멀티미디어 정보의 효율적인 교환을 위한 파생 DTD 관리 시스템의 개발

서 승 현[U] 김 유 성
인하대학교 전자계산공학과
g1991272@inhavision.inha.ac.kr yskim@inha.ac.kr

A Development of Driven DTD Management System for Efficient Exchange of Multimedia Information

Seung-Hyun Suh[U] Yoo-Sung Kim
Dept. of Computer Science & Engineering, Inha University

요 약

본 논문은 원본 DTD에서 사용자 요구에 따르는 파생 DTD를 생성, 관리하는 시스템을 설계함으로써 사용자 요구에 적합한 유효한 XML 문서의 생성 및 검증을 효율적으로 처리하는 시스템을 개발하고자 한다. 본 시스템은 원본 DTD에서 파생 DTD를 생성하여 클라이언트에서 생성된 XML 문서에 대한 유효성을 검증하고, 사용자 질의의 결과로 생성된 서버측의 XML 문서에 대한 유효성을 효율적으로 검증하는 기능을 지원하고 있다. 여기서 사용되는 파생 DTD는 사용자 요구를 받아서 하나의 원본 DTD에서 파생 DTD 생성 알고리즘에 의해 생성된다. 이렇게 생성된 파생 DTD로 XML 문서를 DOM API의 에러 체크 메소드를 통해 XML 문서가 유효하고 정확한지 검증한다. 이러한 절차를 통해서 유효성을 검증받은 XML 문서를 유효한 데이터로서 관리한다. 이러한 절차를 통해 클라이언트와 서버 사이에서 전송되는 XML 문서의 유효성 및 정확성을 보장할 수 있을 뿐만 아니라, 클라이언트에서 만든 XML문서를 파생 DTD를 이용해 유효성을 보장하여 다른 클라이언트에 보낼 수 있다.

1. 서론

XML(eXtensible Markup Language)은 현재 EDI(Electronic Data Interchange) 및 무선 인터넷에서 각광을 받고 있고 앞으로 더욱 잠재력을 가진 마크업 언어이다. XML은 기존의 마크업 언어와는 차별화된 특징을 가진다. 그 중 가장 큰 차이점은 태그내의 컨텐츠를 데이터로 사용할 수 있다는 점과 사용자 정의 태그를 사용한다는 것이다. 사용자 정의 태그를 사용하기 위해서는 태그를 정의하는 부분인 DTD(Document Type Definition)가 있어야 한다. 이 DTD에서 전체적인 XML 문서에 필요한 엘리먼트를 정의하게 된다. 그러므로, DTD는 XML 문서에 대한 스키마라고 부르기도 한다. 여기서 스키마란 하나의 XML 문서 안에 어떤 엘리먼트들을 담을 수 있는지에 대한 규칙들의 집합이다. 이런 스키마가 있기 때문에 프로그래머들은 어떠한 애매함이나 혼란 없이도 문서 안에 특정한 방식으로 배치된 정보를 뽑아서 처리하는 소프트웨어를 만들 수 있는 것이다[1]. 또, 문서의 유효성 역시 스키마에 근거해서 판단하게 된다. 스키마에 근거해서 유효한 문서라면 그 스키마에 근거해서 만든 애플리케이션에게도 유효한 것이다[2]. 이렇게 스키마(DTD)에 근거한 문서의 유효성을 검사함으로써 교환하고자 하는 정보가 상대편에게 제대로 이해되고 사용될 것이라는 확신을 더욱 높일 수 있다. 문서를 보내는 쪽은 먼저 문서의 유효성을 검사한 후, 상대방에게 보내고, 받는 쪽은 우선 문서의 유효성을 검사한 후, 문서의 대한 컨텐츠를 처리할 것이다. 그러나, DTD는 확장이 되지 않는 단점이 있기 때문에 유효한 XML 문서(Valid-XML)를 생성할 때 항상 애플리케이션에 맞는 원본 DTD의 규칙에 따르는 문서를 만들어야 한다. 이러한 문제점을 해결하고자 본 논문에서는 원본 DTD에서 사용자의 요구에 맞는 파생 DTD를 정의하여 생성, 관리하는 시스템을 개발한다. 이러한 파생 DTD를 생성함으로써 사용자가 원하는 요소를 가지는 XML 문서에 대한 유효성 및 정확성을 보장할 수 있고 다양한 경우의 유효한 XML 문서를 생성할 수 있다. 즉, 파생 DTD를 사용함으로써 많은 유효하고 정확한 XML 문서를 데이터로서 저장 관리할 수 있다. 물론, 적격 XML 문서(Well-formed XML Document)로서 데이터를 주고받을 수도 있지만 이러한 적격 XML 문서는 DTD의 결여로 문서에 대한 유효성을 검증 받지 못하기 때문에 적격 XML 문서를 받아서 처리하는 수신자 측에서는 원본 DTD로 XML문서의 유효성을 검증할 수밖에 없다. 그러나, 원본 DTD로 XML문서의 유효성을 검증하는 것은 사용자의 요구사항을 완벽하게 만족하지 못한다. 이러한 문제점을 야기하기 때문에 그러한 문제점을 해결하기 위한 방안으로 파생 DTD로 유효성을 검증하는 방안을 제시한다. 본 논문의 구성은 다음과 같다. 2장은 관련 연구로서 DTD의 필요성 및 DOM의 역할 및 필요성에 대해서 기술하고 3장은 파생 DTD를 이용한 확장 및 참조방안에 대해 살펴본다. 4장에서는 결론 및 향후 연구방향으로 본 논문을 맺는다.

2. 관련연구

2.1 DTD의 필요성

XML의 경우 스키마란 데이터가 마크업 되는 방식을 뜻하는데 현재의 DTD가 일종의 스키마라고 할 수 있다. 그리고 DTD는 XML의 구문구조를 사용하지 않고 EBNF(Extended Backus-Naur Form)라는 문법을 사용한다[2].

XML에서 나타나는 스키마의 종류는 다음과 같이 여러 가지로 명명되고 사용되어지고 있다. XML-Data와 DCD(Document

Content Definition for XML) 그리고 XML-DEV 메일링 리스트를 통해 제안된 SOX(Schema for Object-oriented XML)가 있다. 즉, XML 자체의 구문구조에 기반해서 스키마를 기술할 수 있어야 한다는 제안들이 지지를 얻었다. 이렇게 각각의 스키마들은 시스템의 특성에 따라 변화되어 다른 이름으로 불리고 있지만 큰 범주에서는 같은 것이다.

애플리케이션 설계목표를 만족하기 위한 표준 DTD를 만들 때에는 그 분야에서 널리 인정하는 모든 요소들을 가진 DTD를 설계하여야 한다. 따라서, DTD 설계는 복잡하고, 까다로운 작업이 된다. 그러나, 사용자는 모든 요소를 원하지 않을 수도 있다. 그런 경우에는 사용자가 모든 요소를 가지는 원본 DTD에서 사용자가 원하는 요소를 선택해서 새로운 DTD를 만들어서 사용하는 방법을 생각할 수 있다.

2.2 DOM의 역할 및 필요성

DOM(Document Object Model)은 HTML과 XML 문서를 위한 응용프로그램 인터페이스이며, DOM은 문서를 접근하고 조작하기 위한 방법으로서 문서의 논리적 구조를 정의하고 있다. 그리고, 사용자들은 문서를 생성하고 그 문서의 구조에 따라 탐색하고 엘리먼트와 문서 내용을 추가, 수정, 삭제할 수 있다[3]. DOM은 여러 가지 기능을 가진 메소드들과 속성들을 가지고 있고 이러한 메소드들을 이용하여 XML 문서를 하나의 객체로 보고 작업을 수행한다. 메소드들 중에는 XML 문서의 에러를 체크하기 위한 메소드들이 존재한다. 이러한 DOM의 여러 가지 기능들 중에서 XML 문서가 DTD를 기반으로 해서 제대로 파싱되는지 즉, 생성된 XML 문서가 오류 없이 제대로 작동되는지를 살펴보기 위해 본 논문에서는 DOM을 사용한다[4].

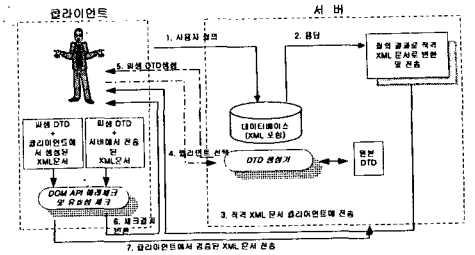
3. 파생 DTD를 이용한 확장 및 참조방안

3.1 애플리케이션 레벨에서 파생 DTD의 필요성

본 논문에서 사용되는 파생 DTD는 XML 관련 애플리케이션을 만들 때 우선적으로 그 시스템에 맞는 DTD를 설계하고 그 DTD에 기반한 애플리케이션을 만드는 것이 일반적이다. 그렇기 때문에 DTD를 얼마나 효율적으로 설계하느냐는 그 애플리케이션의 효율성을 좌우한다고 해도 과언이 아니다. 그래서, DTD는 애플리케이션에서 제공하는 모든 엘리먼트들을 제공할 수 있게 하기 위해서 사용될 수 있는 모든 엘리먼트들을 정의해야 한다. 사용자가 자신이 원하는 엘리먼트의 정보를 요구할 때는 적격 XML문서로 데이터를 보여주는 것이 일반적이다. 적격 XML 문서로 나타내는 것은 DTD를 포함하지 않기 때문에 간편하게 나타낼 수도 있다. 그러나, 그 데이터의 성질에 따라서 적격 XML 문서가 아닌 유효한 XML 문서로도 표현할 수 있어야 한다. 이러한 유효한 XML 문서로 나타내기 위해서는 DTD가 있어야 된다. 그러나, 사용자 요구를 만족시키기 위해서는 사용자 요구에 맞는 DTD가 있어야 한다. 그러므로, 파생 DTD가 필요하게 된다. 이러한 파생 DTD로서 그에 해당하는 XML 문서의 유효성도 평가할 수 있게 되고, 적격 XML 문서뿐만 아니라 유효한 XML 문서를 사용자가 받아 보게 됨으로써 그 문서의 컨텐츠에 대한 유효성 및 정확성을 보장할 수 있다. 그러므로, 보다 나은 애플리케이션을 추구한다면 이러한 파생 DTD를 이용한 XML 문서의 검증이 반드시 필요하고 이렇게 검증된 XML 문서를 데이터로서 저장하고 교환함으로써 데이터로서 더욱 가치 있게 만든다.

3.2 파생 DTD를 이용한 확장 시스템 구조

파생 DTD를 이용한 애플리케이션의 전체적인 구조는 [그림 1]과 같다.

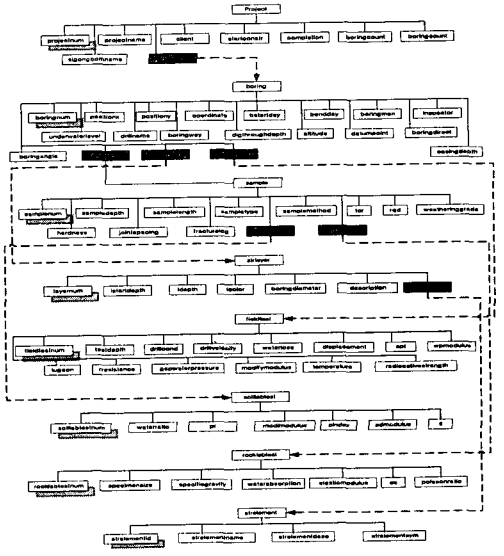


[그림 1] 애플리케이션의 전체적인 구조도

사용자가 서버에 있는 데이터베이스에 질의를 던지면 그 질의의 결과를 XML 문서로 변환하여 클라이언트에 보낸다. 이때 넘겨받는 XML 문서는 적격 XML 문서이다. 왜냐하면 사용자 질의에 따르는 DTD를 생성할 수 없기 때문에 DTD를 가지지 않는 적격 XML 문서를 보낸다. 넘겨받은 적격 XML 문서의 유효성을 판단하기 위해 클라이언트는 서버측의 DTD 생성기에서 클라이언트가 원하는 엘리먼트를 체크하여, 생성된 파생 DTD와 DOM API의 에러체크 메소드로 적격 XML 문서의 유효성 및 정확성을 판단한다[4][5]. 또한, 클라이언트에서도 새로운 XML 문서를 생성하여 이에 맞는 파생 DTD로 유효성을 검증한 XML 문서를 서버측에 전송하여 서버에서 그 문서를 데이터로서 저장 및 관리할 수 있게 한다.

3.3 원본 DTD 결정

본 논문의 실험을 위한 원본 DTD는 시추공 정보를 나타내는 메타데이터를 위한 DTD를 사용한다. 이 DTD는 레벨 5의 깊이를 가지고 있다. 세부적인 구조는 [그림 2]와 같다.



[그림 2] 시추공 DTD의 구체적인 구조

[그림 2]에서 그림자를 가지는 사각형은 주 키(Primary Key)를 나타내고 바탕색이 검정색인 사각형은 하위 엘리먼트를 가지는 엘리먼트를 나타내고 있다. 시추공 DTD를 살펴보면 최상

위 레벨은 레벨 1인 Project 루트 엘리먼트이고 레벨 1은 레벨 2를 하위 엘리먼트로 가지며, 레벨 2는 projectnum, projectname, ..., boring 엘리먼트이며, 레벨 2는 레벨 3을 하위 엘리먼트로 가지고, 레벨 3은 boringnum, positionx, ..., sample, strlayer, fieldtest이며, 레벨 4는 sample, strlayer, fieldtest 엘리먼트의 하위 엘리먼트들이고, 레벨 5는 sample의 하위 엘리먼트인 soillabtest와 rocklabtest의 하위 엘리먼트들과, strlayer의 하위 엘리먼트인 strelement의 하위 엘리먼트들로 구성된 계층적인 구조로 되어 있다. DTD의 계층적인 특징에 따라 자식 엘리먼트와 부모 엘리먼트와의 관계를 표현하는 것은 아주 중요하게 된다. 본 논문에서는 시추공 DTD를 파생 DTD를 생성하기 위한 원본 DTD로 사용한다.

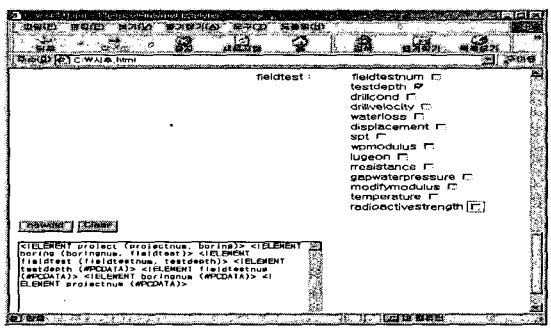
3.4 파생 DTD 생성 알고리즘

[그림 1]의 구조에서 DTD 생성기에 사용되는 파생 DTD를 생성 알고리즘에 대해서 살펴보면 <표 1>과 같다. 우선, 모든 엘리먼트를 체크할 경우, 원본 DTD를 반환하고, 어떤 특정한 레벨의 엘리먼트를 체크할 경우, 그 레벨에 맞는 부모 엘리먼트들과 루트 엘리먼트의 주 키를 가져온다. 이러한 과정을 반복하여 원본 DTD에서 파생 DTD로 생성한다.

<표 1> 파생 DTD 생성 알고리즘

```
Make_newDTD() {
  if 모든 엘리먼트 체크=TRUE then return 원본 DTD
  else if 임의의 레벨의 엘리먼트 체크=TRUE then
       return {체크된 엘리먼트 + 체크된 레벨 주 키 +
               체크된 레벨의 부모 레벨들의 주 키 +
               루트 레벨의 주 키}
}
```

<표 1>의 알고리즘을 바탕으로 생성되는 파생 DTD의 예제를 살펴보면 다음 [그림 3]과 같다.

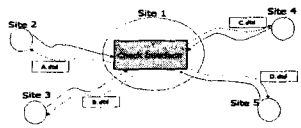


[그림 3] 파생 DTD 생성 예제

[그림 3]은 레벨 3의 fieldtest 엘리먼트의 하위 엘리먼트인 testdepth의 엘리먼트만으로된 DTD를 생성할 때, 인터페이스이다. 여기서 살펴보면 체크된 레벨의 주 키인 fieldtestnum와 상위 레벨의 주 키인 boringnum 그리고, 루트 레벨의 주 키인 projectnum가 묵시적으로 포함된다. 이렇게 함으로써, 체크된 엘리먼트의 레벨과 위치를 알 수 있고, 각 레벨 사이의 관계성을 알 수 있다.

3.5 파생 DTD 참조방안

본 논문에서는 DTD를 참조할 때 외부참조를 한다. 왜냐하면, 내부참조를 하게 되면 DTD의 변경이 발생할 때 XML 문서를 수정해야 한다는 단점을 가지기 때문이다. 이러한 단점을 극복하기 위해 외부참조를 한다. 그리고, 한 사이트에 여러 사용자의 DTD를 둘 경우, DTD를 참조할 때, 한 사이트에서 요구되는 많은 부하를 줄이기 위해서 사용자가 생성한 DTD는 사용자의 머신에 둔다.



[그림 4] 파생 DTD 참조방안

[그림 4]에서 보는 바와 같이 각 사용자 요구의 DTD는 사용자의 지역 머신으로 가져오게 된다. 그리고, 사용자가 질의한 조건에 따르는 적격 XML 문서를 파생 DTD로 외부 참조시켜 유효한 XML 문서 형태로 만들어서 DOM API의 ParseError 메소드를 사용하여 이 XML 문서가 유효한지 체크한다[5]. IE(Internet Explorer) 5.0에서 XML 문서를 파싱할 때에는 IE 5.0이 그 문서가 DTD를 가지는 유효한 XML 문서의 형태를 가지더라도 그 문서가 시작태그와 종료 태그가 일치하면 DTD에서 명시한 태그를 사용하지 않더라도 파싱을 하기 때문에 반드시 DOM API의 ParseError 메소드를 사용해서 파싱해야 한다. 이렇게 DOM API에 의해 파싱된 XML 문서는 유효하고 정확하다는 것을 보장받을 수 있다.

4. 결론 및 향후 연구방향

본 논문에서는 시추공 DTD를 원본 DTD로 하여 여러 가지 파생 DTD를 생성, 관리하여 XML 문서의 효율적인 검증을 가능하게 하는 시스템을 설계하였다. 이렇게 함으로써, 원본 DTD에서 파생 DTD를 생성하여 지금까지의 적격 XML 또는, 원본 DTD를 따르는 유효한 XML 문서만을 데이터 교환의 수단으로서만 사용하는 시스템이 아닌 사용자 요구에 맞는 파생 DTD를 가지는 적격 XML 문서를 사용한다. 그래서, 사용자 정의에 따르는 많은 XML 문서를 데이터로서 사용할 수 있고, 사용자 요구에 따르는 적격 XML 문서의 유효성을 검증할 수 있으며 서버로부터 전송된 적격 XML 문서를 파생 DTD의 외부 참조로써 유효한 XML 문서로도 사용할 수 있는 여러 가지 장점을 가지고 있다. 그리고, 각 XML문서의 유효성과 정확성을 동시에 만족하는 효과를 가져온다.

향후 연구방향으로 파생 DTD 생성하는 모듈과 현존하는 DBMS 시스템들과 연계하여 복잡한 질의에 따르는 적격 XML 문서의 생성과 사용자 질의에 맞는 파생 DTD를 자동 생성하는 방향으로 연구가 필요하다.

참고문헌

[1] Minos Garofalakis, Aristides Gionis, Rajeev Rastogi, S.Seshadri, Kyuseok Shim. "A System for Extracting Document Type Descriptors from XML Documents", Proceedings of the 2000 ACM SIGMOD, 2000.
[2] Dave Hollander, Trevor Jenkins, "XML Applications", Wrox, 1998.
[3] W3C, Document Object Model (DOM) Level 2 Specification Version 1.0, 1999.
[4] Didier Martin etc.., "PROFESSIONAL XML VOLUME 2", Wrox, 2000.
[5] Benoit Marchal, "XML BY EXAMPLE", QUE, 2000.