웹 서비스를 위한 멀티 모달 사용자 인터페이스

송기섭° 김연석 이경호
연세대학교 컴퓨터과학과
{kssong°, yskim}@icl.yonsei.ac.kr, khlee@cs.yonsei.ac.kr

Multimodal User Interfaces for Web Services

Ki-Sub Song°, Yeon-Seok Kim, and Kyong-Ho Lee
Dept. of Computer Science, Yonsei University

요 약

본 논문에서는 웹 서비스의 WSDL 문서로부터 멀티 모달 유저 인터페이스를 동적으로 생성하는 방법을 제안한다. 이를 위해 W3C에서 제안한 사용자 인터페이스 관련 기술인 XForms와 VoiceXML을 소개하고, XForms에 기반한 사용자 인터페이스 생성 알고리즘을 제안한다. 제안된 방법은 WSDL 문서의 구조를 분석하고, 스키마로부터 데이터의 타입에 따른 적합한 컨트롤을 매핑하여 최적의 멀티 모달 사용자 인터페이스를 구성한다.

1. 서 론

웹 서비스(Web Services)는 플랫폼에 독립적인 애플리케이션 상호 운용을 가능하게 하는 기술로서, SOAP, WSDL등의 표준을 사용한다. 이러한 웹 서비스는 이종의 시스템 간의 통합 또는 상호 호환을 하는데 사용된다. 또한 웹 서비스는 사용자의 직접 접근을 요구하기도 하는데, 이 경우 사용자에게 적절한 사용자 인터페이스를 제공할 필요가 있다. 하지만 기존의 웹 서비스는 제한된 형태의 인터페이스만을 제공하기 때문에, 다양한 형태의 디바이스의 출현에 따라서 사용자의 편의성 측면의 한계를 가진다. 때문에 웹 서비스에 대한 편리한 사용자 접근을 위해 디바이스에 독립적인 멀티 모달(Multi-modal) 인터페이스를 제공해야할 필요성이 대두되고 있다.

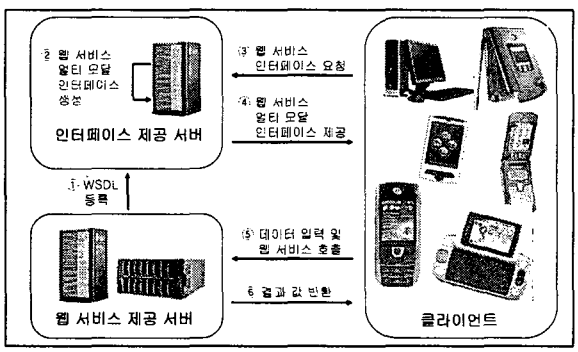

그림 1. 웹 서비스의 멀티 모달 인터페이스 제공을 위한 프레임워크

웹 서비스의 멀티 모달 인터페이스 제공 <그림 1>과 같은 형태로 구성된다. 웹 서비스를 제공하는 서비스 제공자가 인터페이스 제공 서버에 WSDL을 등록하면 인터페이스 제공 서버는 멀티 모달 인터페이스를 자동 생성하여 저장한다. 이후에 클라이언트의 요청이 있을 시, 인터페이스 제공 서버는 사용자 환경에 맞는 멀티 모달 인터페이스를 제공하여 클라이언트의 웹 서비스 접근을 가능하게 한다.

멀티 모달 사용자 인터페이스와 관련하여 W3C가 제안한 기술은 음성 기반 인터페이스 구현을 목적으로 하는 VoiceXML[1]과 웹 기반의 그래픽 사용자 인터페이스 구현을 목적으로 하는 XForms[2]가 있다. 이 기술들은 다양한 디바이스에서 웹 기반의 인터페이스를 제공하기 위한 기술이므로 웹 서비스를 위한 멀티 모달 인터페이스로서 이용 가능하다.

웹 서비스를 위한 사용자 인터페이스 생성에 관한 기존의 연구는 미리 정의된 GUIDD(GUI Deployment Descriptor)를 사용하여 WSDL로부터 HTML 폼(form)을 생성하는 방법[3]과 WSDL문서를 읽어들여 XForms 및 VoiceXML에 기반한 인터페이스를 생성하는 프레임워크[4]가 있다. 그러나 이 연구들은 서비스 제공자의 개입을 필요로 하거나, 현재까지 시나리오 제안 단계라는 측면에서 개선의 필요가 있다. 최근의 연구로는 DTD 정의로부터 관련된 XForms 인터페이스를 생성하는 연구[5]가 있다. 하지만 이 연구는 DTD에 기반하고 있으므로 단순한 규칙을 가진 스키마에만 적용할 수 있다는 한계를 가진다. 그러므로 훨씬 복잡한 구조를 정의할 수 있는 XML Schema를 사용하는 WSDL문서에 적용하기 위해서는 더욱 세밀한 규칙 정의가 필요하다.

본 논문에서는 웹 서비스의 멀티 모달 사용자 인터페이스를 생성하는 방법을 제안한다. 이를 위해 WSDL 문서를 읽어들여 XForms 기반의 사용자 인터페이스로 변환하는 규칙 기반의 방법을 제시한다.

2. 제안된 인터페이스 생성 방법

WSDL 문서로부터 XForms 기반의 사용자 인터페이스를 생성하기 위해 두 단계로 나누어 진행한다. WSDL 구조 분석 단계에서 인터페이스 생성에 필요한 정보를 추출하여, XForms 변환 단계에서 인터페이스를 생성한다.

※ 이 연구는 정보통신부(정보통신연구진흥원)에서 지원하는 2005년도 IT기초기술연구지원사업의 연구결과임.

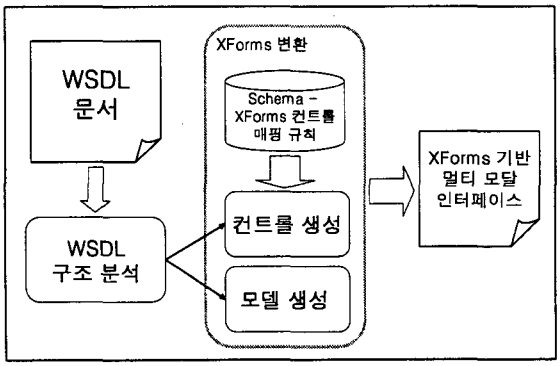

그림 2. 제안된 인터페이스 생성 방법

2.1 WSDL 문서 구조 분석

WSDL 문서 구조 분석 단계에서는 입력된 WSDL 문서를 DOM 트리를 생성하여 하향식 너비 우선 탐색 기법으로 분석하여 인터페이스 생성에 필요한 오퍼레이션 목록, 바인딩 주소, 메시지 타입, 스키마를 추출 한다.

2.2 XForms 모델 정의 및 컨트롤 생성

XForms는 코드의 식별성 및 데이터의 재사용성을 위해 사용자 인터페이스 정보와 데이터를 개념적으로 분리하여 다양한 디바이스의 지원을 가능하게 한다. 따라서 XForms 기반의 인터페이스를 생성하기 위해 모델과 컨트롤의 생성을 분리하여 개별적인 두 단계로 처리한다. XForms 모델 생성 단계에서는 웹 서비스의 호출을 위한 SOAP 메시지를 정의하
고, 데이터 전송을 위해 바인딩 주소를 서밋 정보에 추가한다.

웹 서비스의 호출시 서비스 제공자에게 전달되는 데이터는 사용자로부터 XForms 컨트롤을 통해 입력받는다. XForms는 사용자의 편의를 위해 다양한 형태의 컨트롤을 제공하고 있는데, 이러한 컨트롤은 전달되는 데이터의 형식에 따라 선택된다. WSDL 문서에서 서비스 제공자에게 전달되는 메시지 형식은 XML Schema[7]로 정의되어 있다. 따라서 제안된 방법에서는 XML Schema에서 정의된 데이터 형식에 따라 적절한 XForms 컨트롤을 선택하는 규칙을 제공한다.

XML Schema로 정의되는 데이터의 형식은 XML Schema 내장 데이터 타입, 사용자 정의 데이터 타입, 요소 내용 모델, 그리고 카디널러티 정의로 분류하여 처리한다. 각각 데이터 형식에 따라 제안된 규칙에 의해 XForms 컨트롤과 매핑된다.

• 심플 데이터 타입의 처리

XML Schema가 제공하는 44개의 기본 타입과 상속 타입에 대해서 W3C의 XForms 1.0 권고안은 input 컨트롤을 통하여 사용자 친화적인 입력 툴을 제공하게 한다. 따라서 내장 데이터 타입은 XForms의 input 컨트롤과 매핑 한다.

사용자 정의 데이터 타입은 <simpleType>으로 유도되는데, restriction, list, union의 3가지 기본 유도 형식으로 정의된다. restriction은 내장 데이터 타입으로부터 12가지의 facet을 통해 유도하는 형식으로, 패싯의 조합에 따라 적합한 컨트롤을 매핑 한다. restriction으로 유도된 형식은 다음과 같이 분류하여 규칙을 적용한다.

① string 또는 normalizedString 타입을 기본으로 하며 일정한 임계값보다 큰 값을 가진 minLength, maxLength, length의 패싯을 통해 정의된 타입은 textarea 컨트롤과 매핑 한다.

② enumeration 패싯을 통해 유도된 타입은 select1 컨트롤과 매핑 한다.

③ 시간, 정수형의 내장 데이터 타입을 기본으로 하여 maxInclusive, maxExclusive, minExclusive, minInclusive의 패싯을 통해 정의된 타입은 range 컨트롤과 매핑 한다.

④ 그 외의 restriction으로 유도된 형식은 input 컨트롤과 매핑 한다.

list형식은 하나 이상의 데이터들을 공백 단위로 구분하여 나열시킨다. list 형식으로 정의된 데이터 타입 중 유도되는 베이스 타입이 restriction 타입을 사용하여 enumeration 패싯으로 유도되었을 경우 XForms의 select 컨트롤과 매핑 한다.

```
...
<xforms:model>

ⓐ <xforms:instance id="message">
    ...
      <tns:payment ... >
        <tns:card />   } ②
        <tns:cash />   }
      </tns:payment>
    ...
  </xforms:instance>

ⓑ <xforms:instance id="selectIns">
    <tmp:temp selector="card"/>
  </xforms:instance>          ①

ⓒ <xforms:bind nodeset="tns:card" required="true()"
     relevant="instance('selectIns')//@selector='card'"/>
   <xforms:bind nodeset="tns:cash" required="true()"
     relevant="instance('selectIns')//@selector='cash'"/>
   ...
  </xforms:model>
  ...
ⓓ <xforms:select1
     ref="instance('selectIns')//@selector">
    <xforms:choices>
     <xforms:item>
      <xforms:label>card</xforms:label>
      <xforms:value>card</xforms:value>
     </xforms:item>
     <xforms:item>
      <xforms:label>cash</xforms:label>
      <xforms:value>cash</xforms:value>
     </xforms:item>
    </xforms:choices>
  </xforms:select1>
  ...
```

그림 3. <choice>로 정의된 모델로부터 생성된 XForms 코드의 예

• 복합 데이터 타입의 처리

XML Schema의 요소의 내용 모델은 <complexType> 또는 <group>을 통해 정의되며, <sequence>, <all>, <choice>, <group>선언을 사용하여 내용 모델을 규정한다. 각 선언에 대한 컨트롤 변환 규칙은 다음과 같다.

- <group> 또는 <complexType>으로 정의된 내용 모델은 XForms의 group 모듈로 치환한다.
- <all>, <sequence>로 정의된 내용 모델은 각 요소에 적당한 컨트롤로 변환한다. 이때 사용자 정의 데이터 타입에 대한 규칙을 적용하거나, 요소의 내용 모델에 대한 규칙을 재귀적으로

적용할 수 있다.

<choice>로 정의된 내용 모델은 전송될 요소를 사용자가 선택할 수 있어야 하므로, select1 컨트롤을 사용하여 전송하기를 원하는 요소의 데이터를 입력할 수 있는 컨트롤만을 선택할 수 있는 컨트롤을 생성한다. 이때 XForms 모델 내에 위치한 bind 요소의 relevant 속성을 이용하여 select1 컨트롤의 값에 따라 서로 다른 컨트롤을 출력한다.

<그림 3>은 <choice>로부터 생성된 모델이다. ⓓ의 select1 컨트롤의 선택에 따라 ①의 값이 변화하면 ⓒ에서 정의된 bind에 의해 ⓐ의 ②에 위치한 노드 중 하나가 활성화된다. 입력 컨트롤은 활성화된 노드에 따라 선택적으로 출력된다.

• 빈도 지시자 처리

XML Schema에서 요소 또는 요소의 내용 모델은 minOccurs와 maxOccurs 속성을 통해 카디널러티를 규정한다. 카디널러티가 규정된 타입은 XForms의 repeat요소와 insert, delete요소로 매핑한다.

```
...
ⓐ <xforms:model id="form_repeat">
    <xforms:instance>
      ...<NodeA>...</NodeA> ...
    </xforms:instance>
    <xforms:bind nodeset="NodeA" required="true()"
      constraint="count(.) &gt;= 2 and count(.) &lt;= 5">
    ...                                  ①
  </xforms:model>

  <xforms:group>
ⓑ <xforms:trigger id="insert_node">
    <xforms:label>Add</xforms:label>
    <xforms:action ev:event="DOMActivate">
      <xforms:insert at="index('NodeA')" position="after"
        model="form_repeat" nodeset="NodeA"/>
    </xforms:action>
  </xforms:trigger>

ⓒ <xforms:trigger id="delete_node">
    <xforms:label>Remove</xforms:label>
    <xforms:action ev:event="DOMActivate">
      <xforms:delete at="index('NodeA')" nodeset="NodeA"/>
    </xforms:action>
  </xforms:trigger>

ⓓ <xforms:repeat id="middle" model="form_repeat"
    nodeset="nodeA">
    [NodeA의 자식 노드들에 대한 컨트롤]
  </xforms:repeat>
  </xforms:group>
...
```

그림 4. 빈도 지시자에 대한 XForms 코드 생성 예

<그림 4>는 빈도 지시자로 정의된 스키마에 대한 XForms 코드의 예이다. ⓐ에서 정의된 모델은 ⓓ의 컨트롤에 바인딩 된다. "nodeA" 노드는 반복적으로 나타나게 되므로 repeat요소로 래핑되며, 사용자의 입력에 의해 동적으로 개수를 변화시킬 수 있어야 하므로 Add 버튼(ⓑ)과 Remove 버튼(ⓒ)을 생성한다. minOccurs와 maxOccurs의 값으로 제한된 빈도수 제한은 그와 관련된 xpath 구문을 bind 요소의 constraint 속성으로 지정하여 설정한다.

3. 실험 결과

제안된 방법을 이용하여 기존에 존재하는 웹 서비스의 WSDL 문서를 받아 웹 서비스의 유저 인터페이스를 생성할 수 있다.

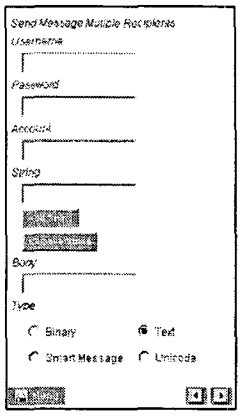

그림 5. 인터페이스 실행 화면 (모바일 브라우저)

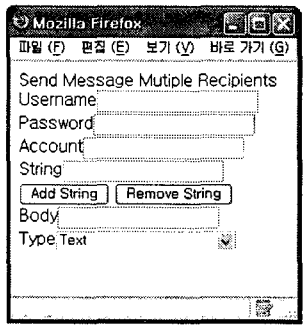

그림 6. 인터페이스 실행 화면 (웹 브라우저)

<그림 5>와 <그림 6>은 생성된 인터페이스를 모바일 브라우저와 웹 브라우저에서 실행한 화면이다. 이렇게 생성된 인터페이스를 통해 효과적으로 웹 서비스에 접근할 수 있다.

4. 결론 및 향후 연구

웹 서비스의 저변이 보다 확대됨에 따라, 웹 서비스로의 편리한 접근에 대해 관심이 높아지고 있다. 또한 다양한 디바이스의 등장에 따라 멀티 모달 인터페이스에 대한 요구가 증대되고 있다. XForms는 기존의 HTML 기반의 웹 폼을 개선할 수 있는 웹 기반의 인터페이스로서 XML기반의 데이터 송수신을 제공하기 때문에 웹 서비스의 인터페이스에 적합할 뿐만 아니라, 모델과 데이터를 분리함으로서 멀티 모달 인터페이스를 제공할 수 있도록 설계 되었다. 이에 따라 본 연구에서는 웹 서비스의 기술 문서인 WSDL 문서로부터 XForms 기반의 인터페이스를 자동 생성하는 알고리즘을 제안하여 다양한 디바이스에서의 웹 서비스 이용에 대한 편의성을 높였다. 향후에는 음성 통신 기반의 웹 서비스 접근 인터페이스를 개발하여 사용자의 편의를 더욱 증진시킬 계획이다.

5. 참고 문헌

[1] VoiceXML. http://www.w3.org/Voice/

[2] XForms. http://www.w3.org/MarkUp/Forms/

[3] R. Steele, K. Khankan, and T. Dillon, "Mobile Web Services Discovery and Invocation Through Auto-Generation of Abstract Multimodal Interface," Proc. Int'l Conf. Information Technology: Coding and Computing, Vol. 02, pp. 35-41, 2005.

[4] M. Kassoff, D. Kato, and W. Mohsin, "Creating GUIs for Web Services," IEEE Internet Computing, Vol. 7, No. 5, pp. 66-73, 2003.

[5] E. Lee and T. Kim, "Automatic Generation of XForms Code Using DTD" Proc. Fourth Annual ACIS Int'l Conf. Computer and Information Science, pp. 210-214, 2005.