XForms 기반의 UI 코드 자동생성 시스템 개발

이 은 정[†] · 김 태 훈[††]

요 약

XML이 데이터를 주고 받기 위한 수단으로 활발히 도입되면서 XML 데이터 관리를 위한 사용자 입력 폼 인터페이스의 자동생성 기술이 웹 기반 응용의 구축이나 웹 서비스 클라이언트 개발에서 주목받고 있다. 본 연구에서는 DTD의 XML 구조 정의에 기반하여 XForms 언어를 이용한 사용자 인터페이스 코드를 자동 생성하는 방법을 살펴본다. 본 연구에서는 계층적이고 반복적인 XML 데이터의 특징을 고려하여 반복부에 대한 삽입 삭제를 허용하는 반복부 편집 행위 모델을 제안하고 이를 바탕으로 XForms 코드 생성 알고리즘을 기술하였다. 또한 생성된 코드는 새로운 웹 폼 표준 언어인 XForms를 목적 언어로 하여 MVC(Model, View, Control) 구조를 가진다. 제안된 방법을 검증하기 위하여 Orbeon 프리젠테이션 서버 플랫폼 상에서 동작하는 코드생성기 XFormsGen 시스템을 개발하였다.

키워드 : 입력폼, 사용자 인터페이스, 코드 자동생성, XML, XForms

XFormsGen : XForms-Based Automatic UI Code Generator

Eunjung Lee[†] · Tae-Hun Kim[††]

ABSTRACT

As XML is more proliferate as a data type between internet programs, automatic generation methods of user input form interfaces get more attention. Recently, generating user interfaces for web application or web service clients become very popular. In this paper, we study the method of automatic generation for XForms-based user interface codes based on XML structure definitions in DTD. With the repetitive edit action model, we present a formal generation method of input form codes and the overall user interface pages. For the generated code is using XForms as a target language, the result is clear and efficient with the MVC structure. Also, we have developed XFormsGen system, which implements the proposed method over the Orbeon presentation server.

Key Words : Input Forms, User Interface, Code Generation, XML, XForms

1. 서 론

최근 XML 사용이 일반화되면서 XML 기반의 어플리케이션이 늘어나고 있다. 그 결과 XML 데이터를 생성하거나 수정하는 폼 기반 사용자 인터페이스(User Interface : 이하 UI)의 사용도 증가하고 있다. 폼 기반 UI를 개발하기 위해서는 XML 데이터의 타입과 사용자 행위를 정의하고 인터페이스를 설계하여 코드로 작성하는 단계를 거치는데, 어플리케이션 개발에서 이러한 UI 개발 부분이 큰 비중을 차지한다[12].

UI 개발의 부담을 덜기 위해 XML 데이터 정의로부터 UI 코드를 자동 생성하는 방법이 많이 연구되었다. XML 데이터 정의에서 UI 코드를 자동으로 생성하는 것은 전체 개발 시간을 단축시켜 줄 뿐 아니라, XML 데이터 정의가 변하는 경우에도 다시 생성될 수 있다는 장점을 가진다. 웹기반 응용을 위한 XHTML이나 WML의 자동 생성이나 자바 UI 코드의 자동 생성 시스템이 많이 발표되었다[8,12,14]. 또한 최근에는 웹서비스 클라이언트를 손쉽게 개발하기 위하여 WSDL 호출부의 일부로 입력 폼 UI의 자동 생성을 지원하는 플랫폼들이 있다[26]. 한편 UI의 일반적인 모델링을 위한 언어로 UIML, XIML 등이 제안되었고 모델링 언어의 기술에서 프로그래밍 언어의 코드를 자동 생성하는 방법이나[2, 19] 일반 소프트웨어 개발 모델과 같이 UML을 이용해 설계하고 코드를 자동 생성 방법도 연구되었다[6].

본 논문에서는 XML 데이터 정의로부터 수정과 삽입/삭제 기능을 제공하는 UI의 자동 생성을 다루는데 코드 생성의 목적 언어로 XForms 언어를 사용한다. XForms는 XML 기반의 UI 기술 언어로 W3C에서 2003년에 표준으로 제정되었다[21]. XForms는 XML 문서를 데이터 모델로 가지며 행위와 데이터 전송을 포함한 입력 폼을 기술할 수 있는 언어이다. 이 언어는 기존에 인터넷에서 많이 사용되던 HTML 폼과 스크립트 언어를 대치할 것으로 기대되지만 기술 보급은 아직 초기 단계이다. XForms 코드는 폼 컨트롤

† 정 회 원 : 경기대학교 정보과학부 조교수
†† 준 회 원 : 경기대학교 대학원 게임웨어학과 석사과정
논문접수 : 2005년 8월 24일, 심사완료 : 2005년 9월 29일

과 데이터 처리 행위를 기술하고 페이지 레이아웃과 프리젠테이션은 XHTML이나 XSL[26]과 같은 스타일 컨테이너 언어에 의존한다[22]. 본 연구에서는 미리 정의된 행위 모델과 스타일 템플릿을 이용하여 XForms 코드와 XSL 파일을 함께 생성한다.

본 연구에서 제안하는 방법은 기존의 접근과 몇가지 점에서 차이가 있다. 첫째, 데이터 편집 행위 모델을 제시하여 XML 데이터 정의로부터 폼 컨트롤 및 UI의 생성을 위한 정형화된 알고리즘을 제시하였다. 둘째, HTML의 후속으로 차세대 웹 폼 기술 언어로 꼽히는 XForms를 목적 언어로 사용하여 활용성을 높였다. 셋째, XML의 반복, 중첩 구조를 반영하는 템플리트를 사용하여 사용자가 원하는 스타일의 코드가 생성 가능하도록 하였다. 제안된 방법을 검증하기 위하여 XForms 코드의 실행 환경인 Orbeon 사의 프리젠테이션 서버 플랫폼[21] 상에서 코드 자동생성기 XFormsGen 시스템을 구현하였다.

이 논문의 구조는 다음과 같다. 2장에서는 관련 연구를 살펴보고 3장에서는 XML 및 편집 행위 모델을 소개한다. 4장과 5장에서는 개발된 XFormsGen 시스템의 설계 및 구현 결과를 소개하고 6장에서 결론을 맺는다.

2. 관련 연구

2.1 UI 코드 자동 생성 기술

일반적인 UI 설계를 위해 광범위한 UI의 구성요소와 기능을 표현할 모델링 언어에 관한 연구가 많이 있었는데, 모델링 언어를 이용하여 설계된 내용을 코드로 자동 변환할 수 있다. UI 모델링 언어로 UI의 요소들을 표현할 수 있는 XML 기반의 언어들이 많이 제안되었다[9, 11, 13, 17, 20, 21]. 대표적인 것으로 UIML[9]과 모질라 브라우저에서 사용되는 XUL[17], 그리고 W3C에서 표준으로 제시된 XForms가 있다[20]. UIML은 UI의 모든 요소와 사용자 상호 작용을 표현할 수 있으며, 사용자가 기술한 모델링 언어의 명세에 따라 다양한 플랫폼과 단말, 그리고 목적 언어에 대해 코드를 자동생성할 수 있다. UIML은 범용적인 UI 기술 언어로 어떤 UI든지 기술할 수 있어, 다양한 미디어 폼과 사용자의 UI 설계 의도, 그리고 모든 플랫폼과 활용 분야에 적용가능한 강력한 표현력을 가진다. 이에 비해 XForms와 XUL은 인터넷 웹 폼을 기술하기 위한 제한된 목적으로 설계된 언어로, 원격 시스템과의 XML 데이터 인터페이스를 표현하는데 사용될 수 있다.

또 한가지 웹 폼 기술 언어로 들 수 있는 것은 닷넷 환경의 Web Forms이나 IBM의 XML Forms가 있다. 이 기술들은 시각적 구성 요소를 가지는 웹 폼 페이지와 각 폼 컨트롤의 처리를 위한 코드 부분으로 나누어진 구조를 제공한다. 그러나 이들은 코드 자동 생성이라기보다 API 기반의 개발 플랫폼이라 할 수 있다[27, 28].

한편 XML 기반의 데이터 관리를 위한 UI 생성은 적용 대상 분야를 한정하여 일관된 사용자 인터페이스를 사용할 수 있고 XML 문서 정의를 기반으로 사용자 행위 유형이 예측 가능하여 코드의 자동 생성이 가능하다. Lay 등은 XML 스키마에서 자바 스윙 UI를 자동 생성하는 방법을 소개하였다. 이들은 XML 정의에 따라 데이터 값의 수정, 삽입/삭제 행위를 생성하며, 데이터 타입의 처리와 반복부 및 선택부의 처리를 위한 방법을 소개하였다[11]. 이들의 연구는 본 연구와 유사한 접근 방법을 사용하고 있으나 데이터 및 행위 모델에 대한 설계 없이 XML 구조에 직접 매핑시키고 있어 중첩된느 반복부가 있는 경우 생성되는 UI 페이지가 복잡하여 활용성이 떨어진다. 다른 예로 아파치 오픈소스의 하나로 제공되는 Xydra는 웹서비스 호출부의 코드를 생성해 주는 시스템으로[26] 위해 XML 스키마로부터 데이터 입력을 위한 폼 인터페이스를 XHTML 코드로 자동생성한다. 이 시스템은 터미널 값에 대한 다양한 타입을 지원하고 템플리트을 이용한 UI 레이아웃과 스타일 처리를 지원하는 면에서 본 연구에서 개발하는 XFormsGen 시스템과 유사하다. 그러나 단순한 XML 트리의 구조를 가정하여 중첩된 반복구조를 지원하지 못한다.

2.2 XForms 표준

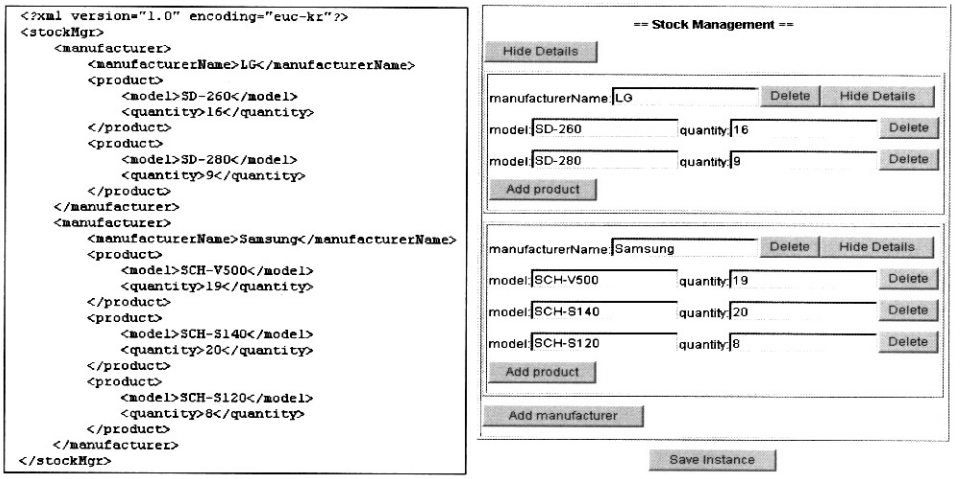
XForms는 W3C에 의해 2003년 권고안으로 발표된 인터넷 웹 폼 언어로 XML을 지원하면서 기존의 브라우저 용 클라이언트 스크립트가 하던 사용자 입력과 데이터 전송 기능을 담당한다[20]. XForms 코드는 보통 모델과 사용자 UI 부분으로 나누어지며, 모델은 XML 데이터 인스턴스를 사용한다. UI 부분은 XML 데이터에 바인딩되는 폼 컨트롤과 submit 액션 등을 포함하게 된다. 또 UI 부분은 XSL이나 XHTML 등의 폼 컨테이너 스타일 언어를 사용하여 전체의 레이아웃이나 스타일을 지정한다. (그림 1)은 XForms를 이용하는 웹 페이지의 예를 보여준다.

XForms 페이지는 XML 문서를 입력으로 받아 사용자 행위에 의해 새로운 또는 수정된 XML 페이지를 생성한다. (리스트 1)의 코드는 (그림 1)의 화면에 대응하는 XForms 코드이다. 이 폼 페이지는 XML 데이터의 구조에 의해 결정되는 데이터 편집창과 삽입 삭제 버튼을 가진다.

XForms 코드를 포함한 문서는 일반적인 브라우저에서 지원되지 않으므로 클라이언트 측에서 전용 처리기를 설치하거나 서버에서 HTML 코드로 변환시켜 주는 방법이 사용되고 있다. 대표적으로 브라우저의 플러그인으로 사용될 수 있는 FormsPlayer[16]와 서버 변환 방식인 Orbeon Presentation Server 시스템[18]이 있다. 본 연구에서는 생성된 XForms 코드의 실행 환경으로 Orbeon 사의 플랫폼을 사용한다.

3. 데이터 및 편집 행위 모델

본 논문에서 다루는 XML 데이터는 DOM 트리로 표현된다. 트리는 요소 노드와 속성 노드로 이루어지며, 요소 노드는 이름을 가지고, 자식이 없는 터미널 요소 노드는 값을



(그림 1) XML 데이터와 XForms 폼 인터페이스 페이지

```
<xforms>
<xforms : model>
<xforms : label>home</xforms : label>
<xforms : submit><xforms : label>Save </></>
<xforms : repeat nodeset="제조사">
        <xforms : input    ref="./@name">
          <xforms : label>제조사명</> </>
        업체구분 :
         <xforms : select1   selection="closed">
               <xforms : item>
                     <xforms : label>국내</>
                     <xforms : label>해외</>
           </></>
          <xforms : input    ref="./@phone">
            <xforms : label>제조사명</>
          </>
```

```
<xforms : repeat nodeset="제품">
    <xforms : input ref="./stockMgr : model">
       <xforms : label>모델</></>
    <xforms : input ref="./stockMgr : quantity">
       <xforms : label>수량</xforms : label>
     </>
     <xforms : input ref="./stockMgr : option">
       <xforms : label>옵션</>
     </>
     <xforms : submit>
       <xforms : label>제품삭제</>
       <xforms : delete nodeset="제품set"
         position=".../index('제품set')"/></>
</>
    <xforms : submit>
       <xforms : label>제품추가</> // 이하생략
```

(리스트 1) (그림 1)의 재고 관리 화면 XForms 코드

가진다. 속성 노드도 이름과 값을 가지므로 트리 구조를 다룰 때는 터미널 노드로 취급될 수 있다. 단, 속성 노드는 나열형이나 필수 여부 등 요소 노드와 다른 타입 정보를 가진다.

XML 데이터 타입은 DTD나 XML 스키마에 의해 정의되는데, 본 연구에서는 DTD를 사용한다. DTD는 XML 스키마에 비해 데이터 타입 지원이 제한되지만, 트리의 구조적 요소를 좀더 분명하게 보여줄 수 있어 명료한 알고리즘의 기술을 위해 유리한 장점을 가진다. (그림 2)는 위의 XML 데이터에 대한 문서 정의를 보여준다. (그림 2) (나)의 DTD 그래프는 터미널 요소와 속성의 타입 정보를 생략하고 DTD 구조를 표현해주는 자료구조로 사용된다. DTD 그래프 생성 방법은 [12]에서 정의된 바를 따른다. DTD 그래프에서 기호 노드는 (*)와 같이 표시한다.

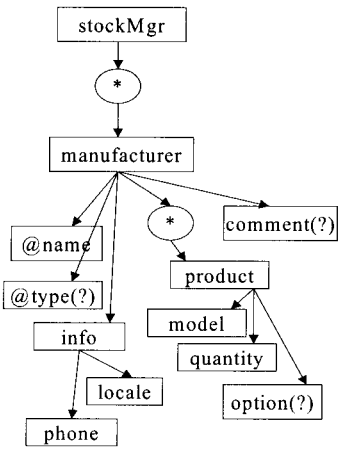
XML 데이터를 생성 또는 수정하는 UI의 사용자 입력 폼은 주어진 XML 트리에서 터미널 데이터 값의 수정과 부분 트리의 삽입 및 삭제 기능을 가진다. 한편 XML 트리는 계층적, 반복적 구조를 가지는데, 사용자 입력 폼은 (그림 1)에서 보는 바와 같이 중첩된 레이아웃이나 확장 축소 기능을 지원한다. 사용자 입력 폼의 코드는 사용자 행위를 지원하는 폼 컨트롤 부분과 레이아웃과 스타일을 지원하는 프리젠테이션 부분으로 나누어 볼 수 있다.

폼 컨트롤은 사용자가 XML 트리를 수정하기 위한 편집 행위에 대응된다. 편집 행위는 XML 트리의 비교 분야에서 많이 사용되는 개념으로 XML 트리에 대한 변경을 데이터 수정(Update), 부분 트리 삽입(Insert) 및 삭제(Delete) 행위로 모델링한다[12]. 부분 트리 삽입 및 삭제 편집 행위 모델은 트리 비교의 목적이나 편집 환경에 따라 다르게 정의되는데, 본 연구에서는 부분 트리의 삽입 삭제를 반복부에 대해서만 허용하는 *반복부 편집 행위 모델*을 제안하고 이를 바탕으로 사용자 입력 폼과 UI 코드를 생성하고자 한다. 반복부 편집 행위 모델은 XML 데이터에 대한 편집 행위로 아래에 정의된 반복부 편집 행위만을 허용하는 모델이다.

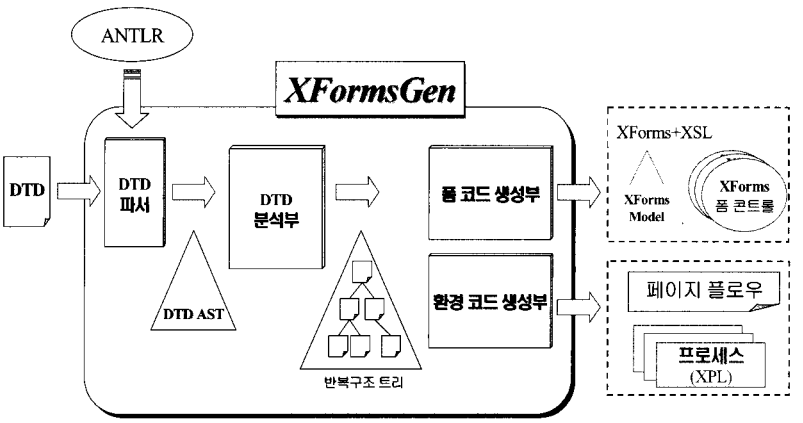
[정의 1] 주어진 XML 트리에 대해 다음과 같은 *반복부 편집 행위*가 정의된다.

(i) Update(p, $value_{new}$), node(p)는 터미널 노드이다.
(ii) ListInsert(p, Tnew), node(p)는 반복부 부모 노드

```
<!ELEMENT stckMgr (manufacturer)*>
<!ELEMENT manufacturer
                (info, product*, comment?)>
<!ELEMENT product (model, quantity, option?)>
<!ELEMENT info (address, phone)>
<!ATTLIST manufacturer
        name CDATA #REQUIRED>
<!ATTLIST manufacturer
        type CDATA #IMPLIED>
<!ELEMEENT phone #PCDATA>
<!ELEMEENT address #PCDATA>
<!ELEMEENT model #PCDATA>
<!ELEMEENT quantity #PCDATA>
<!ELEMEENT option #PCDATA>
<!ELEMEENT comment #PCDATA>
```



(그림 2) 재고관리 예의 DTD와 DTD 그래프



(그림 3) XFormsGen 시스템의 구성도

이고 Tnew는 마지막 노드로 삽입된다.

(iii) ListDelete(p), node(p)는 반복부 노드이다.

반복부 편집 행위 모델을 이용하면, 삽입 삭제 등 트리 구조 변경 행위가 반복부에 대해서만 제공되므로 일관적이고 예측가능한 UI 코드를 설계할 수 있어 XForms 코드 및 컨테이너 스타일 파일을 자동 생성할 수 있다. 또한 사용자의 행위를 제약하지 않으면서 결과 XML 데이터의 유효성을 보장할 수 있게 된다.

반복부 중심의 편집 행위 모델을 적용하기 위해서는 XML 문서가 반복부 중심 구조를 가지도록 정의되어야 한다. DTD 그래프에서 (*) 노드가 형제 노드를 가지지 않고 자식 노드로 유일한 요소 노드를 가지면 DTD가 *반복부 팩토링*되었다고 한다. 이것은 반복부 편집 행위 모델을 정의하기 위한 조건으로 일반 DTD에 대해 가상 노드를 도입하여 반복부 팩토링할 수 있다[1]. 반복부 팩토링된 DTD에 대해 유효한 XML 트리에서 *반복부* 노드란 DTD에서 (*)의 자식에 대응하는 요소 노드들을 의미한다. 또한 *반복부 부모* 노드는 반복부를 자식으로 가지는 노드이다.

한편 반복부 팩토링된 DTD 그래프에서 선택(|) 노드가 없다면 *단순 반복부 팩토링 DTD*라고 한다. 본 논문에서는 생성된 폼의 일관된 형태를 보장하고 사용하기 편한 인터페이스 제공을 위하여 단순 반복부 팩토링 DTD를 대해 먼저 고려하고 이후에 선택부의 확장 방법을 살펴본다. (|)로 표시되는 선택부를 지원하기 위해서는 추가의 사용자 행위가 필요하다. 예를 들어 사용자가 새로운 항목을 추가한다면 선택적인 구조를 가지는 경우 사용자가 먼저 해당하는 구조를 선택하고 그에 따라 생성된 폼을 이용할 수 있다.

4. XFormsGen 시스템의 설계

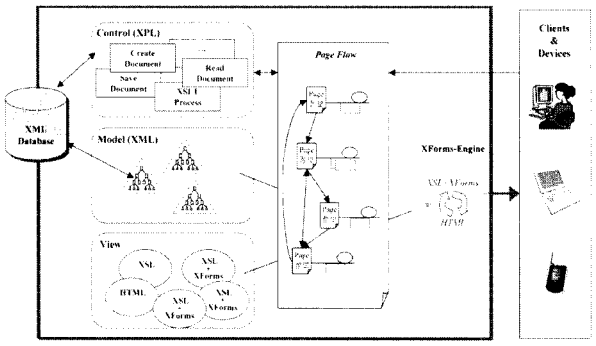
XFormsGen 시스템은 앞절에서 소개한 반복부 편집 행위 모델에 따라 XForms 코드와 XSL 컨테이너 페이지를 생성한다. (그림 3)은 XFormsGen 시스템의 구성을 소개한다. 이 시스템은 DTD를 입력으로 받아 생성된 AST로부터 반복부 정보를 추출하는 DTD 분석부와 분석 결과로부터 목적 코드를 생성하는 코드 생성부로 나누어진다. DTD 파서는 ANTLR 파서생성기를 이용하였다. 본 연구에서는 생성된 코드의 실행 환경으로 오번사의 프리젠테이션 서버 시스템을 사용하였는데, 아래에서는 생성된 코드의 실행환경인 오번사의 플랫폼을 소개하고 XFormsGen의 각 부분 시스템을

소개한다.

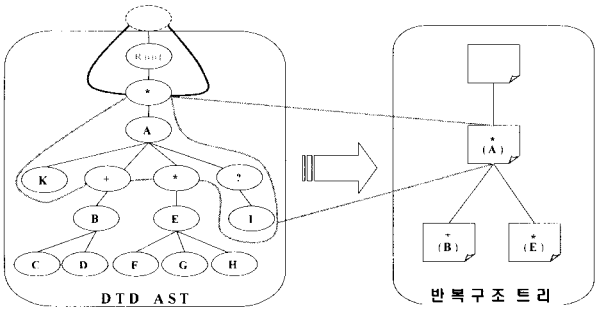
4.1 오번사 플랫폼의 XForms 처리 환경

Orbeon 사의 프리젠테이션 서버(Orbeon Presentation Server, 이하 OPS)는 XForms 처리기의 하나로 서버에서 XForms 파일을 변환하는 서버형 처리기이다[19]. (그림 4)는 OPS의 XML 데이터 처리 플랫폼 구조를 보여준다.

본 연구에서는 DTD로 주어진 데이터 구조 정의를 바탕으로 XForms 코드를 포함하는 XSL 파일을 자동 생성한다. 또한 편집 행위 모델에 의해 결정되는 시나리오에 따라 OPS 환경에서 동장하는 XPL 파일과 페이지 플로우 파일의 템플릿을 이용하여 DTD에 의존적인 부분을 자동으로 생성한다. 그 결과 DTD로 기술된 데이터 정의만 있으면 XForms Gen 시스템에 의해 OPS 환경에서 동작하는 XML 관리 어플리케이션을 사용자의 개입없이 자동으로 생성할 수 있다.



(그림 4) Orbeon 사의 프리젠테이션 서버 플랫폼



(그림 5) (그림 2)의 DTD에 대응하는 DTD 그래프와 반복구조트리

4.2 반복구조트리

XForms 페이지는 XML 문서에서 반복되는 데이터 부분의 구조와 터미널 노드에 대한 입력 컨트롤로 구성된다. DTD 분석부는 DTD를 분석하여 (*) 또는 (+) 노드로 표현되는 반복부를 구분하여 내부 자료구조인 반복구조 트리를 생성한다.

반복구조트리는 DTD에 표현된 반복부의 계층적인 관계와 터미널 노드에 대한 정보를 포함하는 자료구조다. (그림 5)는 (그림 2)의 재고관리 DTD에 대응하는 DTD 그래프와 반복구조 트리를 보여준다. 반복구조트리의 노드는 DTD 그래프에서 반복부 노드에 대응하는데 반복부 노드는 (*)와 (+) 노드 뿐 아니라 (?) 노드도 포함하며, 루트 노드에 대한 가상의 (*) 노드를 가정한다.

반복구조 트리 T_R의 생성 방법은 다음과 같다. 그래프 G를 DTD 그래프라 하고 T_R를 반복구조 트리라 하자. 매핑 m는 G로부터 T_R로의 노드 매핑으로 $v \in G$에 대해 $m(v)$는 v가 (*) 노드이면 T_R의 대응하는 노드이고 그 이외의 경우는 널 값을 가진다. T_R의 부모자식 관계는 G의 선행 관계를 그대로 가지되, (*) 노드 간의 사이클이 있다면 반복구조 트리의 노드들은 루트로부터 펼침 트리를 구성한다. 그러면 T_R의 노드 s에 대해 parent(s)는 다음과 같이 정의된다. m(v) = s이고 m(w) = parent(s)를 만족하는 DTD 그래프의 노드 w는 v의 조상 중 제일 가까운 (*) 노드 중에서 루트 노드로부터 경로가 가장 짧은 노드가 된다.

한편 DTD 그래프의 노드들 중에서 값을 가지는 터미널 노드는 대응하는 XForms 입력 컨트롤을 생성하여야 한다. 이를 위하여 반복구조트리의 노드를 중심으로 자손 터미널 노드들을 묶는다.

[정의 2] DTD 그래프 G에서 노드 v가 (*) 노드라고 하자. 노드 v의 직접 자손 터미널 노드 집합은 $DD_T(v)$로 *표시되며 다음과 같이 정의된다.* $DD_T(v) = \{w \mid$ (i) 노드 w가 v에서 도달가능하고 (ii) v에서 w까지의 경로에 다른 (*) 노드가 포함되지 않는다.$\}$

DTD 그래프에 사이클이 없다면 $DD_T(v)$ 집합에 의해 DTD 그래프의 터미널 노드들이 분할된다. (그림 5)의 DTD 그래프는 사이클이 없고 터미널 노드는 {K, I}, {C, D}, {F, G, H}와 같이 분할된다. 반복구조 트리 노드 s에 대해서 대응하는 DTD 그래프의 노드가 v라 하면, 즉 m(v) = s이면 s는 $DD_T(v)$에 속하는 노드들의 리스트를 가지며, 각 터미널 노드 w에 대해서 다음과 같은 속성이 정의된다.

(i) label(w)
(ii) relative_path$_v$(w)
(iii) date_type(w)

여기서 relative_path$_v$(w)는 v로부터 w까지의 XML 경로(요소레이블의 연속)로 정의된다. 속성 노드는 터미널 노드로 취급되며, data_type은 필수 여부와 열거형 타입 등 DTD 상에 나타난 타입 정보를 포함한다.

4.3 코드 생성부

XForms 코드에서 생성되어야 하는 폼 컨트롤은 터미널 데이터 값의 편집 컨트롤, 반복부 각 항목의 삭제 버튼, 반복부에 대한 추가 버튼이다. 또한 각 반복부에 대해 XML 데이터의 해당부를 반복시킬 수 있어야 한다. 이를 위하여 생성되어야 하는 XForms 코드는 (1) <xforms : input> 데이터 입력 컨트롤, (2) <xforms : submit> 삭제 및 추가 버

[XForms 코드생성 알고리즘]
입력 : T_R
출력 : XForms 코드
XForms 코드의 머리부를 생성한다.
T_R의 모든 노드에 대해 bind 태그를 생성한다. (노드의 절대경로와 레이블로 노드셋을 정의함)
T_R의 루트 노드를 v_R이라 한다.
  v_R의 모든 자식 노드 s에 대해
    *ProcessNode(s)*를 호출한다.
XForms 코드의 마무리 부분을 생성한다.

function ProcessNode(노드 s ∈ T_R)
(1) s의 반복부가 보이기 상태인가를 검사함 (<xsl : if> 태그와 s의 절대경로를 이용)
(1.1.) s에 대한 <xforms : repeat> 태그를 생성함 (<bind> 부분에서 정의한 nodeset을 이용)
(1.1.1) 모든 노드 u ∈ $DD_T(s)$에 대해
(1.1.1.1) <xforms : input> 태그를 생성함
  − ref 속성은 relative_path(s, u)로 하고
  −data_type(u)에 따라 입력 컨트롤 종류를 결정함
(1.1.2) s의 모든 자식노드 t에 대해
(1.1.2.1) ProcessNode(t)를 호출함
(1.1.3) 삭제 버튼을 생성함
  − <submit> 태그와 <delete> 태그를 생성함
  − <delete> 태그의 노드셋은 바인딩에서 정의된 노드셋과 index 함수를 이용함
(1.2) s에 대한 추가 버튼을 생성함
  − <submit> 태그와 <delete> 태그를 생성함
  − <insert> 태그의 노드셋은 바인딩에서 정의된 노드셋과 index 함수를 사용함
  − 삽입 위치는 형제들 중 제일 마지막 노드로 함

(리스트 2) XForms 코드 생성 알고리즘

```
<xsl:if test="/form/document/stockMgr:stockMgr/@product_details = 'DETAIL'">
                                  product 요소 부분이 [[보이기]] 상태이면
 <xforms:repeat nodeset="stockMgr:product" id="productSet">
                                  모든 product 요소에 대해 반복
   <xforms:input ref="./stockMgr:model">
   <xforms:label>model</xforms:label>
   </xforms:input>                 모델명에 대한 입력 컨트롤
                                   DDT에 대한 입력 컨트롤
     ...  중첩되는 반복부에 해당하는 부분 (재귀호출로 생성함)
     ...
   <xforms:submit xxforms:appearance="image">
     <xxforms:img src="/images/remove.png"/>
     <xforms:delete                          제품 항목의 삭제 버튼
           nodeset="/form/document/stockMgr:stockMgr/
                stockMgr:manufacturer[index('manufacturerSet')]/stockMgr:product"
                at="index('productSet')"/>

   <xforms:submit>
     <xforms:label>Add product</xforms:label>
     <xforms:insert                          제품 항목의 추가 버튼
           nodeset="/form/document/stockMgr:stockMgr/
                stockMgr:manufacturer[index('manufacturerSet')]/stockMgr:product"
           at="last()" position="after"/>
```

(그림 6) 자동생성된 코드 구조

튼 컨트롤, 그리고 (3) 반복부에 대한 <xforms : repeat> 또는 <xsl : for-each> 태그이다.

반복부 노드에 대해서는 부모 반복부 노드로부터의 상대경로와 절대경로가 필요하다. 절대경로는 보이기/숨기기 테스트와 삽입 삭제 처리부에서 사용된다. 단, <xforms : delete> 태그에서 nodeset 속성은 특정 노드를 표현하기 위해 전체 경로에 반복부 노드의 인덱스를 사용하고 있다. 이것은 OPS 플랫폼에서 <xforms : delete>를 지원하는 방식이며, XForms 처리기마다 다른 방식을 사용하고 있다.

중첩된 반복구조의 뷰를 효과적으로 관리하기 위하여 보이기/숨기기 버튼을 추가하였다. <xsl : if> 태그를 이용하여 해당 요소 부분이 보이기 상태인가를 검사한다. OPS 환경에서 XSL 코드의 변수를 지원하지 않아 XML 데이터의 속성을 이용하여 상태 값을 저장하였다.

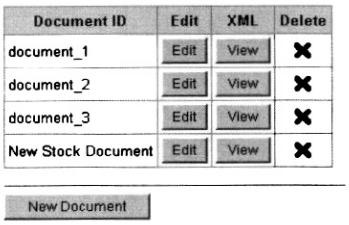
(그림 6)은 위의 알고리즘으로 생성된 XForms 코드의 예를 보여준다. 이 코드는 (그림 1)의 제품 관리 화면에서 product 반복부 부분에 해당하는 컨트롤들을 포함하고 있다.

계층적 반복부의 프리젠테이션 스타일을 생성하기 위해서 컨트롤 및 반복부의 레이아웃을 정해야 한다. 중첩된 반복부를 어떻게 표현할 것인가는 응용 어플리케이션에 따라 달

라져야 하므로 본 연구에서는 스타일 템플릿을 활용한다. 스타일 템플릿에 대해서는 5장에서 살펴본다.

5. 구현 결과

XForms 코드는 스타일 정보를 가지는 XSL 파일 안에 포함되어 생성된다. 폼 컨트롤의 코드는 4장에서 설명된 바와 같이 반복구조 트리를 따라 XForms 코드로 생성되고, 폼 컨트롤의 배치와 확장 축소 등의 사용자 인터페이스는 XSL 코드를 이용하여 작성된다.



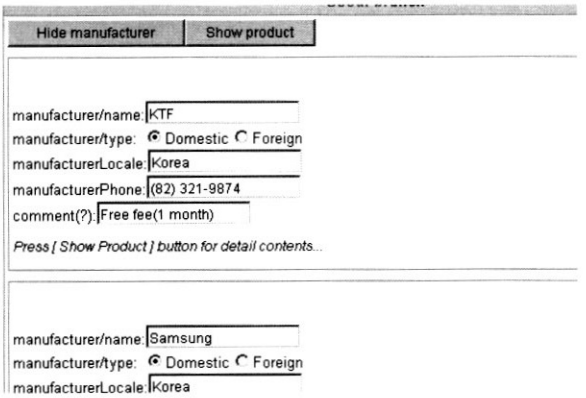
(그림 7) OPS 초기 화면

먼저 OPS 환경에서 원하는 문서를 불러오는 초기 화면이 (그림 7)과 같다. 선택된 문서는 XML 데이터베이스로부터 로드되어 인스턴스 트리로 메모리 상에 존재하며, 화면에 파일을 그대로 브라우징할 수도 있으며 수정 버튼을 누르면 데이터 수정을 위한 폼 인터페이스 페이지가 (그림 9)와 같이 나타난다. 이것은 XFormsGen 시스템에서 생성된 XForms + XSL 파일을 OPS의 XForms 엔진이 변환하여 클라이언트 브라우저에게 보내준 것이다.

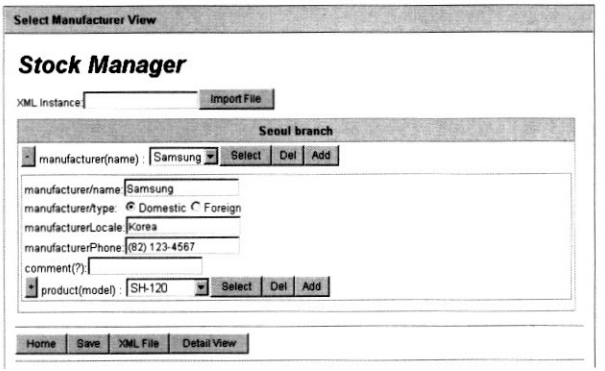
XFormsGen 시스템은 데이터 수정 페이지의 컨트롤과 반복부의 배치와 스타일 코드를 자동 생성하기 위하여 스타일 템플릿을 사용하였다. 계층적인 접근이 가능한 형태의 스타일 템플릿으로 하부 데이터를 모두 펼쳐서 보여주는 전개형 스타일 템플릿과 선택창에 의해 선택된 데이터에 대해서만 하부 데이터를 펼쳐서 보여주는 선택형 스타일 템플릿을 제공한다. 아래 (그림 8)과 (그림 9)은 각각 (그림 1)의 XML 기반의 재고 관리 응용에 대한 선택형과 전개형의 폼 화면을 보여준다.

전개형 스타일의 (그림 8)는 제조사 별로 표가 반복되고 제조사 안에 제품에 대한 행이 반복되는 형태이다. [Show product] 버튼을 누르면 제품 정보를 보여주는 행들이 보여진다. 계층적인 접근을 보이기 / 숨기기를 통해 제공하는 형태이다. 제조사의 삭제는 각 표의 우상단에 있는 X 버튼으로 제공되며, 해당 제조사의 제품을 추가하는 것은 각 제조사 별로 표의 하단에 [Add product] 버튼으로 제공된다. 한편 새로운 제조사의 추가는 제조사에 대한 표의 반복 후 아래 쪽에 해당 버튼이 있다. 이렇게 각 반복부에 대한 보이기/숨기기와 추가 삭제 기능을 중첩된 표 형태로 제공한다.

다음으로 (그림 9)는 선택형 스타일에서는 계층적 데이터 접근을 위해 선택 컨트롤을 이용하며, 계층 별로 데이터를



(그림 8) 전개형 스타일의 재고 관리 화면



(그림 9) 선택형 스타일의 재고관리 화면

선택하면 선택된 다음 계층의 데이터가 펼쳐진다. 각 계층의 반복부는 선택창으로 처리된다. 이 화면에서는 현재 제조사로는 삼성이 선택되었고 선택창에서 제품을 선택할 수 있는 상태이다. 여기서는 삽입과 삭제가 각 반복부에 해당하는 선택창 옆에 버튼으로 제공되며, 중첩된 데이터의 확장/축소는 반복부 선택창의 좌측에 +/- 버튼에 의해 제공된다.

6. 결 론

본 연구에서는 DTD를 이용하여 XForms로 기술된 사용자 인터페이스 코드를 자동 생성하는 방법에 관하여 살펴보았다. 본 연구는 기존의 사용자 인터페이스 코드 자동 생성 방법과 비교하여 몇가지 장점을 가진다. 우선 생성된 코드는 최신 웹 폼 표준 언어인 XForms로 작성되어 MVC(Model, View, Control) 구조를 가지며, XSL과 결합하여 데이터와 액션, 스타일을 분리할 수 있었다. 또한 본 연구에서는 코드 자동생성 방법을 체계화하기 위하여 계층적이고 반복적인 XML 데이터의 특징을 고려하여 반복부에 대한 삽입 삭제를 허용하는 반복부 편집 행위 모델을 제안하고 이를 바탕으로 XForms 코드 생성 알고리즘을 기술하였다.

제안된 방법을 검증하기 위하여 Orbeon 프리젠테이션 서버 플랫폼 상에서 동작하는 코드를 생성하는 XFormsGen 시스템을 개발하였다. 개발된 시스템은 다양한 템플릿을 이

용하여 사용자가 XML 데이터를 편집하기에 매우 효과적인 인터페이스를 자동생성해 준다.

향후 XFormsGen 시스템을 XML 스키마에 대하여 확장할 계획이며, UI 자동 생성 기술을 웹서비스 클라이언트와 결합하는 방법을 연구하고 있다.

참 고 문 헌

[1] 이은정, "XML 반복부 데이터 공유를 위한 리스트 잠금 프로토콜", 정보처리학회논문지D, 제11-D권 7호, pp.1367-1374, 2004년 12월.

[2] L.Baresi, et.al., "Modeling and validation of service-oriented architectures : application vs. style," ESEC/FSE'03, pp.1-5, Sep., 2003, Helsinki.

[3] J.Barton, T.Kindberg, et.al., "Sensor-enhanced mobile web clients : and XForms approach", WWW conference 2003.

[4] J.Bishop and N.Horspool, "Developing principles of GUI programming using views", SIGCSE'04, pp.3-7, March, 2004.

[5] John Boyer, Mikko Honkala : The XForms Computation Engine : Rationale, Theory and Implementation Experience. IMSA 2002 : 196-204.

[6] P. Campos, N. Nunes, "A UML-Based Tool for Designing User Interfaces, UML Modeling Languages and Applications," UML 2004 Satellite Activities,Lisbon, Portugal, pp.11-15, October, 2004.

[7] R.Cardone, et.al., "Using XForms to simplify Web programming," Proc. of 14th Inter. conf. on WWW, Tokyo, 2005.

[8] R. Engelen, et.al, "Developing web services for C and C++," IEEE Internet Computing 7(2), pp.53-61, 2003.

[9] James Helms, "UIML and XForms," http://xml.coverpages.org/ UIMLandXForms20020826.html, XML coverpages 웹사이트, 2002.

[10] Mikko Honkala and Petri Vuorimaa, "A Configurable XForms Implementation," Proc. of the the IEEE Multimedia Software Engineering, 2003.

[11] J.Shanmugasundaram, et.al, "Relational databases for querying XML documents : limitations and opportunities," In Proc. of 25th VLDB Conference, pp.302-314, 1999.

[12] P.Lay, et.al, "Transforming XML schemas into Java Swing UIs", WAM'04, 2004.

[13] R.N. Taylor, et.al., "A component- and message-based architectural style for UI software", IEEE Transactions on Software Engineering, 22(6) : 390-406, June, 1996.

[14] V.Turau, "A framework for automatic generation of web-based data entry applications based on XML," Proc. of 2002 ACM Symposium on Applied Computing, Madrid, 2002.

[15] Y. Wang, et.al, "X-Diff : A Fast Change Detection Algorithm for XMLDocuments", ICDE, pp.519-530, 2003.

[16] P.Terence, ANTLR 웹사이트, http://www.antlr.org.

[17] Mozilla, XML user interface language(XUL) 웹사이트, http://www.mozilla.org/projects/xul/

[18] Novell, "XForms Explorer," http://ftp.novell.com/pub/forge/xforms-explorer/docs/home.html.

[19] Orbeon Ltd., "Orbeon Presentation Server," http://www.orbeon.com.

[20] XForms version 1.1, W3C Working Draft 15 November 2004. htto//www.w3.org/TR/xforms11

[21] XML coverpages, "XML markup languages user interface definitions", http://xml.coverpages.org/userInterfaceXML.html.

[22] XPath version 1.0, W3C Recommandation 16 November 1999. http://www.w3.org/TR/xpath

[23] x-pot Ltd. FormsPlayer 웹사이트, http://www.formsplayer.com

[24] XSLT version 1.0, W3C Recommandation 16 November 1999 http://www.w3.org/TR/xslt

[25] X-Smiles 웹사이트, http://www.xsmiles.org.

[26] Xydra 시스템, http://www.extreme.indiana.edu/xgws/xydra.

[27] IBM XML Forms, http://www.alphaworks.ibm.com/tech/xmlforms.

[28] Microsoft Web Forms, http://msdn.microsoft.com/asp.net/.

이 은 정

e-mail : ejlee@kyonggi.ac.kr

1988년 서울대학교 계산통계학과(학사)

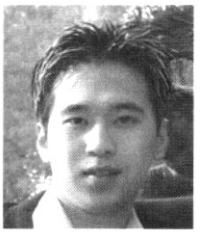
1990년 한국과학기술원 전산학과(공학석사)

1994년 한국과학기술원 전산학과(공학박사)

1994년~2000년 한국전자통신연구원 선임연구원

2001년~현재 경기대학교 정보과학부 조교수

관심분야 : 컴파일러 이론, XML 처리 기술 등



김 태 훈

e-mail : d75030@kyonggi.ac.kr

2004년 경기대학교 화학과(학사)

2004년~현재 경기대학교 대학원 게임웨어학과 석사과정

관심분야 : XML, 웹 서비스