# 동적인 RDF 문서 생성을 위한 XML 스키마

정의현 강수경° 조성윤
안양대학교 이공대학 디지털미디어 공학과
{jung,river°}@anyang.ac.kr, scho@aycc.anyang.ac.kr

An XML Schema for Dynamical RDF Document Generation

Eui-Hyun Jung, Su-Kyoung Kang° and Seong-Yun Cho
Dept. of Digital Media, Anyang University

**요 약**

시맨틱 데이터는 인간이 의도한 정보를 효과적으로 기계에게 전달하는 정보 양식이다. 그러나 시맨틱 정보를 웹 브라우저를 통해 CGI 코드로 처리하는 방식은 정보 시스템내의 정보구조와 이의 처리 코드가 고착되는 문제점을 안고 있었다. 이를 해결하기 위해 본 논문에서는 시맨틱 정보를 담는 RDF 술어를 기술하는 XML 스키마를 제안한다. 해당 스키마로 기술된 RDF 술어는 HTML 폼으로 자동으로 생성되며, 해당 폼에서 받아들인 데이터를 기반으로 대응되는 RDF 문서를 생성하는데 사용된다. 제안된 스키마를 이용함으로써, 정보 시스템 내에서 RDF 구조의 동적인 변경이 용이하며, 표준화된 방식의 RDF 데이터 입력이 가능해지는 효과를 얻게 된다.

## 1. 서 론

렉시컬(lexical) 기반의 기존 정보 시스템은 사용자가 원하는 정보에 정확히 접근하지 못하는 한계점을 드러내고 있다[1]. 이것은 다수의 컨텐츠 제공자가 독자적인 형식으로 정보를 제공하는 웹과 같은 분산 정보 시스템에서는 같은 의미의 정보라도 다른 형식으로 나타나는 것이 일반적이기 때문이다[2]. 예를 들어, 고양이에 대한 정보는 일반적인 단어인 "고양이" 혹은 "cat"이라는 단어를 이용해서 정보를 표시하지만, 일부 컨텐츠는 고양이를 "나비"나 "kitten"이라고 표시하는 경우도 있다. 이러한 렉시컬과 실제 정보를 담고 있는 시맨틱과의 불일치는 기존 정보 시스템에서는 해결하기 어려운 문제중의 하나이다[2]. 이를 해결하기 위해, W3C는 기존 웹 컨텐츠를 의미기반으로 접근할 수 있는 방법인 시맨틱 웹(Semantic Web)[2][3]을 제안하였다. 시맨틱 웹은 기존 웹 리소스에 시맨틱 정보를 추가하여, 기계가 해당 리소스의 의미를 이해하고 사용자가 의도한 정보에 접근할 수 있는 방법을 제시하고 있다.

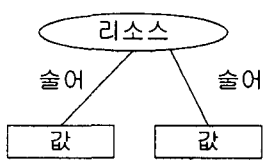
시맨틱 정보를 추가하는 과정을 시맨틱 태깅(semantic tagging)이라 하는데, 시맨틱 태깅의 목표는 리소스의 시맨틱 정보를 시맨틱 기술 언어(description language)로 기술하는 것이다. 이러한 기술에 중요하게 사용되는 것이 RDF와 같은 시맨틱 기술 언어이다. 특히 RDF는 표준적인 방식으로 많은 시맨틱 웹 컨텐츠의 기술 언어로 널리 사용되고 있다. RDF를 이용하여 웹 컨텐츠를 저작하기 위해서는 HTML의 메타 태그를 적용하거나, 부가적인 태깅 방식을 이용해서 웹 컨텐츠 생성시에 부가적인 정보로 추가해야 한다. 이러한 접근 방식은 웹 컨텐츠에 고정된 시맨틱 정보를 추가하는데는 적합하지만, 웹 사용자에게서 동적으로 데이터를 받아서 시맨틱 정보를 구축하고자 하는 경우에는 적절하지 않다. 이를 해결하기 위해서는 HTML 폼(FORM)에서 데이터를 받아서, RDF 문서를 동적으로 추가하는 방식을 이용해야 한다. 그러나 이러한 접근방안에서는 저장될 RDF 문서의 술어(Predicate)가 사전에 결정되면, 해당 술어에 대해서 HTML 폼과 그의 처리 코드가 고정되는 결과를 낳게 된다. 이러한 구조는 저장될 RDF 문서의 술어를 변경하는 경우에 정보 시스템 내에서 이에 동적으로 대응할 수 없는 단점을 가지게 된다.

이를 해결하기 위해 본 논문에서는 정보 시스템에 저장될 RDF 문서의 술어의 구조와 이에 대응되는 HTML 폼을 기술할 수 있는 XML 스키마인 Predicate Description 스키마를 설계하였다. 또한, 이 스키마로 기술된 XML 문서를 바탕으로 HTML 폼의 동적인 생성과 폼에서 입력받은 데이터를 RDF 문서로 저장할 수 있는 웹 컴포넌트를 구현하여, RDF 데이터 접근과 관련된 일관된 프레임워크를 제시하였다.

## 2. Predicate Description 스키마

### 2.1. HTML폼과 RDF 술어의 역할

RDF는 그림 1과 같이 트리 형태의 그래프로 시맨틱 정보를 나타내고 있으며, 해당 리소스의 술어는 그래프의 아크(arc)로 해당 값(value)은 다시 노드로 기록하도록 되어 있다[6]. 이 RDF 그래프는 [Resource, Predicate, Value]의 삼항(Triple)으로 나타낼 수 있으며, XML로 표현될 수 있다.

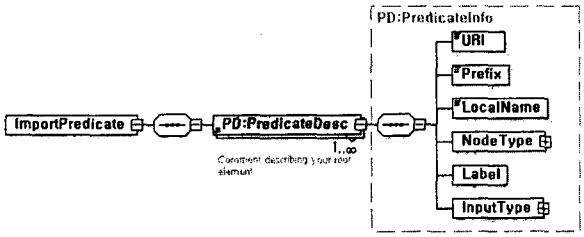



[그림 1] 리소스와 술어의 관계

일반적으로 리소스는 시맨틱 태깅의 대상이므로, 술어를 지정하여, 사용자로부터 해당 술어에 대한 값을 얻는 것이 HTML 폼에서는 적절하다. 그러나 이러한 방식에서는 사전에 필요한 술어들을 HTML 폼의 입력 컴포넌트들의 Name 태그에 지정하게 되고, 이를 처리하기 위한 CGI 코드를 연결하는 일반적인 웹 프로그래밍 방식을 사용하게 된다. 물론 이러한 방식으로도 동적인 RDF 문서를 만들 수 있지만, 대상(target) 술어 개개에 대한 수작업 코딩을 해줘야 하기 때문에, 목적 RDF 문서의 술어를 변경하고자 하는 경우에는 해당 코드를 모두 재구성해야만 하는 문제점을 갖게 된다.

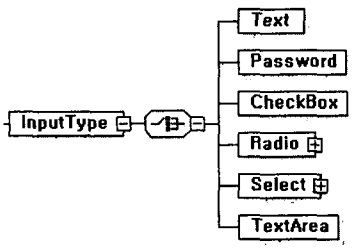
2.2. XML 스키마의 구성

제안된 XML 스키마는 그림 2에서 볼 수 있는 것처럼 여러 개의 술어를 기술할 수 있는 ImportPredicate를 최상위 엘레먼트로 갖고 있으며, 각 술어는 PredicateDesc 엘레먼트로 기술된다. PredicateInfo는 PredicateDesc의 데이터형인 복잡형(Complex Type)으로 URI, Prefix, LocalName, NodeType, Label, InputType을 갖고 있다.




[그림 2] PredicateInfo의 구조

PredicateInfo의 하위 엘레먼트들은 크게 2가지 그룹으로 나눌 수 있다. 첫 번째 그룹은 HTML 폼 그룹이다. HTML 폼 그룹에는 Label과 InputType 엘레먼트가 있다. Label은 단순히 HTML 폼에서 해당 술어의 입력 컴포넌트의 앞에 라벨로 나타난다. 이에 비해, InputType 엘레먼트는 술어의 값 입력을 위한 HTML 폼 입력 컴포넌트를 명시하는 역할로 사용된다. HTML 입력 컴포넌트는 TEXT, PASSWORD, CHECKBOX, RADIO, SELECT, TEXTAREA가 존재하며, 입력 컴포넌트에 대해 한 개의 입력 폼만을 선택할 수 있기 때문에, 그림 3과 같이 XML 초이스(choice) 형태를 갖게 된다.




[그림 3] InputType의 형태

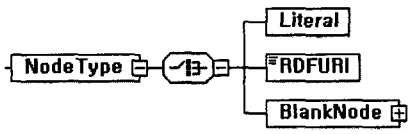
하위 엘러먼트의 두 번째 그룹은 술어의 저장에 관련된 그룹이다. RDF 문서에 술어를 저장하기 위해서는 해당 술어에 대한 정보가 필수적으로 필요하다. 표 1은 이 정보를 설명하고 있다.

| 엘레먼트 | 설명 |
|---|---|
| URI | 술어가 속한 온톨로 에서 정의된 유일한 값 |
| Prefix | 대상 RDF 문서에서의 술어의 Prefix |
| Localname | 대상 RDF 문서에서의 술어의 Localname |
| NodeType | 술어가 가질 수 있는 값의 형태를 명시 |

[표 1] RDF 문서 저장에 관련된 엘레먼트

2.3. 블랭크 노드의 처리

술어의 값으로 가질 수 있는 것은 RDF 표준에 의하면, Literal, RDF URI, 그리고 블랭크(blank) 노드가 될 수 있다[3]. 그림 4에서 볼 수 있는 것처럼, 제안된 스키마에서는 NodeType 엘레먼트에서 이를 기술할 수 있도록 하였다.




[그림 4] NodeType의 choice 값

Literal과 RDF URI는 하나의 값으로 표현되지만, 블랭크 노드인 경우에는 하나의 술어에 대한 값이 여러 개의 술어를 통해서 나타나기 때문에, 이런 하위 술어를 기술할 수 있어야 한다. 이를 처리하기 위해서, 스키마에서는 블랭크 노드의 타입인 BlankNode는 BlankNodeInfo라는 복잡형을 쓰도록 하였다.

```
<xs:complexType name="BlankNodeInfo">
<xs:sequence>
<xs:element name="GroupName" type="xs:string"/>
<xs:element name="SubPredicate"
  type="PD:PredicateInfo" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
```

BlankNodeInfo 복잡형은 하위 엘레먼트로 SubPredicate을 갖게 되는데, 이 SubPredicate은 PredicateInfo를 데이터형으로 갖게 되어, 하위에 다시 PredicateInfo 엘레먼트가 존재하는 재귀적인 형태를 갖게 된다. 이러한 재귀 구조를 이용함으로써, 하나의 술어가 다시 여러 개의 술어로 표시되는 블랭크 노드 구조를 표현하는 것이 가능해진다.

## 3. 평가

제안된 Predicate Description 스키마의 효용을 검증하기 위해, Dublin Core 온톨로지의 creator 술어와 RDF Pic 온톨로지의 lens 술어를 스키마를 이용하여 XML 문서로 다음과 같이 기술하였으며, 처리하기 위한 웹 컴포

넌트를 JSP에서 연동이 가능한 자바빈즈(Java Beans)의 형태를 갖도록 설계 및 구현하였다.

```
<?xml version="1.0" encoding="UTF-8"?>
<PD:ImportPredicate xmlns:PD="http://www.pd.org/2004/1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<PD:PredicateDesc>
 <PD:URI>http://purl.org/dc/elements/1.1/</PD:URI>
 <PD:Prefix>DC</PD:Prefix>
 <PD:LocalName>creator</PD:LocalName>
 <PD:NodeType>
  <PD:Literal/>
 </PD:NodeType>
 <PD:Label>저술자</PD:Label>
 <PD:InputType>
  <PD:Text size="20"/>
 </PD:InputType>
</PD:PredicateDesc>
<PD:PredicateDesc>
 <PD:URI>http://www.w3.org/2000/PhotoRDF/dc-1-0/</PD:URI>
 <PD:Prefix>photordf</PD:Prefix>
 <PD:LocalName>lens</PD:LocalName>
 <PD:NodeType>
  <PD:Literal/>
 </PD:NodeType>
 <PD:Label>렌즈 타입</PD:Label>
 <PD:InputType>
  <PD:Radio>
   <PD:Value checked="true">AF:70-210</PD:Value>
   <PD:Value>AF:80-230</PD:Value>
  </PD:Radio>
 </PD:InputType>
</PD:PredicateDesc>
</PD:ImportPredicate>
```

기술된 문서를 보면 알 수 있듯이, Dubline Core의 creator 술어는 문자입력 크기가 20인 텍스트 입력 컴포넌트를 사용하였으며, NodeType은 Literal형으로 기술되었다. 그리고 URI에는 Dublin Core 온톨로지에서 creator가 갖고 있는 실제 URI가 사용되었다. 이와 달리, RDF Pic 온톨로지의 lens 술어는 Radio 타입으로 지정되었다. 이렇게 지정되었기 때문에 Radio에 필요한 정보들이 <PD:Value> 태그로 지정되었다. 이렇게 기술된 내용은 다음과 같은 HTML 페이지를 생성하게 된다.

```
<form action="/cgi-bin/rdf_input.jsp?http://foo.edu/resource_uri">
  저술자: <input  name="DC_creator" type="text" size="10" />
  <br></br>
  렌즈타입: <input name="photordf_lens" type="radio"
            checked="true">AF:70-210</input>
  <input name="photordf_lens" type="radio">
            AF:80-230</input><br></br>
</form>
```

이 페이지에서 저술자에 "홍길동", 렌즈 타입을 "AF:80-230"으로 선택하게 되면, 이에 대응되는 RDF 문서는 다음과 같은 형태로 나오게 된다.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:DC="http://purl.org/dc/elements/1.1/"
xmlns:photordf="http://www.w3.org/2000/PhotoRDF/dc-1-0/">
<rdf:Description rdf:about="http://foo.edu/resource_uri">
  <DC:creator>홍길동</DC:creator>
  <photordf:lens>AF:80-230</photordf:lens>
</rdf:Description>
</rdf:RDF>
```

제안된 스키마로 각 정보 시스템에서 적절한 술어에 대해 기술함으로써, 해당 술어에 대한 데이터를 받기 위한 HTML 폼의 자동 생성과 그에 대응되는 RDF 문서의 생성을 자동화할 수 있었다. 이러한 결과는 본 스키마가 다양한 RDF 술어를 기존 정보 시스템의 수정없이 지원이 가능하며, 사용자로부터 RDF 술어에 대응되는 데이터를 받아서 이를 이용할 수 있다는 것을 보여준다.

## 4. 결론

시맨틱 정보를 컨텐츠에 추가하기 위한 여러 방법론이 있지만, 웹 상의 사용자로부터 시맨틱 정보를 동적으로 받아들이기 위해서는 HTML 폼을 이용해야 한다. 그러나 기존의 웹 프로그래밍 방식은 데이터의 구조가 HTML 폼에 고착되기 때문에, 입력 데이터의 형태가 고정되며 수작업을 통한 프로그래밍을 해야만 했다. 본 논문에서는 RDF 술어를 기술하는 새로운 스키마를 제안하였으며, 이 스키마를 이용하여 다양한 RDF 술어를 데이터로 받아들일 수 있는 자동화된 방법을 제안하고, 구현하였다. 이 스키마는 실제 적용 결과에서 우수한 자동화 결과를 보여주었으며, RDF 술어의 추가, 변동이 자유로움을 확인할 수 있었다. 이러한 RDF의 동적생성은 일반적인 클라이언트(thick-client) 형태의 정보 시스템에서도 이용될 수 있을 것이다.

[참고 문헌]


[1] Tirr, H.: Search in vain, challenges for Internet search. IEEE Computer. Vol.36. No.1. (2003) 115-116

[2] Berners-Lee.T., et.al, "The Semantic Web," Scientific American, Vol.284, No.5, pp.34-43, 2001

[3] Heflin, J., Hendler, J., "A Portrait of the Semantic Web in Action." IEEE Intelligent Systems. Vol.16. No.2. pp.54-59, 2001

[4] Dave, B., "RDF/XML Syntax Specification," W3C Working Draft 10, pp.4-20, 2003