RDF와 SPARQL을 이용한 PC내 영상 메타데이터 저장 및 검색

유영진O, 이병래
한국방송통신대학교
iteng@skcc.comO, brlee@knou.ac.kr

Storage and search of image metadata inside personal computers using RDF and SPARQL technology

Youngjin YouO, Byungrae Lee
Korea National Open University

요 약

본 논문은 영상의 메타데이터를 RDF를 이용하여 표현, 저장하고, SPARQL을 이용하여 검색하는 솔루션을 구현하여, RDF와 SPARQL을 통한 멀티미디어 파일의 메타데이터 관리 방안을 제시한다. 솔루션은 Jena2 프레임워크 기반으로 구현하였으며, Controller, EXIF Extractor, Metadata Schema, RDF Generator, Repository Manager, SPARQL Executor의 모듈로 구성된다.

1. 서론

휴대폰, 디지털 카메라 등의 디바이스가 대중화된 결과, 현재 일반인의 PC에는 수 많은 개인의 영상이 저장되어 있다. 이러한 영상들에 유용성을 부여하고, UCC 서비스와 같은 웹 2.0 시대의 킬러 어플리케이션을 발굴하기 위해서 영상에 대한 메타데이터에 도입과 관리가 필요하다.

본 논문에서는 개인의 PC에 저장되어 있는 영상의 메타데이터를 RDF를 이용하여 저장하고, SPARQL을 이용하여 검색하는 방안과 솔루션을 제안함으로써, RDF와 SPARQL을 이용한 영상 메타데이터 관리의 가능성을 제시한다. 2장에서는 관련 연구로 메타데이터의 개념과, 메타데이터를 표현하고 저장하는 방안으로 RDF, 메타데이터를 검색하는 방안으로 SPARQL, 이를 구현하기 위한 프레임워크로 Jena2를 소개한다. 3장에서는 본 논문에서 위 기술을 이용하여 구현하고 제시한 솔루션에 대하여 소개하고, 4장에서는 결론으로 연구의 의의와 향후 연구 방향을 제시한다.

2. 관련 연구

RDF 기반의 PC 내 영상 메타데이터 저장 및 검색 솔루션을 구현하기 위해 수행한 관련 연구를 간략히 소개한다.

2.1. 메타데이터

'메타데이터는 대상 개체를 정의, 발견, 평가, 관리할 수 있는 정보와 특징을 표현하는 구조화된 데이터이다.'

위는 WIKIPEDIA 백과사전을 참조한 메타데이터의 정의이다. 이 외에도 다양한 메타데이터의 정의가 존재하며, 'data about data'라고 간결하게 정의할 수도 있다. 본 논문에서는 영상의 메타데이터를 '영상을 정의하고 묘사할 수 있는 유용한 정보를 가지고 있는 부가적 정보'라고 정의한다.

영상의 메타데이터는 생성 방법에 따라 다음 표와 같이 구분할 수 있다.

[표 1] 영상의 메타데이터 구분

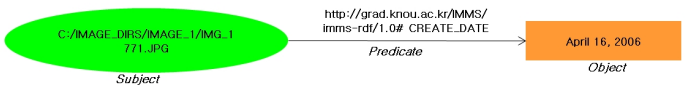
구분	입력 방식
촬영 인물, 촬영 장소 등처럼 사람이 직접 입력하는 컨텍스트 정보	사람에 의한 수동 입력
사진의 EXIF(Exchangeable image file format)와 같이 생성시 시스템(카메라)에서 자동으로 입력하는 정보	시스템에 의한 자동 입력
온톨로지의 상관관계, 계층관계 등을 이용한 의미적, 논리적 추론을 통해 얻어진 2 차적 정보	
Feature Extraction 과 같이 자동적으로 영상의 특징을 추출하여 얻어진 정보	

2.2. RDF(Resource Description Framework)

RDF 는 W3C 에서 개발한 메타데이터의 표현과 교환을 위한 프레임워크이며, 자원에 관한 정보를 표현하는 언어이자, 구조화된 메타데이터의 생성, 교환, 재사용 등을 가능하게 해주는 기반구조이다. 본 논문에서 RDF 는 영상 메타데이터를 서술하고, 저장하는 방법을 제공하는 핵심적인 요소이다.[1]

RDF 는 자원 (Resource), 특성 (Property), 문(Statement)의 세가지 중요 개념이 있다. 자원은 RDF 에서 표현되는 모든 대상으로서, URI 를 통해 식별되며, 웹 페이지와 같은 웹 자원으로부터 인간과 같은 웹과 관련 없는 자원까지 모두를 대상으로 할 수 있다. 특성은 책과 인간의 관계인 Author 와 같이 자원 사이의 관계를 표현한다. 마지막으로 문(Statement)은 특정 자원(Subject)의 특성(Predicate)을 어떤 값(Object)으로 진술한 것으로서, "Subject + Predicate + Object(다른 자원이거나 리터럴)"로 정의할 수 있다. 예를 들면, "C:/IMAGE_DIRS/IMAGE_1/IMG_1771.JPG has a Create-Date whose value is April 16, 2006"이 문이다.

RDF 의 데이터 모델은 노드와 아크의 그래프로 표현이 되는 그래프 모델이다. 위에서 예로 표현한 문은 다음 그림과 같이 그래프 모델로 표현할 수 있다.



[그림 1] RDF 그래프 모델

위 RDF 그래프 모델은 다음 표와 같이 Triple 모델로 표현할 수 있다.

[표 2] RDF Triple 모델

Subject	Predicate	Object
C:/IMAGE_DIRS/IMAGE_1/IMG_1771.JPG	http://grad.knou.ac.kr/IMMS/imms-rdf/1.0#CREATE_DATE	April 16, 2006

다시 RDF 그래프 모델을 아래와 같이 XML 기반의 RDF 구문으로 표현할 수 있다.[2]

[표 3] RDF 구문

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:imms="http://grad.knou.ac.kr/IMMS/imms-rdf/1.0#" >
  <rdf:Description
rdf:about="C:/IMAGE_DIRS/IMAGE_1/IMG_1771.JPG">
    <imms:CREATE_DATE>2004:04:16
21:01:54</imms:CREATE_DATE>
  </rdf:Description>
</rdf:RDF>
```

2.3. SPARQL

SPARQL 은 RDF 그래프 모델로 표현된 메타데이터로부터 원하는 정보를 추출할 수 있는 W3C 가 제안하는 쿼리 표준이다. SPARQL 을 통해 다음 사항의 수행이 가능하다.[3]

- URI, blank nodes, literal 로 표현된 형태로부터 정보를 추출
- RDF 서브 그래프 추출
- 쿼리 결과로부터 새로운 RDF 그래프 구성

다음 표는 Triple 모델로 표현된 간략한 RDF 그래프로부터 제목을 추출해 내는 SPARQL 쿼리의 예제이다.

[표 4] SPARQL 의 간략한 예제

```
<http://example.org/book/book1>
<http://purl.org/dc/elements/1.1/title> "SPARQL Tutorial"

SELECT ?title
WHERE
{<http://example.org/book/book1>
<http://purl.org/dc/elements/1.1/title> ?title .}
```

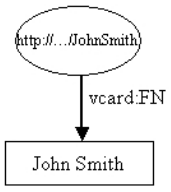
2.4. Jena2

Jena 는 자바로 만들어진 오픈 소스 시맨틱 웹 프레임워크이다. HP 연구소에서 개발된 온톨로지 관리 시스템으로서, 시맨틱 웹을 개발하는 사람이라면 누구에게나 공개적으로 제공되며 시맨틱 웹의 응용을 제작하는 사람들의 표준 개발 툴이라 할 수 있다. 현재는 Jena1 에 이어 Jena2 가 발표되었다. Jena 는 RDF(S), OWL 문서의 파서(Parser), RDF(S), OWL 의 in-memory 객체인 Model, OntModel 과 데이터베이스를 이용한 Persistent 객체(예: Postgresql, MySQL)와 개발자를 위한 여러 기능적인 API 를 제공한다. 질의처리 언어로 RDQL 과 SPARQL 을 제공한다.[4]

다음은 Jena2 을 이용하여 간략한 RDF 그래프를 구현하는 예제이다.[5]

[표 5] Jena2 를 이용한 RDF 그래프 구현

```
// some definitions
static String personURI    =
"http://somewhere/JohnSmith";
static String fullName     = "John Smith";
// create an empty Model
Model model =
ModelFactory.createDefaultModel();
// create the resource
Resource johnSmith = model.createResource(personURI);
// add the property
 johnSmith.addProperty(VCARD.FN, fullName);
```



3. RDF와 SPARQL을 이용한 메타데이터 관리

본 논문에서는 PC 내 존재하는 다양한 영상에 대한 메타데이터를 RDF 를 이용하여 저장하고, SPARQL 을 이용하여 검색하는 방안을 제시하고, 솔루션을 구현한다. 제시한 솔루션을 IMMS (Image Metadata Management Solution)로 지칭하겠다.

3.1. IMMS 구현

3.1.1. 메타데이터 모델

본 논문에서는 영상의 메타데이터를 다음 표와 같이 간략히 정의하였다.

[표 6] 메타데이터 모델

Property	Value Type	입력 구분	설명
imms:TITLE	Text	사람	제목
imms:EVENT	Text		촬영하게된 이벤트
imms:LOCATION	Text		촬영 장소
imms:CHARACTER	Bag Text		촬영 인물
imms:DESCRIPTION	Text		설명
imms:CREATE_DATE	Date	시스템 (EXIF)	생성일
imms:MAKE	Text		카메라 제조업체
imms:MODEL	Text		카메라 모델
imms:WIDTH	Integer		영상의 넓이
imms:HEIGHT	Integer		영상의 높이
imms:XRESOLUTION	Integer		수평해상도
imms:YRESOLUTION	Integer		수직행상도

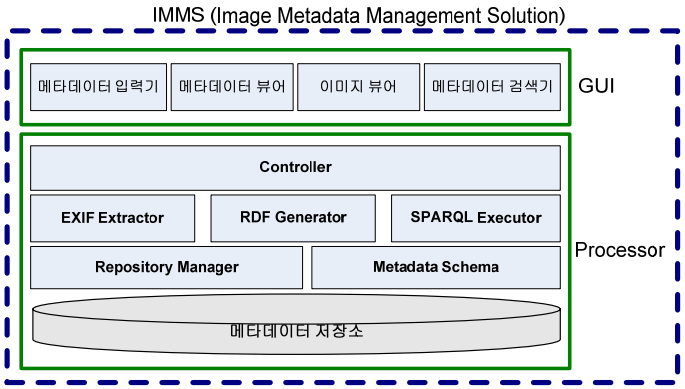
※ imms: http://grad.knou.ac.kr/IMMS/imms-rdf/1.0#

메타데이터의 항목은 사용자가 직접 입력한 상황(컨텍스트) 메타데이터와 영상으로부터 자동으로 입력되는 메타데이터(EXIF)으로 구성된다. [표 8]에 나열된 항목들은 예시이며, 실제 어플리케이션에서는 목적에 맞게 메타데이터 항목들을 결정할 수 있다.

3.1.2. 기능 모듈

IMMS 는 Controller, EXIF Extractor, Metadata Schema, RDF Generator, Repository Manager, SPARQL Executor, GUI 의 모듈로 구성된다.

아래 그림은 IMMS 의 기능 모듈 다이어그램이다.



[그림 2] 기능 블록 다이어그램

3.1.1.1. EXIF Extractor

EXIF Extractor 는 영상 생성시 카메라가 입력한 메타데이터를 추출해는 기능을 수행한다. 추출 가능한 다양한 EXIF 정보 중, 메타데이터 모델에서 정의된 메타데이터만 RDF 로 저장이 된다.

3.1.1.2. Metadata Schema

Metadata Schema 는 IMMS 에서 사용하기로 정의한 논리적인 메타데이터 모델을 RDF Generator 에서 참조할 수 있도록 물리적으로 구현한 메타데이터 스키마이다.

3.1.1.3. RDF Generator

RDF Generator 는 사용자가 입력한 Context 메타데이터와 영상으로부터 추출한 EXIF 메타데이터를 메타데이터 모델에 맞게 RDF 로 표현하고, 파일과 메모리에 저장하는 역할을 수행한다.

3.1.1.3. Repository Manager

Repository Manager 는 RDF 로 저장된 메타데이터 파일 저장소를 관리한다. SPARQL Executor 가 동작할 수 있도록 디스크에 저장된 RDF 파일들과 메모리를 동기화하는 역할을 수행한다.

3.1.1.4. SPARQL Executor

SPARQL Executor 는 RDF 로 저장된 메타데이터 모음으로부터 SPARQL 을 이용하여 사용자가 요청하는 검색을 수행한다. 다음은 이를 구현한 부분이다.

[표 8] SPARQL Executor 의 검색 구현 부분

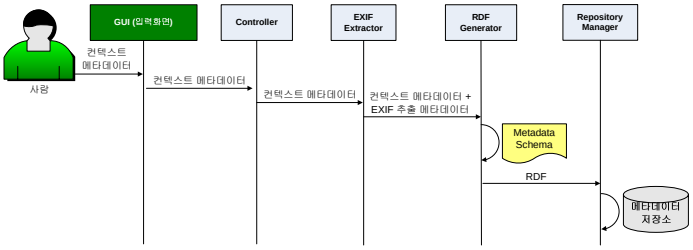
```
public static void doSearch(ImageData meta, Hashtable resultTable)
throws Exception {
  //PREFIX 설정
  String prolog = "PREFIX imms: <" + ImageMetaSchema.getURI() +
  ">" ;
  //쿼리 생성
  StringBuffer queryString = new StringBuffer();
  Iterator keys = meta.table.keySet().iterator();
  while (keys.hasNext()) {
    String tag = (String)keys.next();
    String value = (String)meta.table.get(tag);
    if (value != null && !value.equals("")) {
      PropsContainer pc =
      (PropsContainer)ImageMetaSchema.properties.get(tag);
      if (pc.getType().equals("literal")) {
        queryString.append(" ?resource imms:" + tag + " ?" + tag +
        " FILTER regex(?" + tag + ", '" + value + "') .");
      }
      else if (pc.getType().equals("bag")) {
        String bagName = "A" + System.currentTimeMillis();
        queryString.append(" ?resource imms:" + tag + " ?" +
        bagName + " . ?" + bagName + " ?p ?" + tag + " FILTER
        regex(?" + tag + ", '" + value + "') .");
      }
    }
  }
  queryString.append(" ?resource imms:FILE_NAME ?file ");
  //쿼리 수행
  Query query = QueryFactory.create(prolog + NL +
  "SELECT ?resource ?file WHERE {" + queryString.toString() + "}") ;
  QueryExecution qexec = QueryExecutionFactory.create(query,
  RepositoryManager.model) ;
  try {
```

```
    ResultSet rs = qexec.execSelect() ;
    while (rs.hasNext()) {
      QuerySolution rb = rs.nextSolution() ;
      RDFNode resource = rb.get("resource");
      RDFNode file = rb.get("file");
      resultTable.put(resource.toString(), file.toString());
    }
  }
  finally {
    qexec.close() ;
  }
}
```

3.3. IMMS의 동작

3.3.1. 메타데이터 등록

다음은 메타데이터 등록의 시나리오이다.



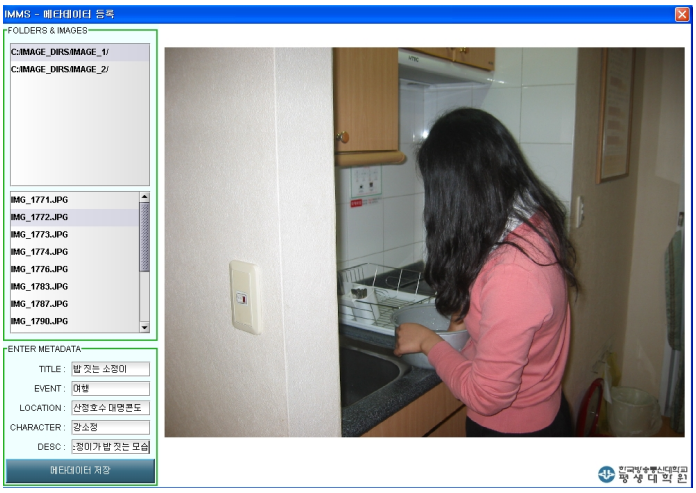
[그림 3] 메타데이터 등록 시나리오

사용자가 GUI 기반 입력화면을 통해 영상을 선택하고, Context 메타데이터를 입력하면, 여기에 EXIF 정보로부터 자동으로 추출된 메타데이터가 합쳐져서 메타데이터 모델에 맞게 RDF 로 서술되고 저장된다. 다음은 IMMS 에 의해 RDF 로 저장된 메타데이터의 예제이다.

[표 9] RDF 로 서술된 영상 메타데이터

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:imms="http://grad.knou.ac.kr/IMMS/imms-rdf/1.0#" >
  <rdf:Description rdf:nodeID="A0">
    <rdf:_1>강소정</rdf:_1>
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-
syntax-ns#Bag"/>
  </rdf:Description>
  <rdf:Description
rdf:about="C:/IMAGE_DIRS/IMAGE_1/IMG_1771.JPG">
    <imms:YRESOLUTION>180</imms:YRESOLUTION>
    <imms:EVENT>여행</imms:EVENT>
    <imms:CREATE_DATE>2004:04:16
21:01:54</imms:CREATE_DATE>
    <imms:MODEL>Canon PowerShot S30</imms:MODEL>
    <imms:CHARACTER rdf:nodeID="A0"/>
    <imms:WIDTH>1600</imms:WIDTH>
    <imms:DESCRIPTION>소정이가 밥 짓는
모습</imms:DESCRIPTION>
    <imms:XRESOLUTION>180</imms:XRESOLUTION>
    <imms:MAKE>Canon</imms:MAKE>
    <imms:HEIGHT>1200</imms:HEIGHT>
    <imms:FILE_NAME>1173023845680.rdf</imms:FILE_NAME>
    <imms:LOCATION>대명콘도</imms:LOCATION>
    <imms:TITLE>밥 짓는 소정이</imms:TITLE>
  </rdf:Description>
</rdf:RDF>
```

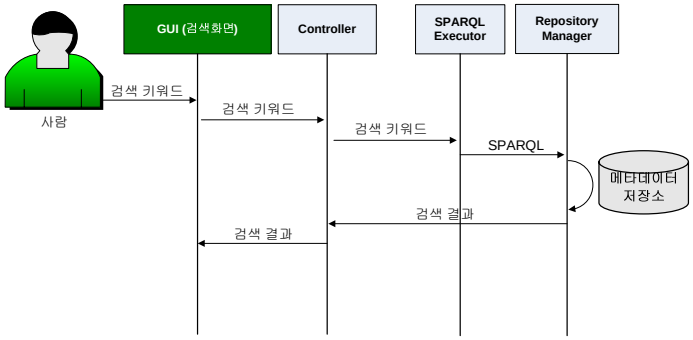
다음은 사용자의 입력을 받아 메타데이터 등록을 수행하는 IMMS 의 실제 화면이다.



[그림 4] IMMS 를 통한 메타데이터 등록

3.3.2. 메타데이터 검색

다음은 메타데이터 검색의 시나리오이다.



[그림 5] 메타데이터 검색 시나리오

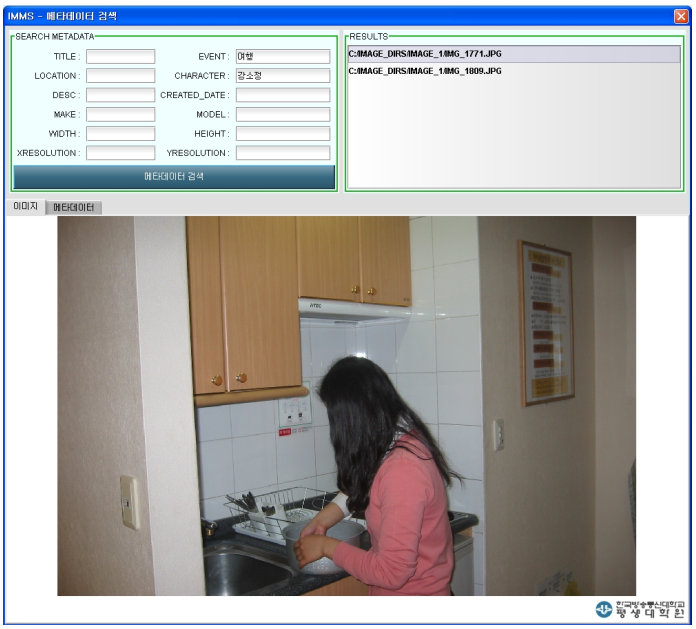
사용자가 GUI 기반 검색화면을 통해 검색 키워드를 입력하면, SPARQL Executor 는 해당 검색 요청을 SPARQL 로 변환하여 RDF 저장소에 대한 검색을 수행한 후, 결과를 다시 사용자에게 돌려보낸다. 다음은 촬영인물 중 '강소정'이 있고, 촬영 이벤트가 '여행'인 영상을 검색하기 위해 IMMS 에 의해 생성된 SPARQL 이다.

[표 10] 검색을 위한 SPARQL

```
PREFIX imms: <http://grad.knou.ac.kr/IMMS/imms-rdf/1.0#>

SELECT ?resource ?file
WHERE { ?resource
imms:CHARACTER ?A1175158439342 . ?A1175158439342 ?p ?CHA
RACTER FILTER regex(?CHARACTER, '강소정') .
?resource imms:EVENT ?EVENT FILTER regex(?EVENT, '여행') .
?resource imms:FILE_NAME ?file }
```

다음은 사용자의 입력을 받아 메타데이터 검색을 수행하는 IMMS 의 실제화면이다.



[그림 6] IMMS 를 통한 메타데이터 검색

4. 결론

4.1. 연구의 의의

W3C 의 표준인 RDF 를 이용하여 메타데이터를 표현하고 저장하여 활용하려는 연구는 많이 수행되어 왔다. "RDF 기반의 학습 메타데이터 관리"에서는 RDF 를 이용하여 차세대 학습 메타데이터를 관리하는 방안을 제시하였으며[6], "공간관계 표현 기반 RDF 메타데이터를 이용한 의미적 영상 검색"에서는 기존의 text-based 의 영상 메타데이터 관리를 개선하기 위하여 공간관계 어휘들이 의미적으로 표현된 RDF 메타데이터를 제안하였다.[7]

본 논문은 이를 기반으로, 영상을 표현하기 위한 메타데이터 모델, 메타데이터의 RDF 로의 표현 및 저장, 저장된 메타데이터의 SPARQL 을 통한 검색으로 구성되는 체계화된 영상의 메타데이터 관리 방안을 제시하고, 이를 Jena2 기반으로 구현하여 RDF 와 SPARQL 이 상용 영상 관리 툴에 유용하게 사용되어질 수 있음을 제시하였다. 현재 여러 업체에서 제공하고 있는 영상 관리 툴은 주로 영상 프로세싱에 초점이 맞추어져 있으며, 메타데이터 관리가, 단순한 키워드 매칭이 고작인, 체계적이지 못한 것이 사실이다. 본 논문에서와 같이 영상 관리 어플리케이션에 RDF 와 SPARQL 이 도입되어 새로운 서비스가 가능해 질 것을 기대해본다.

4.2. 향후 연구

본 논문에서 수행한 연구는 향후 연구를 통해 다음과 같이 확장될 수 있다.

- RDF 로 표현된 메타데이터와 온톨로지와의 연계를 통한 논리적 추론
- 솔루션에 통신 기능을 추가하여, 연결된 다른 PC 의 영상 검색 / 활용
- MPEG-7 과의 연계

이러한 연구들은 웹 2.0 또는 3.0 시대에 새로운 서비스의 발굴을 가능하게 하는 기술적 기반이 될 것이다. 이와 더불어 저작권, 프라이버시 등과 같은 문제에 대한 제도와 인식, 기술도 함께 발전되고 연구되어야 한다.

<참고문헌>

[1] Frank Manola 외 1 인, "RDF Primer", http://www.w3.org/TR/rdf-primer/, 2004.02

[2] Dave Beckett, "RDF/XML Syntax Specification", http://www.w3.org/TR/rdf-syntax-grammar/, 2004.02

[3] Eric Prud'hommeaux 외 1 인, "SPARQL Query Language for RDF", http://www.w3.org/TR/rdf-sparql-query/, 2007.03

[4] 정호영외 2 인, "관계형 데이타베이스 기반의 RDF 와 OWL 의 저장 및 질의처리", 한국정보과학회 논문지 C - 컴퓨팅의 실제 VOL.11, 2005

[5] Brian McBride, "An Introduction to RDF and the Jena RDF API", http://jena.sourceforge.net/tutorial/RDF_API/index.html, 2006.12

[6] 이영석외 5 인, "RDF 기반의 학습 메타데이터 관리", 한국정보처리학회 논문지 A VOL.13-A NO.01, 2006.02

[7] 황명권외 2 인, "공간관계 표현 기반 RDF 메타데이터를 이용한 의미적 영상 검색", 한국정보처리학회 논문지 B VOL.11-B NO.05, 2004