메타데이터 관리를 위한 RDF 저작도구 설계에 관한 연구

최호찬* · 김차종**

A Study on the Design with RDF Authoring Tool for Metadata Management

Ho-Chan Choi* · Cha-Jong Kim**

요 약

오늘날 차세대 웹으로 부각되고 있는 시맨틱(semantic) 웹을 실현하기 위해서는 모든 웹 문서들이 메타데이터(metadata) 형식에서 정의하고 있는 시맨틱, 구문, 구조를 모두 수용할 수 있는 통합된 구조가 필요하다.

본 논문에서는 이를 위한 방법으로써 W3C에서 메타데이터의 기술과 교환을 위한 프레임워크로 개발한 시맨틱 웹의 핵심 기술 중의 하나인 RDF(Resource Description Framework) 문서로 변환하여 상이한 메타데이터들을 효율적으로 관리하고 이용할 수 있는 시스템 설계 및 구현에 대하여 연구하였다.

본 시스템에서는 Dublin Core 메타데이터를 이용한 RDF 생성, XML(eXtensible Markup Language) 문서를 RDF로의 변환, RDF를 NTriple 형태로 표현, 웹 서비스 메타데이터인 WSDL(Web Service Description Language)과 RDF와 통합 이렇게 4개의 모듈에 관하여 설계하였고, 자바를 사용하여 시스템을 구현하였다. 본 시스템을 이용하여 사용자는 세부적인 내용을 몰라도 메타데이터를 쉽게 통합하고 관리할 수 있다.

ABSTRACT

Recently, the semantic web stands out in the next generation web. To realize the semantic web, the metadata integration is required between different metadatas on the web for metadata management.

Otherwise, the integrated structure is needed to accommodate all of semantic, syntax and structure. RDF is one of the core technology and is more efficient framework for technology and interchange of metadata in the web, In this paper, for metadata management, we designed and implemented the RDF authoring system which converts each of metadata into RDF and makes it easy to manipulate and manage the different metadatas.

For this, we researched about the RDF creation using Dublin Core metadata, the conversion of XML document into RDF, the RDF expression by N-Triple form and the integration of WSDL and RDF, and implemented the system on the Java platform. Basically, users using this system can integrate and manage metadatas easily even if they are not expert.

키워드

RDF, XML, Dublin Core, WSDL, NTriple

* 한밭대학교 정보통신 전문대학원 컴퓨터공학과　　** 한밭대학교 정보통신컴퓨터공학부 교수

접수일자 : 2004. 1. 17

Ⅰ. 서 론

1. 본 연구 배경 및 목적

정보통신 발달 및 인터넷 사용 환경의 편의성 등으로 인하여 웹은 예측할 수 없을 정도로 빨리 성장하면서 웹의 사용이 일반화 되었을 뿐만 아니라 이미지, 음성, 영상 등 다양한 형식의 데이터를 서로 교환할 수 있는 정보의 바다로 부각되었다.

이와 함께 정보자원의 탐색 및 검색 기능을 지원해 주는 "데이터에 대한 데이터"라는 의미를 가지고 있는 메타데이터의 중요성도 커지고 있다. 이러한 메타데이터의 중요성의 인식과 더불어 다양한 메타데이터가 나타나고 있으며, 그 형식과 종류도 또한 매우 다양하다. 그러나 이러한 상이한 형식의 메타데이터들은 오직 사용자에 의해서만 정보의 의미를 이해하고, 해석할 수 있다. 그러나 차세대 웹으로 부각되고 있는 시맨틱 웹은 정보의 의미를 이해하고 의미를 조작할 수 있다. 즉 컴퓨터가 "정보의 의미"를 이해할 수 있도록 정보 리소스들의 의미가 정의되어있고, 의미적 연결성을 지원하기 위해 RDF를 사용한다. RDF는 웹 리소스들의 메타데이터를 표현하기 위한 데이터 모델로서, 컴퓨터간에 XML 데이터를 상호교환하고 XML 데이터를 활용할 수 있는 일반적인 구조를 제공한다. [1] 그래서 RDF를 이용하여 메타데이터들간의 통합이나 각각의 메타데이터 형식에서 정의하고 있는 의미, 구문, 구조를 모두 수용할 수 있다. 하지만 사용자가 RDF를 이용하여 메타데이터들을 통합하거나 교환하기 위해서는 RDF의 의미, 구문, 구조를 모두 이해해야 한다는 단점이 있다. 그래서 사용자가 구문적 복잡성을 이해하지 못해도 쉽게 상이한 메타데이터들을 통합하고, 이용할 수 있는 에디터가 필요하다.

따라서 사용자가 사용하기 쉬운 GUI 환경을 가진 RDF 에디터들이 새로운 대안으로 제시되고 있으며, 대표적인 시스템에는 IsaViz[2], Semtalk[3] 등이 있다. 하지만 아직까지 XML을 RDF로 변환하는 기능과 웹 서비스 메타데이터인 WSDL[4]과 RDF[5]의 통합 기능을 지원하는 RDF 에디터는 전무한 실정이다.

본 논문은 다양한 메타데이터들을 효율적으로 통합하고 관리할 수 있도록 XML을 RDF로의 변환, Dublin Core 메타데이터를 적용한 RDF 생성, RDF문서를 NTriple 형태로 표현, 웹 서비스 메타데이터인 WSDL과 RDF와의 통합 이렇게 4개의 모듈을 설계하고 구현하였다.

본 논문의 전체 구성은 다음과 같다. 제 Ⅱ 장에서는 인터페이스 표준에 대하여 설명하고, 제 Ⅲ 장에서는 시스템 개요 및 구성에 대해서 설명하고, 제 Ⅳ 장에서는 RDF 저작도구의 각 모듈 설계에 대해서 설명하였다. 제 Ⅴ 장에서는 기존 시스템과 구별되는 특징을 설명하고, 구별된 특징을 바탕으로 구현한 에디터에 대해서 설명하였다. 마지막 Ⅵ 장 결론에서는 메타데이터 관리를 위한 RDF 저작도구 설계의 문제점과 개선점을 제시하는 것으로 끝을 맺는다.

본 연구의 제한점은 자료를 기술하기 위한 메타데이터의 형식의 수가 급속히 증가하고 있으며, 그 상세성의 정도도 편차가 심하기 때문에 메타데이터 관리를 위해서는 다양한 형식의 기술요소에 대해서 검토가 요구되지만 본 논문에서는 WSDL과 Dublin Core 메타데이터, XML로 제한하였다.

Ⅱ. 인터페이스 표준

2.1 RDF

RDF[5]는 메타데이터의 기술과 교환을 위한 프레임워크로 W3C에서 개발한 시맨틱 웹의 핵심 기술중의 하나로써 다양한 메타데이터간의 공통적인 규칙을 지원하는 메커니즘을 통해 기계가 이해할 수 있는 형태로 의미 표현을 제공하기 위해 공통의 기술언어인 XML을 사용한다.

2.1.1 RDF Data Model

RDF 데이터 모델은 기술하는 자원의 속성과 자원 사이의 관계를 표현하며, 메타데이터의 교환과 통합을 지원하기 위한 모델로 자원, 속성타입, 속성 값으로 구성된다. 자원은 RDF 데이터 모델에서 기술되는 URI를 갖는 모든 객체를 말한다.

하나의 자원은 여러 속성 타입과 속성 값을 가질 수 있고 속성 타입은 자원의 속성 명을 의미하며, 속성 값은 속성 타입에 해당되는 값으로 문자열이나 자연어로 기술 될 수 있다. 그림 1은 위의 내용을 RDF 모형으로 표현한 것이다.

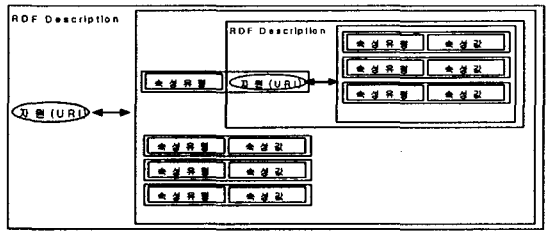

그림 1. RDF 모형
Fig. 1 RDF Model

RDF 기본 기술문(Statements)은 자원(Subject), 속성유형(Predicate), 속성 값(Object)으로 구성되며, 그림 2는 RDF 기본 기술문을 방향성 그래프로 나타낸 것으로서 http://www.w3c.org/Home/Lassila]), 속성유형([Creator]), 속성 값([Ora Lassila])을 그래프로 나타낸 것이다.

RDF 기술문을 RDF 데이터 모형 명세에서 제시하고 있는 노드와 아크 그래프를 이용해 도식화한 것은 메타데이터를 시각화하기 위해 유용한 반면 웹에서 애플리케이션을 공유하기 위한 명확한 방법을 제공하지는 못한다. [6]

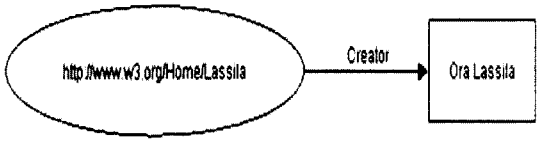

그림 2. RDF 데이터 모델의 구조
Fig. 2 Structure of RDF Data Model

그림 2의 RDF Data Model을 기술문으로 표기하면 "속성 값은 자원에 대한 속성유형의 값이다.", "자원은 속성 값과 속성유형을 가진다.", "자원의 속성유형은 속성 값이다."의 세 가지로 표현할 수 있다.

2.1.2 RDF 구문

RDF 데이터 모형은 메타데이터를 정의하고 사용하기 위한 추상적이고 개념적인 구조를 제공하는 것이며, 실제로 메타데이터를 작성하고 이를 상호교환하기 위해서는 구체적인 구문이 필요하다. 이 구문은 RDF를 통해 표현된 내용들을 기계가 이해할 수 있는 형태로 변환 할 수 있어야 한다.

RDF는 어의의 일관적인 표현을 위하여 XML을 사용하고 있는데, XML의 DTD는 사용하지 않고, XML의 정형화된 규정을 이용하고 있다.

RDF의 구문 표현 방법에는 그림 3의 연속구문(serialization syntax)과 그림 4의 축약구문(abbreviated syntax) 두 가지 형식으로 표현할 수 있다. 연속 구문은 속성 값을 엘리먼트로 기술하는 것이고, 축약 구문은 속성 값을 엘리먼트의 속성 값으로 기술하는 방법으로 HTML 문서 내에 RDF 메타데이터를 포함시킬 경우에 사용된다. 두 구문에서 RDF 데이터 모델과 RDF 파서(parser)에 의한 해석은 동일하다.[5]

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
    xmlns:s="http://description.org/schema/" >
  <rdf:Description rdf:about = "http://www.w3.org/Home/Lassila" s:Creator="Ora Lassila" />
</rdf:RDF>
```

그림 3. 연속 구문
Fig. 3 Serialization Syntax

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
    xmlns:s="http://description.org/schema/" >
  <rdf:Description rdf:about = "http://www.w3.org/Home/Lassila">
      <s:Creator>Ora Lassila</s:Creator>
  </rdf:Description>
</rdf:RDF>
```

그림 4. 축약 구문
Fig. 4 Abbreviated Syntax

2.2 WSDL

WSDL은 서비스 제공자가 보유하고 있는 서비스의 인터페이스를 XML을 사용하여 서비스 사용자들에게 제공하기 위한 웹 서비스 표준 기술언어로써 IBM, MS, Ariba에 의해 정의되어 있으며, 현재 WSDL 1.2 버전으로 W3C에 의해서 Working Draft한 상태이다.

WSDL을 살펴보면 웹 서비스의 정보가 추상적인 정의와 실제 구현 부분으로 나뉘어져 복잡한 구조로 되어있다. 이러한 구조는 WSDL에서 정의한 엘리먼트들의 재 사용을 목적으로 하기 때문이다. WSDL은 웹 서비스를 기술하기 위해 그림 5와 같은 구조를 가지고 있다. [4]

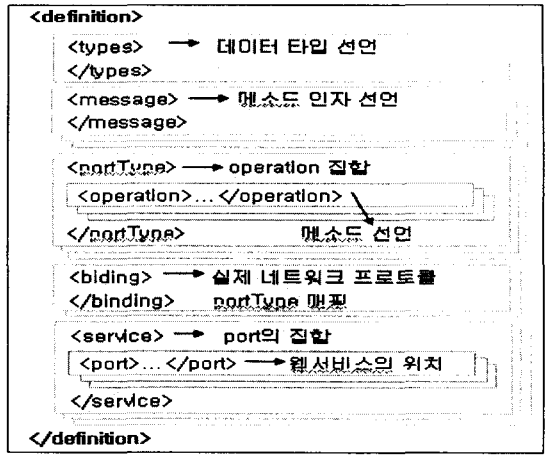

그림 5 WSDL의 구조
Fig. 5 WSDL Architecture

이 엘리먼트 중 message, operation, portType은 메소드 호출을 위한 추상적인 정의를 가지는 엘리먼트이며, 이러한 추상적인 정보를 가지는 엘리먼

트는 binding 엘리먼트를 통해 실제 프로토콜과 매핑된다. 이렇게 매핑된 정보는 port와 service 엘리먼트를 통해 실제 웹 서비스 위치와 연결된다.[7]

2.3 Dublin Core

Dublin Core 메타데이터 엘리먼트 셋은 간결하게 인터넷 자원의 필수적인 특징을 기술하고, 메타데이터간의 호환성을 제공하도록 설계되었다. Dublin Core 메타데이터는 HTML, XML 이나 RDF에 의해 이식된 메타데이터의 스키마를 기술한다. 이는 충분하지 않은 분류화와 방대한 정보에 의해 발생하는 문제들을 해결하는 한 가지 해법이 될 수 있다. Dublin Core에서 연구하는 대부분은 문법이나 구조보다는 의미의 정의에 더 많은 노력을 기울이고 있다. 이것은 특정한 구현 환경에 의해서 발생할 수 있는 제약을 피하면서 개념적 개발을 가능하게 하기 위해서이다.

Dublin Core는 인터넷 자원 검색을 용이하게 하도록 작성되는 메타데이터 요소의 집합으로 대상 자원의 형태에 관계없이 웹 기반의 메타데이터를 기술하기 위해 15개의 엘리먼트(Dublin Core 메타데이터 Element Set, version 1.1)를 정의하고 있다. 이 엘리먼트들은 자원의 내용과 관련된 요소, 지적 재산권과 관련된 요소, 자원의 물리적 표현과 관련된 요소로 구분되어 있다.[8]

III. 시스템 개요 및 구성

3.1 시스템의 개요

본 시스템은 RDF를 사용하여 웹 상의 메타데이터를 효율적으로 통합하고 관리하기 위한 시스템으로 JDK1.4.1과 JDOM Beta 9 [9], 시맨틱 웹 Application을 작성하는데 있어서 가장 신뢰 있고 많이 사용하고 있는 API로 Jena 2.0 [10]을 사용하여 구현하였다.

3.2 시스템 구성

그림 6은 시스템의 전체적인 구조를 나타낸 것으로 본 시스템은 Dublin Core를 RDF로 적용하는 부분, XML을 RDF로 변환하는 부분, WSDL과 RDF를 통합하는 부분, RDF를 NTriple 형태로 표현하는 부분 이렇게 네 개의 모듈로 구성된다.

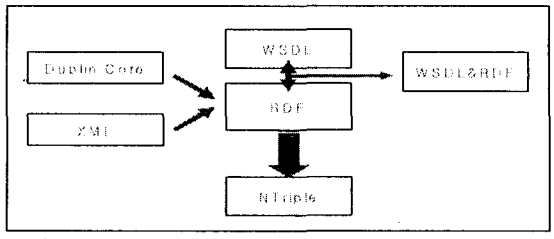

그림 6 시스템의 구조
Fig. 6 System Architecture

Dublin Core를 이용하여 RDF 문서를 생성하는 부분은 사용자가 원하는 Dublin Core 엘리먼트를 선택하여 RDF 문서를 생성하고, 생성된 RDF 문서도 엘리먼트를 선택하여 쉽게 수정, 삭제할 수 있다. XML을 RDF로 변환하는 부분은 XML 문서를 RDF 문서로 변환하는 기능과 각각의 문서를 RDF 에디터에서 쉽게 노드를, 수정, 추가, 삭제할 수 있는 기능을 지원한다. WSDL과 RDF를 통합하는 부분은 RDF를 WSDL로 통합하는 기능을 지원한다. RDF문서를 NTriple 형태로 표현하는 부분은 로컬에 있는 RDF문서를 NTriple로 표현하는 방법과 파일이 위치한 웹 상의 URI를 적어서 NTriple 형태로 표현하는 방법 두 가지 기능을 지원한다.

IV. 시스템 설계

4.1 Dublin Core 메타데이터를 이용한 RDF 생성

Dublin Core는 웹 문서의 의미를 표현하기 위해 메타데이터의 표준으로 주목할만한 시도였지만 다른 메타데이터 스킴들을 포함하지 않으며 모든 요구사항의 의미를 표현하는데 충분치 않다. 그러므로 Dublin Core 메타데이터를 RDF에 적용하기 위하여 우선 Dublin Core의 15가지 엘리먼트를 Dublin Core 스키마로 정의하고, 네임스페이스 기법을 이용하여 Dublin Core를 선언한 후에 실제 정보 자원의 기술을 위한 Dublin Core를 작성하여 RDF 문서를 생성하였다.

RDF 문서를 생성하기 위한 XML 파서로 DOM [11], SAX [12] 파서 대신 JDOM Beta 9 를 사용하였다.

4.2 XML을 RDF로의 변환

XML은 트리 형태의 문서로 고정되게 설계되어 인덱스 되기 때문에 메타데이터를 표현하기 위한 유연성이 부족하다. 반면에 RDF에서 노드는 인덱

스 되지 않고 URI를 갖는 자원이기 때문에 메타데이터를 표현하기 위한 유연성을 지원한다. 이와 같은 이유로 XML 문서를 RDF로 변환하였다. [10, 13]

XML 문서를 RDF로 변환 하기 위해서는 몇 가지 규칙을 적용 해야한다.

첫째는 RDF 문서는 XML 문서와 다른 네임스페이스를 사용하므로 RDF로 변환시 RDF 문서에 아래와 같은 네임스페이스를 추가하였다.

<xmlns:rdf="http://www.w3c.org/1999/02/22-rdf-syntax-ns#">

둘째는 XML 문서의 엘리먼트내에 속성명을 네임스페이스를 사용해서 정의한 경우 네임스페이스에 대해서 하나의 Description 엘리먼트을 만들고, XML 문서의 속성명과 속성 값은 RDF의 Description 엘리먼트내에 포함 시키도록 하였다.

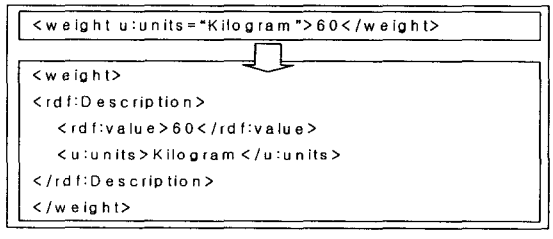

그림 7. XML 문서 변환 예1
Fig. 7 Example 1 of XML document conversion

위의 RDF를 그림 8처럼 다르게 표현할 수도 있다. 왜냐하면 RDF 문서의 속성 값이 한 엘리먼트에 여러개의 값 들을 포함하고 있으면 속성 값을 익명 리소스(anonymous resource)로 표현할 수 있다. 그러면 RDF 문서는 이 리소스에 대해서 설명할 필요가 없다. 그래서 이와 같은 경우는 익명 리소스를 이용해서 RDF 문서를 생성하는게 좋다.

그림 8은 rdf:parseType="Resource"를 이용해서 rdf:Description 엘리먼트를 만들지 않고 표현하였다.

```
<weight rdf:parseType ="Resource">
    <rdf:value>60</rdf:value>
    <u:units>Kilogram</u:units>
</weight>
```

그림 8. XML 문서 변환 예2
Fig. 8 Example 2 of XML document conversion

셋째는 XML에서 단순히 리터럴 값을 포함하고 있는 엘리먼트는 RDF에서 아래와 같이 표현 하면 된다. <paragraph rdf:parseType="Literal"> 이 지시자는 Resource/Property/Value의 NTriple 형태를 파싱하지 않는다.

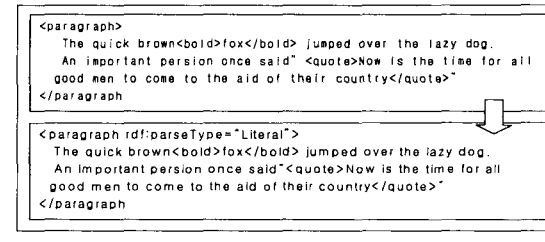

그림 9. XML 문서 변환 예3
Fig. 9 Example 3 of XML document conversion

4.3 WSDL과 RDF의 통합

WSDL은 네트워크에서 메시지를 주고 받는데 사용할 수 있는 서비스를 기술하기 위한 XML 형식의 언어이다. 현재는 WSDL을 이용하여 웹 서비스 사용에 필요한 정보를 이용하고 있다. 하지만 WSDL만을 이용해서 사용자가 원하는 정보를 사용하는 것보다 웹 리소스들의 메타데이터를 의미적으로 연결하기 위해 사용되는 RDF와 사용하면 좀더 효율적인 서비스를 이용할 수 있다.

그리고 WSDL과 RDF와의 통합 설계를 통하여 변환된 RDF&WSDL Description을 사용해서 그림 10과 같이 SOAP(Simple Object Access Protocol) 프로토콜을 통한 웹 서비스를 사용할 수 있고, RDF Query Engine에서 RQL(RDF Query Language)을 사용하여 질의 처리하고 온톨로지(Ontology) Crawler에서는 RDF Query Engine에서 제공하는 온톨로지를 검색하는 서비스를 이용할 수 있다. [14]

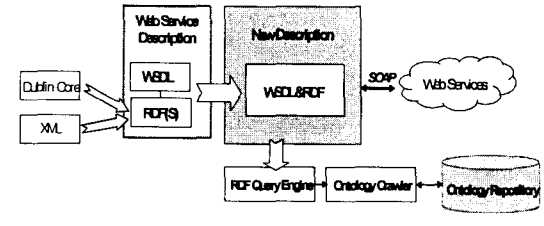

그림 10. WSDL&RDF 통합 서비스
Fig. 10 WSDL&RDF integration service

WSDL에 RDF 문서를 통합 하는 경우 웹 서비스에서 교환되는 데이터를 정의하기 위해 사용되는 WSDL의 message 엘리먼트에 RDF 문서를 포함한다. RDF는 rdf:RDF와 RDF 기술문을 포함하여 사

용할 수 있는 두 개의 엘리먼트들에 대한 XML 스키마는 WSDL 확장 엘리먼트로 사용할 수 없다. 그러나 RDF 기술문을 WSDL에 포함해야 하므로 WSDL 확장 엘리먼트로 새로운 엘리먼트를 만드는 것이 적절하다. 그림 11은 wsdl:newElement를 만들어 이용한 예제이다. 여기서 <WSDL:newElement>의 의미는 <rdf:Description rdf:about="wsdl 이 참조하고 있는 URI"/>이다.

```
<WSDL:message name="Price">
<WSDL:newElement>
  <ns:responsibleArchitect rdf:resource=mailto:ielab.hanbat.ac.kr />
</WSDL:newElement>
</WSDL:message>
```

그림 11. WSDL과 RDF 통합
Fig. 11 Integration of WSDL and RDF

4.4 RDF문서를 NTriple형태로 표현

RDF에서의 정보 표현은 주어(Subject), 동사(Predicate), 목적어(Object)로 구성된다.

NTriple에서도 아래와 같은 형식으로 RDF와 같은 정보를 표현한다. "<#chan><#knows><#kim>."

주어가 되는 모든 것은 URI로 식별되며, 앞의 예제와 같이 "#"이전에 URI가 생략되면 모든 참조 객체들은 동일 문서내의 것으로 간주한다. 목적어에는 이 값으로 올 수 있다. 동사인 know 부분은 RDF에서의 속성에 해당하며 주어와 목적어에 대한 관계를 표현하게 된다.

이런 몇 가지 규칙을 적용하여 RDF문서를 NTriple 형태로 표현하여 사용자가 RDF 문서를 쉽게 이해하고 접할 수 있도록 하였다. [10]

V. 시스템 구현

5.1 기존 RDF 에디터의 응용사례

본 절에서는 RDF에 관련된 에디터에 대해서 알아보고자 한다. RDF문서를 처리하는 도구에는 RDF 문서의 유효성을 검증하는 RDF 파서와 RDF 문서를 저작하는 저작도구, 브라우저 등으로 구성된다. W3C와 관련단체, 대학에서는 W3C 표준에서 제시하는 규격을 준수하는 RDF 관련 도구들을 개발하고 있다.

현재 사용하는 RDF 에디터로 그래픽한 기능을 기반으로 사용자가 실제 RDF 구문에 대한 자세한 이해가 없어도 작성할 수 있는 장점을 가진 IsaViz에 대해서 살펴 보고자 한다.

IsaViz 프로그램은 RDF 문서를 그래픽 환경에서 작성 할 수 있기 때문에 일반 사용자에게 구문적 복잡성을 모두 이해하지 않아도 되는 부담감을 덜어줄 수 있다. 낯설게만 느껴졌던 RDF 이용자들도 큰 고민 사항을 덜어 줄 것이다. 하지만 상이한 메타데이터와 RDF 문서를 통합하여 이용할 수 있는 부분이 없어 기존의 메타데이터를 시맨틱 웹과 활용하기에는 애로점이 있다.

본 논문에서는 이런 점을 감안하여 메타데이터를 통합하고 관리할 수 있는 RDF 에디터를 구현하였다.

5.2 RDF 에디터 구현

시맨틱 웹이 실현되기 위해서는 모든 웹 문서들이 메타데이터 형식에서 정의하고 있는 의미, 구문, 구조를 모두 수용할 수 있는 통합된 구조가 필요하며, 이를 위하여 메타데이터를 효율적으로 통합하고 관리할 수 있도록 RDF 에디터를 구현한 것이다.

본 시스템의 구현은 크게 네 개의 모듈로 구성되며 그 이유는 다음과 같다.

Dublin Core를 이용하여 RDF 파일 생성하는 부분은 기존의 변환 프로그램도 있긴 하나 15개의 Element를 정확하게 도출해 낼 수 없다. 따라서 이 부분을 지원할 에디터가 필요하다고 생각이 들어 손쉽게 Dublin Core를 이용하여 RDF 문서를 생성하고 수정할 수 있도록 하였다. XML을 RDF로 변환 하는 부분은 아직 구현된 프로그램이 없으며, 시맨틱 웹으로 발전해 가는데 뒷받침이 될 것이다. WSDL과 RDF 통합 부분은 통합으로 인해 효율적인 웹 서비스를 사용할 수 있고, 시맨틱 웹과 웹 서비스의 통합으로 신뢰할 수 있는 웹을 재 사용할 수 있다. RDF 문서를 NTriple 형태로 표현하는 부분은 RDF문서를 쉽게 분석하고 이해할 수 있다.[1]

다음은 시스템의 네 개의 핵심 구현에 대해서 설명하고자 한다.

5.2.1 RDF를 적용한 Dublin Core 관리자

RDF를 적용한 Dublin Core 메타데이터는 Dublin Core의 15개 엘리먼트들을 체크 박스와 에디트 박스를 이용해 사용자가 추가하고자 하는 엘리먼트들을 체크하여 해당 엘리먼트만을 생성, 수정, 삭제 할 수 있는 기능을 제공한다. 그리고 HTML의 <head> 엘리먼트 사이에 메타 태그를 삽입해 웹 문서의 데이터를 위한 정보 기능을 추가할 수 있는 기능을 제공한다.

그림 12의 화면에 나타난 Dublin Core의 15개 엘리먼트의 내용을 삽입하면 체크 박스의 체크 여부를 검토한 후 RDF 문서를 생성하게 된다.

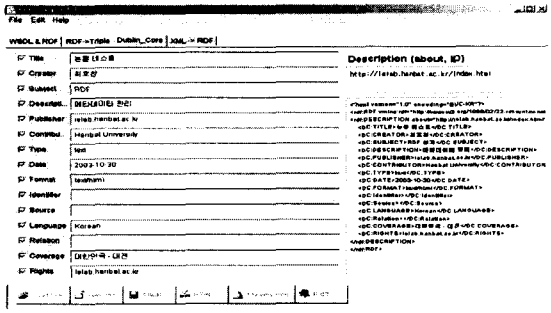

그림 12. Dublin Core 관리자 화면
Fig. 12 Manager Interface for Dublin Core

생성된 RDF 문서는 텍스트 박스로 보여줘 사용자가 쉽게 내용을 보면서 수정, 삭제할 수 있다. 그리고 생성된 RDF 문서를 불러오면 해당되는 엘리먼트와 Value가 RDF를 적용한 Dublin Core 메타데이터 관리자 화면의 체크 박스와 에디트 박스에 매핑되어 RDF 사용자는 쉽게 RDF 문서를 보면서 수정하고, 삭제할 수 있다.

본 시스템의 RDF를 적용한 Dublin Core 메타데이터에서 엘리먼트를 체크해서 저장한 다음 웹 문서인 HTML을 불러와 저장을 하면 HTML의 <head> 엘리먼트 사이에 메타 태그가 삽입되어, 검색의 정확성이 떨어지는 문제를 해결할 수 있다.

5.2.2 RDF 변환 관리자

RDF 변환 관리자는 XML을 RDF로 변환할 수 있는 기능과 XML과 RDF문서의 구조를 간단한 인터페이스를 통해서 트리 형식으로 보여주고, 트리 구조에서 마우스 버튼으로 쉽게 엘리먼트를 수정, 추가, 삭제할 수 있게 하였고, 텍스트 편집하는 기능을 추가하여 마우스로 선택한 엘리먼트의 값을 수정할 수 있는 기능을 제공한다.

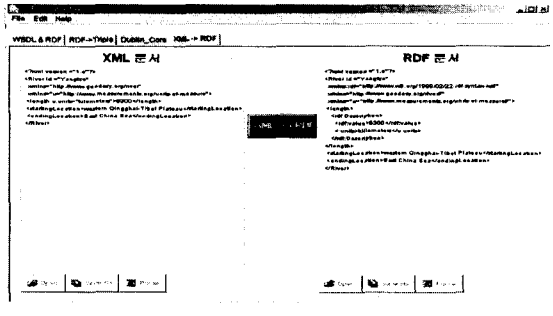

그림 13. RDF 변환 관리자 화면
Fig. 13 Manager interface for RDF conversion

그림 13은 XML문서를 RDF로 변환하여 XML의 객체의 배열 순서가 중요한 트리 지향 모델의 텍스트 마크업 모델로 노드가 인덱스에 포함되어 메타데이터를 표현하는데 유연성이 부족한 점을 해결하였다.

5.2.3 WSDL&RDF 통합 관리자

WSDL&RDF로 통합 관리자는 웹 서비스 메타데이터인 WSDL 문서와 RDF 문서를 통합 하는 기능을 제공하고 있다. 이 두 문서를 통합함으로써 지금 사용하고 있는 웹 서비스를 좀더 효율적으로 이용할 수 있다.

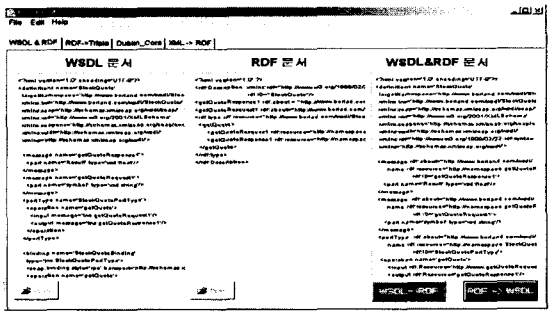

그림 14. WSDL&RDF 통합 관라자 화면
Fig. 14 Manager interface for WSDL&RDF integration

5.2.4 NTriple 표현 관리자

NTriple은 RDF · 로직 · 데이터를 하나의 언어로 표현하기 위한 최적화한 방법을 제공하기 위해 만들어진 것이다. 이것은 로직 언어를 이해하고 있다면 쉽게 접할 수 있도록 사용자 중심으로 설계되어 있다. NTriple 표현 관리자는 RDF를 NTriple 형태로 변환하여 RDF 문서를 쉽게 이해하고 분석할 수 있도록 하였다.

그림 15는 로컬에 있는 RDF 파일을 열어서 NTriple 형태로 표현할 수도 있고, 그림 16처럼 RDF파일이 있는 URI를 적어서 NTriple 형태로 표현할 수 있다.

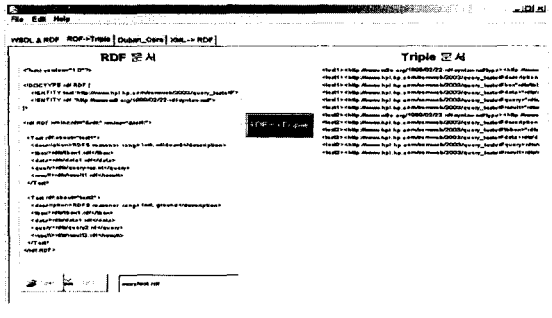

그림 15. RDF문서를 NTriple 형태로 표현(File)
Fig. 15 Expression of RDF document in NTriple form (File)

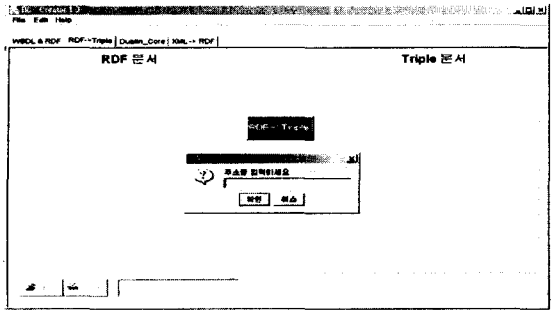

그림 16. RDF 문서를 NTriple 형태로 표현(URI)
Fig. 16 Expression of RDF document in NTriple form (URI)

VI. 결 론

본 논문에서는 시맨틱 웹이 차세대 웹으로 부각되면서 웹상의 메타데이터를 효율적으로 통합하고 관리할 수 있는 RDF 저작도구 설계 및 구현에 관하여 기술하였다.

본 논문에서 설계된 시스템은 RDF를 적용한 Dublin Core 관리자, RDF 변환 관리자, WSDL과 RDF의 통합 관리자, NTriple 표현 관리자 이렇게 네 개의 관리자로 구성되어 다른 시스템과 구별되는 본 시스템이 기존의 다른 RDF 에디터와 구별되는 특징은 다음과 같다.

첫째, 자바로 구현되어 특정 플랫폼에 종속적이지 않고, Dublin Core 메타데이터를 적용한 RDF 문서를 쉽게 생성, 수정, 삭제할 수 있다.

둘째, HTML 문서의 <head> 엘리먼트 사이에 Dublin Core 메타 태그를 삽입하여 웹 페이지 검색의 정확성이 떨어지는 문제점을 해결해 줄 수 있다.

셋째, XML의 데이터 구조적인 부분의 정의를 RDF로 변환하여 의미 있는 부분도 정의 할 수 있도록 하였다.

넷째, 웹 서비스 메타데이터인 WSDL과 RDF를 통합 하여 좀더 효율적이고 의미 있는 웹 서비스를 할 수 있도록 하였다.

본 시스템에서 Dublin Core를 이용한 RDF 문서를 생성하기 위해 XML 파서로 JDOM Beta 9를 사용하여 XML의 네임스페이스를 2개 이상 선언할 경우 발생하는 파싱 에러를 해결 하였다. XML을 RDF로 변환 하는 부분과 RDF문서를 NTriple형태로 보여주는 부분은 Jena 2.0 API를 사용하여 쉽게 구현할 수 있었다. 하지만 XML문서를 RDF 문서로 변환 하는 부분은 다양한 형태의 XML 문서를 RDF 형식으로 변환하는데는 성공하였으나 모든 형태의 XML 문서를 RDF형태로의 변환은 XML문서의 구조에 문제점이 있었다. 그리고 WSDL 문서와 RDF 문서와의 통합에 있어서 WSDL의 XML 스키마에서 표현할 수 없는 속성들을 RDF문서에서는 포함 하고 있기 때문에 부모 엘리먼트와 자식 엘리먼트가 때에 따라서는 같이 사용할 수 없는 등의 문제점이 있었다. 따라서 WSDL 문서와 RDF 문서를 통합 할 수 있는 스키마 표준이 제시되어야 할 것이다.

본 논문에서 구현한 방법은 지속적인 발전을 통해 보다 효율적인 방법으로 발전할 수 있을 것이다. 하지만, RDF를 이용하여 다양한 메타데이터를 관리하기 위해서는 먼저 RDF 기반 기술들이 표준화 되어야 하고, 시맨틱 웹이 활성화되기 위해서는 현재의 웹과 통합하여 사용할 수 있는 기술들이 정립되어야 한다.

참고문헌

[1] 신현성, "시맨틱 웹을 위한 RDF 편집기", 경기대학교 정보통신 대학원 석사학위 논문 pp. 40-44, 2002.

[2] IsaViz : A Visual Authoring Tool for RDF 「http://www.w3.org/2001/11/IsaViz/」

[3] Semtalk 「http://www.semtalk.com/」

[4] 이한수 "웹 서비스 실전 프로그래밍", 한빛미디어 pp. 117-139, 2002.

[5] RDF M&S Specification 「http://www.w3c.org/TR/REC-ref-syntax」

[6] 김학래, "생각하는 지능형 웹에의 도전, RDF 집중 분석" 「마이크로소프트」 pp. 253-254, 4. 2002.

[7] H.M.Deitel, P.J.Deitel, J.P.Gadzik, K.Lomeli, S.E.Santry, S.Zhang "JAVA WEB SERVICES FOR EXPERIENCED PROGRAMMERS" pp. 159-186, 2002.

[8] 조윤희, "RDF기반 인터넷 자원 메타데이터 설계에 관한 연구" 한국 보건 산업진흥원 정보 자료실 논문, pp158-160, 2000.

[9] JDOM : JDOM Beta 9 「http://www.jdom.org/」

[10] Jena 2.0 API 「http://jena.sourceforge.net/」

[11] DOM : W3C Document Object Model 「http://www.w3.org/DOM/」

[12] SAX : SAX 2.0 Extensions 「http://www.saxproject.org/」

[13] Roger L. Costello "XML Design" The MITRE Corporation
[14] RDF mailing list desc@w3.org

저자소개

최호찬(Ho-Chan Choi)

한밭대학교 컴퓨터공학과 ('02 공학사)
한밭대학교 컴퓨터공학과 ('04 공학석사)
※관심분야 : Semantic Web, XML, Web Service

김차종(Cha-Jong Kim)

광운대학교 컴퓨터공학과('84공학사)
광운대학교 컴퓨터공학과('86 공학석사, '91 공학박사)
Pittsburgh of University, USA 초빙교수 ('97.7~'98.7)
한밭대학교 정보통신·컴퓨터공학부교수(현)
※관심분야 : XML기반 Web Service, Semantic Web 컴퓨터그래픽스, Mobile Programming