커다란 도움을 준다.

3.2 메소드 정보

Class 정보		Method 정보		Data Member 정보	
Comment	Annotation	Comment	Annotation	Comment	Annotation
클래스 이름 인스턴스 이름 상위 클래스 이름 하위 클래스 이름 인터페이스 이름 클래스의 종류 클래스의 엑세스 모드	클래스 기능 전제조건 제한사항 예외처리 개발자 정보	메소드 이름 메소드의 종류 메소드의 스코프 메소드의 엑세스 모드 메소드의 반환값 메소드의 각 인수들 호출 메소드 이름 호출 메소드 클래스 이름 호출 메소드의 객체 이름 피호출 메소드 이름 피호출 메소드의 클래스 이름 피호출 메소드의 객체 이름	메소드 기능 전제조건 제한사항 예외처리 각 인수들의 기능 반환 값의 기능	데이터 멤버 (일반 변수나 객체) 데이터 멤버 명의 클래스	데이터 멤버의 기능 특정 블록의 기능 행의 기능

<표 3-1> 요소들의 정보

메소드에 대한 정보는 <표 3-1>의 가운데 부분과 같으며, 이들 중 유의해야 할 것들은 메소드들의 관계에 대한 정보이다. 이는 여러 개의 클래스들과 그들의 다양한 객체들 사이의 관련성을 갖는 프로그램의 경우 메소드들 사이의 호출관계는 사용자들에게 혼란을 줄 수 있기 때문이다[9] XML을 이용한 정보 공유의 목적이 코드의 이해에 필요한 정보를 제공하는 것이기 때문에 현재의 메소드와 호출이나 피호출 관계에 있는 다른 메소드들에 대한 정보들을 제시하는 것은 사용자들에게 많은 도움이 된다.

<표 3-1>의 Annotation Information 부분 중에서 "전제조건"은 메소드의 실행을 위해 필요한 정보들을 제시하는 부분이고, "제한사항"의 경우는 메소드의 실행에 대해 이용자가 알고 있어야 할 사항들을 제시하는 부분이다

3.3 데이터 멤버

데이터 멤버에 대한 정보들은 <표 3-1>의 우측과 같다. 제어문의 경우 데이터 멤버나 변수의 값에 의해 제어의 흐름이 많은 영향을 받으므로 주석에 기술된 데이터 멤버나 기타 다른 변수들의 원시 프로그램 내부에서의 기능은 코드의 이해에 많은 도움이 된다. 또한 개발자 입장에서 특히 중요하게 다루어야 할 부분은 특정 "블록"이나 "행"의 기능에 대한 부분이다. 특정 블록의 경우는 개발자의 알고리즘이나 쉽게 이해하기 어려운 테크닉 등이 나타나는 부분이고, 이로 인해 코드를 이해하려는 이용자의 경우, 심지어 개발자 자신까지도, 많은 시간이 흐른 뒤의 유지보수나 이해에서 어려움을 느낄 수 있기 때문에 서술적인 표현을 이용한 정보의 기술이 필요하다.

4. DTD 설계

DTD(Document Type Definition)란 이름에서 알 수 있듯이 만들고자 하는 문서에 표현할 정보들의 논리적인 구조를 표현하는 것이다. 즉, 사용자 정의 태그(User Definition Tag)를 만들고, 태그에 필요한 선택사항들을 기술하는 일종의 구문(syntax) 정의라고 할 수 있다.

하지만, 이처럼 DTD에는 이용할 수 있는 논리 구조만 포함될 뿐 XML 문서의 디스플레이 형식이나 레이아웃 형식과 같은 포매팅(formatting) 정보는 포함되어 있지 않다 그래서 동일한 문서 정보의 다양한 포맷을 지원하기 위해서는 스타일 쉬트(style sheet)를 이용해야 한다. 스타일 쉬트란 출력 포맷을 지정하는 순서를 의미한다.

표 <4-1>은 Java 프로그램의 모듈 중에서 클래스 DTD에 대한 내용 중의 일부로, 3장에서 제시한 각각의 모듈을 기초로 하여 작성된 것이다. 이 부분은 지속적이고도 많은 Java 프로젝트들을 대상으로 더 많은 정보를 얻어내고 Java DTD에 적용시켜야 한다

```
<!ATTLIST     Class
  NAME        CDATA  #REQUIRED
  ABSTRACT    (true|false)        "false"
  VISIBILITY  (public|private)    #REQUIRED
  ACTIVE      (true|false)        #IMPLIED>
<!ELEMENT     Attribute (Note*)>
<!ATTLIST     Attribute
  VISIBILITY  (public|protected|private)
                                  #REQUIRED
  TYPE        CDATA #REQUIRED
  NAME              CDATA #REQUIRED
  INITVAL           CDATA #REQUIRED
  CONSTRAINT  CDATA #REQUIRED
  DERIVATION        (true|false)  "false"
  CLASSSCOPE        (true|false)  "false">
```

표 <4-1> 클래스 DTD의 일부

5. 결론 및 향후 연구

본 연구에서는 XML을 이용하여 Java 원시 프로그램을 대상으로 분산된 환경에서 효과적인 정보 제공과 정보 공유를 목적으로 DTD를 제안하였다. Java 원시 프로그램의 구성모듈은 클래스를 중심으로 한다. 하지만, 제안된 DTD는 아직 많은 부분이 부족하고, 실제적이고 실질적인 이용을 위해서는 많은 연구가 필요하다.

향후 연구방향은 Java 원시 프로그램에 대한 DTD를 지속적으로 보완 발전시키는 것이 급선무이다. 또한 이와 병행하여 Java DTD를 파싱하고, 에러 유무를 판단하기 위한 파서에 대한 연구가 필요하다. 그리고 다양한 스타일에 대한 추가 연구가 필요하다.

[참고문헌]

[1] Roser S. P., "SOFTWARE ENGINEERING - A Practitioner's Approach 4/E", McGrow-Hill, 1998.

[2] J. Suzuki, Y. Yamamoto, "Managing the Software Design Document with XML", ACM SIGDOC, 1998.

[3] Bray. T, et.al., "Extensible Markup Language 1.0", W3C.

[4] 이원석, "XML Introduction", "http://dblab.comeng.chungnam.ac.kr/~dolphin/xml".

[5] Mark Johnson, "XML for the absolute beginner", JavaWorld, April 1999.

[6] Richard Light, "Presenting XML", Sams.net, 1997.

[7] Adler. S, et.al., "A Proposal for XSL", W3C, 1998.

[8] Rational Software, et.al., "UML Proposal Summary", OMG document number: ad/97/08/02.

[9] Ion P, et.al., "Mathematical Markup Language", W3C.

[10] CheckFree Corp., "Open Financial Exchange Specification 1.0.2", Open Financial Exchange.

[11] Lisa FRIENDLY, "The Design of Distributed Hyperlinked Programming Documentation", Sun Microsystems Inc., 1996

[12] 장근실, 유철중, 장옥배. "주석 패턴을 이용한 Java 프로그램의 문서화 기법", 한국정보처리학회 S/W 품질관리 심포지움 논문집, 1998.