2001년 한국정보처리학회 춘계 학술발표논문집 제8권 제1호

JML 문서를 이용한 자바 정보 추출기에 대한 연구

장근실*
*광양대학 인터넷정보통신과
e-mail:jgs@kwangyang.ac.kr

A Study on Java Information Extractor using JML Document

Geun-Sil Jang*

요약

XML을 중심으로 많은 컴퓨팅 분야에서 다양한 연구가 이루어지고 있는데, 이는 기존의 웹 정보 표현 언어인 HTML이 갖는 부족한 부분을 해결할 수 있는 XML의 특징 때문이다. JML은 Java Markup Language의 약어로서 Java로 작성된 원시코드의 정보를 다양한 목적으로 이용하는데 적합하도록 작성된 XML의 응용으로 클래스 계층구조나 클래스 관계성 및 메소드 등에 관련된 다양한 정보를 효과적으로 표현할 수 있는 DTD를 포함한다. 본 연구의 목적은 역공학 측면에서의 JML의 응용으로, JML문서에 포함된 정보로부터 Java 응용 프로그램의 스켈레턴 코드를 생성하는데 있다. 본 연구의 의미는 기존에 수행된 Java 응용 프로그램의 정보를 추출하여 JML문서를 생성해 주는 도구와 접목시킴으로써 순방향과 역방향 측면에서 모두 접근가능한 도구를 제공하는데 있다.

1. 서론

XML(eXtensible Markup Language)이 발표된 이후 많은 컴퓨팅 분야에서 XML을 이용한 방법론이 적용되고 있다. 웹을 기반으로 하는 멀티미디어 스트리밍 데이터의 표현이나 전자상거래 또는 이기종간의 정보 공유가 필요할 때 메타언어로서 XML을 이용한다[1,2]. 이처럼 다양한 연구가 이루어지는 이유는 기존의 웹 정보 표현 언어인 HTML(Hyper Text Markup Language)이 갖는 부족한 부분을 해결할 수 있는 XML의 장점 때문이다.

JML(Java Markup Language)은 XML 애플리케이션의 일종으로 Java로 작성된 원시 프로그램으로부터 소프트웨어 공학적으로 필요한 다양한 정보를 표현하고 공유할 수 있고[3], 이기종간의 시스템 개발이나 원시 코드의 이해에 필요한 정보를 보다 자세하게 얻을 수 있다. 본 연구는 역공학 측면에서의 JML의 응용방법으로, JML문서에 포함된 정보로부터 적합한 Java 원시 프로그램의 스켈레턴 코드(skeleton code)를 생성하는데 필요한 알고리즘을 개발하는데 있으며, 여기에서는 JML 문서로부터 스켈레턴 코드에 필요한 정보를 추출하는 알고리즘에 대해서 기술한다. 본 연구의 의미는 기존에 수행된 Java 응용 프로그램의 정보를 추출하여 JML문서를 생성해 주는 도구와 접목시킴으로써 순공학(Forward Engineering)과 역공학(Reverse Engineering) 측면에서 모두 접근가능한 도구를 제공하는데 있다.

본 연구의 구성은 다음과 같다. 먼저 2장 관련연구에서는 XML 문서를 프로세싱하는데 일반적으로 이용되는 DOM과 SAX에 대해서 살펴본다. 3장에서는 본 연구에서 이용하는 JML의 특징과 문서의 구조를 기술하는 DTD(Document Type Definition)에 대해서 살펴보고, 4장에서는 JML 문서로부터 정보를 추출하는 정보추출기에 대해서 살펴본다. 마지막으로 5장에서 향후 연구 및 결론에 대해서 언급한다.

2. 관련연구

XML 문서를 처리할 때는 DOM이나 SAX와 같은 API(Application Programming Interface)를 이용한

다[4,5,6,7,8,9]. 이들 API는 C++나 Java와 같은 특정 언어나 환경에서 이용할 수 있도록 패키지화 되어 제공되며, Tcl/Tk나 Python과 같은 스크립트 언어에서도 각각의 환경에 커스터마이즈된 모듈(module)이나 익스텐션(extention)이 존재한다. 표 1은 DOM과 SAX를 비교한 것이다.

2.1 DOM(Document Object Model)

DOM은 1998년 W3C에서 지정한 트리 구조 형태의 API로 전체 문서를 다루는 응용 프로그램의 개발에 유용하며, XML 문서의 구조를 트리형태로 생성한다. 현재 DOM의 버전은 레벨 2이고, 현재 레벨 3에 대한 연구가 진행중이다[1,10]. 다음은 DOM의 사용에 적합한 부분이다.

- XML 문서를 구조적으로 변경
- 메모리에 있는 문서를 다른 응용 프로그램과 공유할 때

2.2 SAX(Simple API for XML)

SAX는 XML-Dev 메일링 리스트 회원들에 의해 제안되고 계속 발전되고 있으며 현재는 버전 2.0까지 발표된 상태이다. DOM과 달리 문서의 데이터 구조를 생성하지 못하며, 엘리먼트의 시작이나 엘리먼트의 끝과 같은 이벤트를 생성한다[1,4]. 응용 프로그램은 이런 이벤트를 가로채고 적절하게 반응하도록 할 수 있다. 데이터 구조를 생성하지 않기 때문에 SAX가 유용한 부분은 다음과 같다.

- 대량의 XML 문서
- 주변의 문서구조가 적절하기 않은 엘리먼트를 처리할 때

비교항목	DOM	SAX
트리구조	생성함	-
Event-Driven	-	지원함
문서의 양	소량에 적합	대량에 적합
속도	느림	빠름

표 1. DOM과 SAX의 비교[1]

3. JML : Java Markup Language

JML은 Java로 작성된 원시코드의 정보를 다양한 목적으로 이용하는데 적합하도록 작성된 XML 애플리케이션으로 클래스 계층구조나 클래스 관계성 및 메소드 등에 관련된 다양한 정보를 효과적으로 표현할 수 있다. 이런 정보들은 소프트웨어 공학 측면의 많은 부분에서 중요한 역할을 한다[11,12].

JML에 포함된 DTD는 [3]에서 제안된 Java 언어용 DTD를 근거로 한다. Java DTD는 크게 3개의 모듈로 구성된다. 첫 번째 모듈은 클래스 DTD이고, 두 번째 모듈은 메소드 DTD이고, 세 번째 모듈은 데이터멤버 DTD이다. 또한 XML의 특성상 루트노드의 역할이 필요한데, 이 부분은 프로젝트에 대한 개괄적인 정보와 "import" 문장을 처리하는 태그(tag)들로 구성된다. 다음의 [그림1]은 JML DTD 파일을 XML Spy에서 읽어들인 결과를 보여준다.

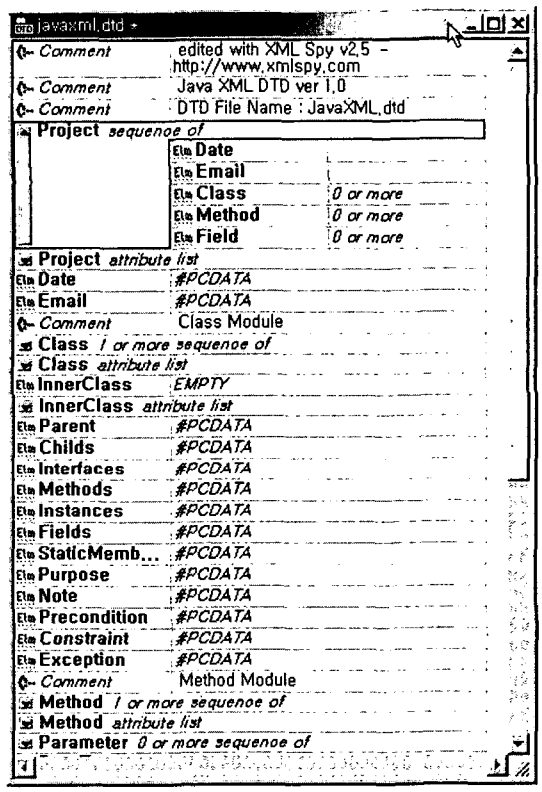

그림 1 JML DTD의 내용

4. 정보추출기(Information Extractor)

JML 문서는 JML 도구를 이용하는 방법과 DTD에 적합하도록 수작업을 통해서 작성할 수 있다. 먼저 JML 도구를 이용하는 경우는 만들고자 하는 문서에 해당되는 Java 원시 코드가 존재할 경우에 유용한 방법이고, 수작업을 통한 방법은 개발자가 DTD에 적합한 내용을 메모장이나 일반 텍스트 편집기를 이용하여 직접 기술해야 한다. 두 번째 방법은 사용자 환경에서 이용가능한 Java 원시 코드가 없을 경우나 클래스 및 메소드 등을 포함하는 스켈레톤 코드를 개발하는 경우에 유용하다. 이런 방법

으로 생성된 스켈레턴 코드는 사용자의 부가적인 정보 제공(예, 순환구문, 분기구문, 수식 등)에 의해 완전한 원시 코드로 변환이 가능하다.

JML 문서에 포함되는 내용들은 원시 코드로부터 얻은 정보들로 구성되므로 완전한 Java 원시 코드를 얻을 수는 없으며, 상속구조나 관계성을 표현할 수 있는 클래스, 메소드 및 해당 데이터 변수를 포함할 수 있다. 'import' 문장이나 'interface' 문장 역시 그 범위에 포함된다. 다음의 표 2는 JML DTD의 클래스 모듈로부터 얻을 수 있는 Java 원시 코드의 정보 구조를 보여준다. 복수개의 클래스로 구성된 문서의 경우는 개수만큼의 클래스 배열이 생성된다. 추출된 정보들 중에서 클래스의 목적을 기술하는 'Purpose' 항목, 기타 정보를 기술하는 'Note' 항목, 클래스의 전제조건을 기술하는 'Precondition' 항목 및 클래스의 예외조건을 기술하는 'Exception' 항목은 생성된 클래스 스켈레턴 코드 부분에 주석으로 나타나게 된다.

배열 이름	CL+클래스명		
배열요소 이름	해당 데이터	Type	구분자
Name	클래스명 타입 액세스모드	리스트형	::
Parent	슈퍼클래스명	문자형	
Child	부모클래스명	문자형	
Interface	인터페이스명	문자형	
InnerClass	인너클래스명	문자형	
Methods	메소드들	리스트형	::
Fields	필드변수들	리스트형	::
StaticMembers	정적변수들	리스트형	::
Purpose	목적	문자형	
Note	노트	문자형	
Precondition	전제조건	문자형	
Exception	예외처리	문자형	

표 2. 클래스 모듈의 자료구조

위에서 언급한 방법으로 작성된 JML 문서는 XML의 특성을 그대로 상속받기 때문에 각 정보의 의미를 나타내는 태그를 파싱함으로써 정보의 이름과 정보의 내용을 얻을 수 있다. 동일한 정보를 표현하는 HMTL 문서의 경우는 태그를 이용한 정보의 추출 및 구분이 불가능하므로[1,4,10] 태그들 사이의 내용을 다시 분석하고 파싱해야만 한다. 이와 같은 방법은 더 복잡하고, 어렵다.

하지만, 본 연구에서는 범용적인 XML 문서를 대상으로 하는 것이 아니므로, 2장에서 살펴본 DOM이나 SAX와 같은 파서를 이용하지 않고, JML 문서만을 처리하는 전용 파서(Special Parser)를 구현할 것이다. 전용 파서를 구현하는 이유는 추출되는 정보로부터 Java 원시 코드 생성에 이용되는 다양한 정보들을 보다 효과적으로 데이터베이스화하고, 문서 생성시간을 감소시키기 위해서이다.

JML 문서의 내용은 [그림 2]에서 보이는 것처럼 트리 형식으로 표현되기 때문에 'Project'라는 루트 노드부터 'Name' 노드나 'Type' 노드와 같은 종단 노드에 이르기까지 prefix notation을 통해 추출된 노드를 정보 배열에 저장하면 하나의 완전한 클래스 정보를 얻을 수 있다. 표 3은 전용 파서에 대한 개괄적인 알고리즘을 보여준다.

```
파일 open
// JML 파일의 확장자는 'xml'이다.
While (!EOF)
   현재 라인 read
   리스트 생성
   if (리스트 요소의 수 == 1)
      // 자식 노드가 있는 태그
      태그 추출
      // 추출되는 태그가 표현가능한 정보의 형식임.
      라인++
      continue
   else
      // 에트리뷰트가 있거나 종료태그가 존재할 때
      if (리스트 마지막 요소에서 "</태그명>" 매칭)
         // 종료태그가 존재할 때.
         정보추출
         // 리스트 두 번째 요소부터
         // 마지막 요소 -1번째까지가 정보가 됨.
         라인++
         continue
      else
         // 에트리뷰트가 존재할 때
         while (리스트 두 번째 요소부터 마지막 요소까지)
            각 요소를 '='로 구분
            좌측은 태그
            우측은 정보
         end of while
         continue
      end of if
   else
      // 종료 태그임
      자식노드를 갖는 태그의 정보추출
      continue
   end of if
end of while
```

표 3 개괄적인 파서 알고리즘

5. 결론

본 논문에서는 이전 연구에서 수행된 연구결과인 Java 원시 프로그램을 위한 XML 응용의 일종인 JML을 이용하여 문서화된 내용으로부터 정보를 추

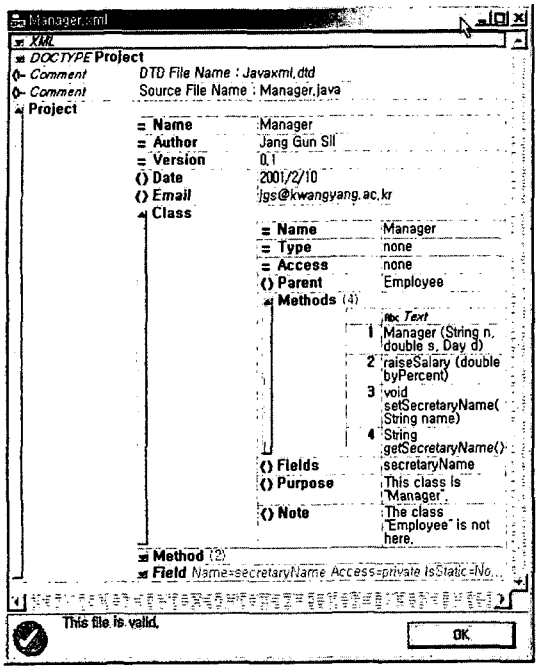

그림 2. JML문서의 구조

출하기 위한 정보추출기(파서)에 대해서 알아보았고, XML 문서를 프로세싱하는데 이용되는 파서(DOM과 SAX)에 대해서 살펴보았다. JML 문서를 생성하는 방법에는 이전 연구의 수행결과인 JML 문서 생성기를 이용하는 방법과 사용자가 일반 편집기나 전용 편집기를 이용하여 직접 기술하는 방법이 있으며, 본 연구는 JML 문서만을 가지고 있을 때나 컴퓨팅 환경에서 쉽게 접할 수 있는 비쥬얼 도구가 사용자의 환경에 없을 경우에 적합하다.

앞으로는 정보추출기로부터 생성된 정보를 이용하여 해당 문서에 적합한 Java 원시 프로그램에 대한 스켈레턴 코드를 생성하는 문서생성기에 대한 연구가 필요하다. 이런 과정을 통하여 JML 문서를 생성하는 것만으로 응용프로그램의 구조를 충분히 구축할 수 있는 스켈레턴 코드를 생성할 수 있고, Java 원시코드에 대해서 순공학적인 측면과 역공학적인 측면에서의 접근방법을 제공할 수 있다.

또한 사용자들이 쉽게 접근할 수 있도록 GUI (Graphic User Interface)를 지원하는 부분과 운영체제나 하드웨어에 독립적으로 운용될 수 있도록 프로젝트를 개발할 예정이다. 또한 기존의 연구결과와의 통합과 DTD의 엘리먼트(Element)와 에트리뷰트(Attribute)를 시각화하여 대부분의 개발도구들이 지원하는 드레그엔드롭(drag and drop) 방식을 제공하여 보다 편리한 문서개발환경으로 확대할 것이다.

[참고문헌]

[1] 이강찬, "지능형 인터넷 세상을 여는 XML", 마이크로 소프트웨어, 2001-3, 196~209.

[2] Junichi Suzuki 외 1인, *Managing the Software Design Documentation with XML,* ACM SIGDOC, 1998.

[3] Jang Geunsil 외 2인, *Information Sharing of Java Program Using XML,* In Proceedings of the ACIS 1st International Conference on SNPD '00, Reims in France, May, 2000, 384~391.

[4] 허준회, "아스키 문서를 대체할 메타 데이터 포맷 XML", 마이크로 소프트웨어, 2000-3, 280~293.

[5] Eddy Schnyder, *"Teach Yourself XML",* IDG Books.

[6] Hiroshi Maruyama 외 2인, *"XML and Java™ Developing Web Applications",* Addison Wesley.

[7] Simon St. Laurent 외 1인, *"Building XML Applications",* McGraw-Hill.

[8] Mark Wilson 외 1인, *"XML Programming with VB and ASP",* Manning Publications.

[9] Ceponcus 외 1인, *"Applied XML",* WILEY.

[10] Keten C. Patel, Storing and Retrieving XML Content, XML Journal, Vol 2. Issue 1, 48~51.

[11] Roger S. Pressman, *"Software Engineering A Practltiners' Approach"* 3rd Ed, McGraw Hill.

[12] 장옥배 외 5인, *"소프트웨어공학-이론과 실제"*, 한산출판사