XML 구문 지향 편집기의 자동 생성

박호병°,조용윤,신경희,김영철,유재우
숭실대학교 컴퓨터학과
{r5me°, pycho, khshin, yckim, cwyoo}@ss.ssu.ac.kr

An Automatic Generation of XML Syntax Directed Editor

Ho-Byung Park° , Yong-Yoon Cho, Kyoung-Hee Shin, Young-Chul Kim, Chae-Woo Yoo
Soongsil University, School of Computing

요 약

XML문서를 작성하는데 있어서 그 규칙이나 DTD에 익숙하지 않은 개발자에게 구문 지향 편집기는 효율적인 환경을 제공해 준다. 이러한 구문 지향 편집기를 생성하는 도구로서 Synthesizer Generator등이 잘 알려져 있는데, 사용자는 Synthesizer Generator를 위해 구문 지향 편집기 생성 정보 표현 언어인 SSL(Syntheizer Specification Language)을 직접 작성해야 한다. 본 연구는 웹 문서 표준인 XML 구문 지향 편집기를 자동 생성하기 위한 방법을 제안한다. 제안된 방법은 입력된 XML DTD를 AST 형태로 변경하여 DAG(Directed Acyclic Graph)를 추출하는 DAG 변환기, 생성된 DAG를 SSL로 변환하기 위한 DAG 핸들러와 SSL 변환기 모듈 그리고 변환된 SSL을 이용해 XML 구문 지향 편집기를 자동 생성하기 위한 Synthesizer Generator 사용을 포함한다. SSL 변환기는 SSL문서를 자동 생성하기 위한 모듈로서 추상 구문변환 모듈, 역 파싱(Unparsing scheme)모듈, 변형 규칙(Transformation rule) 표현 모듈로 구성된다. 사용자는 SSL변환기와 Synthesizer Generator의 사용을 통해 SSL을 직접 코딩해야 하는 노력과 불필요한 학습 시간을 줄이고 빠르고 정확한 XML 구문 지향 편집기를 생성하므로 효율적인 XML 문서 작성할 수 있다.

1. 서 론

구문 지향 편집기(Syntax-Directed Editor)는 사용하는 문법에 따라 구조적 문서 작성을 유도하는 효율적인 편집 환경이다. 그러나 구문 지향 편집기는 특정 문법 구문에 맞게 편집 환경을 제공하므로 사용자는 원하는 문법에 따라 각기 다른 구문 지향 편집기를 작성해야 할 필요가 있다. 따라서 사용자는 필요에 따라 특정 문법에 맞는 구문 지향 편집기를 생성해 주는 자동화 도구를 이용할 수 있다. 이러한 구문 지향 편집기의 대표적인 생성 도구로서 Synthesizer Generator가 잘 알려져 있다. 하지만 Synthesizer Generator는 구문 지향 편집기를 생성하는 과정 중에 필요한 문법 정보를 위해 SSL을 이용한다. 따라서 사용자는 Synthesizer Generator를 이용해 XML 구문 지향 편집기를 생성하기 위해 SSL 표현 방법을 학습하고 XML DTD 문법을 SSL로 직접 작성해야 한다. 이러한 사실은 사용자에게 또 다른 학습 시간과 작업 노력 부담이 된다.

본 연구는 웹 문서 표준인 XML 구문 지향 편집기의 자동 생성을 위한 방법을 제안한다. 현재 인터넷을 통한 문서 표준으로 XML이 주목을 받고 있다. XML 표준을 따르는 마크업(Mark-up) 언어는 다양하며 이러한 특정 마크업 언어에 대한 구문 지향 편집기의 생성은 편집기 자동화 도구의 사용을 필요로 한다.

제안하는 방법은 XML DTD를 입력받아 AST를 생성하고 DAG 구조로 변경하기 위한 DAG 변환기, 구문 지향 편집기 자동 생성 도구인 Synthesizer Generator가 사용하는 SSL(Synthesizer Specification Language)을 출력하기 위한 DAG 핸들러와 SSL 변환기를 포함한다. SSL은 3개의 주요 구성 요소인 추상 구문, 출력 형식, 변형 규칙으로 구성된다. 따라서 SSL 변환기는 3개의 주요 변환 모듈인 추상 구문(Abstract Syntax) 변환 모듈, 출력 형식(Unparsing scheme) 지정 모듈, 변형 규칙 (Transformation rule) 표현 모듈로 구성된다. SSL 변환기에 의해 생성된 XML DTD에 대한 SSL 정보는 Synthesizer Generator를 통해 XML 구문 지향 편집기의 자동 생성에 사용된다. 따라서 사용자는 SSL자동 변환기를 통해 입력되는 XML DTD에 대해 SSL을 직접 작성하는 노력 없이 Synthesizer Generator를 이용할 수 있으며 원하는 구문 지향 편집기를 쉽게 얻을 수 있다.

2. 관련연구

2.1 구문 지향 편집기 (Syntax-Directed Editor)

문법에 익숙하지 않은 초보자가 문법에 맞도록 문서를 작성하는 것은 쉬운 일이 아니다. 구문 지향 편집기는 문서의 작성자가 어려운 문법 규칙을 잘 모르더라도 문서를 작성할 수 있도록 일반적인 편집기에 컴파일러의 문법 체크 기능을 갖고 있는 대화식 편집기이다.

구문 지향 편집기는 문서를 편집하면서 대화적이고 직접적으로 문법적 오류를 검증하기 때문에, 문서작성자는 올바른 구조적 문서작성이 가능하며, 문서작성이 용이하여 효율성이 증대된다.

2.2 추상 구문 트리 (Abstract Syntax Tree)

문서의 구문을 검사할 때 자주 사용하는 파스 트리(Parse Tree)와는 달리 문서의 의미를 파악하기 쉽게 구성한 트리 구조로서 번역(translation)에 관계되는 것만으로 문서를 함축하여 표현한 문법 트리이다.

2.3 Synthesizer Generator

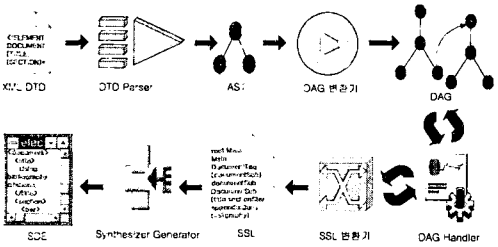
Synthesizer Generator는 언어기반의 구문 지향 편집기를 생성하는 시스템이다. 이는 SSL(Synthesizer Specification Language)을 입력으로 하여 구문 지향 편

집기를 생성하며, 생성된 편집기는 각 각의 문서를 내부적으로 트리 구조로 표현하여 문서를 편집한다.

SSL은 각 언어의 문법규칙을 추상 구문(Abstract Syntax)과, 속성 규칙, 출력형식 지정, 구문 파싱(Parsing Syntax), 변형규칙(Transformation rule)등으로 구성된다. 따라서, Synthesizer Generator를 이용하여 원하는 구문 지향 편집기를 얻기 위해서는 사용자가 직접 이러한 SSL을 작성해야 한다.

3. 시스템 설계

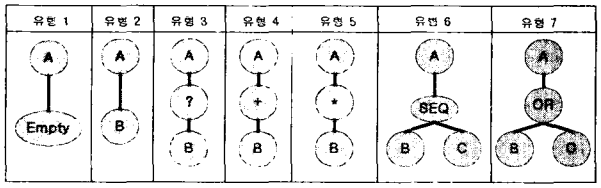
본 연구에서 제시하는 XML 구문 지향 편집기 자동 생성을 위한 전체적인 시스템 구성은 그림[1]과 같다.



[그림 1] XML 구문 지향 편집기 자동 생성을 위한 전체적인 시스템 구성도

위 그림에서 DTD 파서는 XML DTD를 파싱하여 AST자료구조로 변환하고, AST를 DAG 변환기를 통하여 DAG 자료구조로 변환한다. SSL 변환기는 DAG 핸들러를 통하여 DAG를 해석하여 SSL형태로 변환한다. SSL은 Synthesizer Generator를 이용하여 XML DTD에 맞는 XML 구문지향 편집기를 생성한다.

SSL 생성을 위한 DAG는 다음과 같은 유형으로 분류된다.



[그림 2] DTD에서 생성된 DAG의 유형

SSL 변환기는 위의 유형을 추상 구문 변환 모듈과 출력 형식 지정 모듈, 변형 규칙 표현 모듈의 3개 모듈을 통하여 SSL로 변환한다.

3.1 추상 구문 변환 모듈

SSL 변환기에서 가장 핵심적인 부분이며 이를 통해 생성된 추상 구문의 형태가 나머지 모듈과 일관적인 형태를 유지하여 최종 SSL이 생성된다. 추상 구문 변환 모듈은 DAG를 추상 구문으로 변환하기 위해 다음과 같은 규칙을 적용한다.

■ 유형 1의 경우

```
A : A_ELEMEMT();
```

■ 유형 2의 경우

```
A : A_ELEMENT(A_LIST) ;
A_LIST : A_ELEMENT_B(B);
```

■ 유형 3의 경우

```
A : A_ELEMENT(A_LIST);
OPTIONAL A_LIST;
A_LIST : A_NIL() | A_ELEMENT_B(B);
```

■ 유형 4의 경우

```
A : A_ELEMENT(A_LIST);
OPTIONAL LIST A_LIST;
A_LIST : A_NIL() | A_ELEMENT_B(B A_LIST);
```

■ 유형 5의 경우

```
A : A_ELEMENT(A_LIST);
OPTIONAL A_LIST;
A_LIST : A_NIL() | A_ELEMENT_B(B A_LIST);
```

■ 유형 6의 경우

```
A : A_ELEMENT(A_LIST);
A_LIST : A_LIST_ELEMENT(B C);
```

■ 유형 7의 경우

```
A : A_ELEMENT(A_LIST);
A_LIST : A_NIL() | A_ELEMENT_B(B) | A_ELEMENT_C(C);
```

3.2 출력 형식 지정 모듈

출력 형식 변환모듈은 추상 구문의 자료구조를 화면에 표시하기 위해 텍스트 형태로 역 파싱하는 모듈이며, 추상 구문 변환모듈을 통해서 생성된 추상 구문을 이용한다. 출력 형식 변환모듈은 DAG를 출력 형식지정 구문으로 변환하기 위해 다음과 같은 규칙을 적용한다.

■ 유형 1의 경우

```
A : A_ELEMENT [@ : "<A/>%n"];
```

■ 유형 2의 경우

```
A : A_ELEMENT [@ : "<A>%t%n" @ "%b%n</A>"];
A_LIST : A_ELEMENT_B [@ : @];
```

■ 유형 3의 경우

```
A : A_ELEMENT [@ : "<A>%t%n" @ "%b%n</A>"];
A_LIST : A_NIL[@ : "<!-- TRANSFORM IT-->%n"]
       | A_ELEMENT_B [@ : @];
```

■ 유형 4, 5의 경우

```
A : A_ELEMENT[@ : "<A>%t%n" @ "%b%n</A>"];
A_LIST : A_NIL[@ : "<!-- TRANSFORM IT -->%n"]
       | A_ELEMENT_B[@ : @ @];
```

■ 유형 6의 경우

```
A : A_ELEMENT[@ : "<A>%t%n" @ "%b%n</A>"];
A_LIST : A_LIST_ELEMENT[@ : @ @];
```

■ 유형 7의 경우

```
A : A_ELEMENT[@ : "<A>%t%n" @ "%b%n</A>"];
A_LIST : A_NIL[@ : "<!-- TRANSFORM IT-->%n"]
       | A_ELEMENT_B[@ : @]
       | A_ELEMENT_C[@ : @];
```

3.3 변형 규칙 표현 모듈

변형 규칙 표현은 선택된 추상 구문을 변형 규칙 표현 구

문에서 기술한 유형으로 재 구성하는 일종의 템플릿 역할을 하는 부분이다. 이 모듈은 DAG의 모든 유형에 대하여 적용하는 것이 아니라 하위 엘리먼트가 고정되지 않은 선택의 상황에 적용한다.

■ 유형 3, 4, 5의 경우

```
transform A_LIST on
        "B" <A_LIST> : A_ELEMENT_B(<B>);
(유형 4 "B+", 유형 5 "B*")
```

■ 유형 7의 경우

```
tranform A_LIST on
        "B" <A_LIST> : A_ELEMENT_B(<B>) ,
        "C" <A_LIST> : A_ELEMENT_C(<C>) ;
```

4. 시스템 구현 및 실험

4.1 구현 및 실험 환경

DTD 파서는 wutka사의 자바 DTD 파서를 이용하였으며 DAG 변환기와 DAG 핸들러, SSL 변환기는 자바 기반으로 개발되었다. 구문 지향 편집기 자동 생성 도구로서 GrammaTech 사에서 개발한 Synthesizer Generator 5.0을 이용하였고, 컴퓨터 시스템은 Solaris 2.5 운영체제의 Sun ultra enterprise 150에서 개발 실험하였다.

4.2 실험 결과

다음은 XML 구문 지향 편집기 생성을 위한 입력으로 실험에서 사용한 XML DTD 문서이다.

```
<!ELEMENT DOCUMENT
 (TITLE,(SECTION)*,(APPENDIX)?,BIBLIOGRAPHY)>
<!ELEMENT TITLE (#PCDATA)>
<!ELEMENT APPENDIX (SECTION)+>
<!ELEMENT SECTION (PAR)*>
<!ELEMENT PAR (#PCDATA | CITE)*>
<!ELEMENT CITE EMPTY>
<!ELEMENT BIBLIOGRAPHY (BIBITEM)+>
<!ELEMENT BIBITEM (#PCDATA)>
```

다음은 SSL 변환기가 추상 구문 변환 모듈을 통해 XML DTD를 변환해 생성한 AST 정보의 일부분이다.

```
root Main;
Main : Document_ELEMENT (document_LIST);
document_LIST : Document_LIST_ELEMENT (title sectionStar
        appendixQues bibliography);
sectionStar : SectionStar_Nil ()
        | SectionStarList (section sectionStar);
```

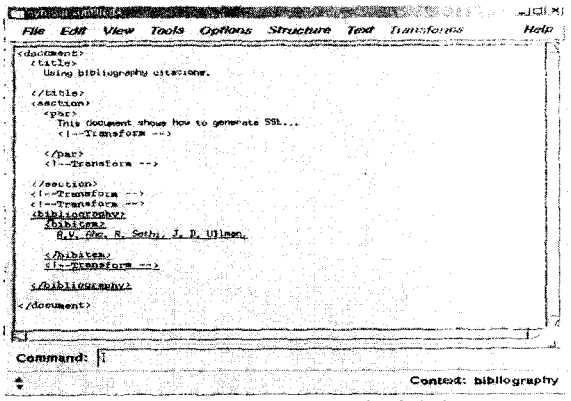
다음은 추상 구문 변환 모듈로부터 얻어진 AST 정보를 이용해 출력 형식 지정 모듈이 생성한 정보이다.

```
Main : Document_ELEMENT [@ : "<document>%t%n" @
        "%b%n</document>"];
document_LIST : Document_LIST_ELEMENT [@ : @ @ @ @];
sectionStar : SectionStar_Nil [@ : "<!--Transform -->%n"]
        | SectionStarList [@ : @ @];
```

다음은 SSL 변환기의 변형 규칙 표현 모듈이 AST 변환 모듈로부터 얻어진 AST 정보를 이용해 생성한 정보이다.

```
transform sectionStar on
        "section section*" <sectionStar> :
        SectionStarList (<section>,<sectionStar>);
transform appendixQues on
        "appendix" <appendixQues> :
        AppendixQuesList(<appendix>);
```

다음은 SSL 변환기로부터 얻어진 SSL을 이용해 Synthesizer Generator가 생성한 XML 구문 지향 편집기이다. 이것은 XML DTD의 문법의 정의에 따라 편집이 가능하다.



[그림 3] 자동 생성된 XML 구문 지향 편집기

5. 결론

XML 구문 지향 편집기는 사용자에게 주어진 DTD 문법에 따라 구조적 문서 작성을 유도하는 효율적인 환경을 제공한다. 본 연구는 XML 구문 지향 편집기를 자동 생성하기 위한 방법을 제안하였다. 제안된 방법은 구문 지향 편집기를 자동 생성하기 위해 자동생성 도구인 Synthesizer Generator를 이용하였으며, 이 과정에서 XML DTD를 SSL 로 변환하기 위한 모듈들을 구현했다. 이러한 모듈들은 XML DTD 파서를 통해 생성된 AST를 DAG로 변경하기 위한 DAG 변환기, DAG로부터 SSL로 변환을 위한 DAG 핸들러와 SSL 변환기 모듈을 포함한다. 또한, 구현된 SSL 변환기는 크게 3개의 주요 변환 모듈인 추상 구문(Abstract Syntax)변환 모듈, 출력 형식(Unparsing scheme)지정 모듈, 변형 규칙(Transformation rule) 표현 모듈로 구성되었다. 제안된 방법을 통해 사용자는 SSL을 직접 작성하지 않고 XML DTD만을 입력하여 추가적인 노력 없이 Synthesizer Generator를 통해 원하는 구문 지향 편집기를 얻을 수 있다.

6. 참고문헌

[1] A.V.Aho, R.Sethi, J.D.Ulman, *Compilers: Principles, Techniques, and Tools*, Addison-Wesley, 1985.

[2] David A Watt & Deryck F Brown, *Programming Language Processros In JAVA*, Prentice Hall, 2000.

[3] Extensible Markup Language (XML), http://www.w3.org/XML/

[4] "The Synthesizer Generator: Specifying an Editing Environment", http://www.grammatech.com/research/SGspecifying/

[5] Thomas W.Reps & Tim Teitelbaum, *The Synthesizer Generator: A System for Constructing Language- Based Editors*. Springer-Verlag, 1989.

[6] 안보희, "SGML문서 저작 도구의 설계와 구현", 숭실대학교 박사학위논문, 1998.

[7] JAVA DTD Parser http://www.wutka.com/dtdparser.html