**International Journal of Informatics and Communication Technology (IJ-ICT)**
Vol.2, No.1, January 2013, pp. 1~8
ISSN: 2252-8776

# XML-Based RDF Data Management for XPath Query Language

**Win Lai Hnin**
University of Computer Studies, Yangon, Myanmar

**Article Info**

***Article history:***

Received Sept 29th, 2012
Revised Nov 01th, 2012
Accepted Nov 20th, 2012

***Keyword:***

Semantic Web
RDF
XML Schema
XPath

**ABSTRACT**

The Semantic Web is an extention of the current Web that will allow to find, share and combine information more easily. The meaning in the Semantic Web is mostly represented by Resource Description Framework (RDF). To harvest such power requires robust and scalable data repositories that can store RDF data. Most of the existing RDF storage techniques rely on relation model and relational database technologies for these tasks. The mis-match between the graph model of the RDF data and the rigid 2D tables of relational model jeopardizes the scalability of such repositories and frequently renders a repository inefficient for some types of data and queries. This paper proposed a system that can store RDF data in the XML repository and discuss the basic idea of conversion from RDF Schema to XML Schema and XPath queries for extract information from XML database.

***Corresponding Author:***

Win Lai Hnin
University of Computer Studies, Yangon, Myanmar
Email: winlaihnin.84@gmail.com

## 1. INTRODUCTION

A Semantic Web document not only offers human understandable content but also formal semantic describing the content of the document in a machine-processable way. Computers essentially play a role in parsing web pages for displaying and processing jobs. They have no reliable way to draw the semantics from a page. The Semantic Web will improve the meaningful content of the web pages. It is not completely a new generation of web, but an extension to the current Web which concentrates on the need of machine understandable metadata for the web documents. There is a gap within the Web of data, on one side; XML provides a popular format for data exchange with a rapidly increasing amount of semi-structured data available. On the other side, the Semantic Web builds on data represented in RDF, which is optimized for data interlinking and merging, the amount of RDF data published on the Web is also increasing, but not yet at the same pace. It would clear be useful to enable reuse of RDF data in the XML world and vice versa.

XML queries are significantly different from the conventional RDBMS queries in that the former routinely involve a tree-shaped pattern that is to be matched against the database. As in the context of RDB and XML, the selection of storage models is critical to a data as it is the domination factor to determine how to evaluate queries and how the system behaves when scales up. With currently available tools and language, translating between an existing RDF format and XML is a tedious and error-prone task.

The rest of the paper is organized as follows: Section 2 discusses related research in RDF data repositories. Section 3 introduces about RDF, UML, ODL and XML. Section 4 describes mapping method between RDF and XML. Section 5 explains XPath query language and Section 6 define a detailed description of the proposed system. Finally, the paper is concluded with section 7.

## 2. RELATED WORK

Most of the existing RDF data repositories [3, 6] rely on relational models for data storage and evaluate SPARQL queries by rewriting them into SQL queries and then executing them in the RDB engine. Among them there are two major directions: (1) keeping the simple triple data model of RDF data, e.g. *triple store* [8]; and (2) decomposing RDF triples into relations, either based on predicates, e.g. *vertical partition* or based on semantics, e.g. *property table* [6]. The *triple store* does not scale well as the evaluation of a complex SPARQL query invokes many self-joins. Various indexing techniques [2] were proposed as remedies, at the cost of huge increase in storage space and decrease in the scalability and update efficiency. The *vertical partition* [3] works well for SPARQL queries when all predicates in the WHERE clause are known. Otherwise, all tables have to be accessed and results unioned. For example, the RDF data are stored in five tables. All need to be accessed to evaluate the SPARQL query above. The *property table* incurs small number of joins for some queries because a selection in one property table can match multiple simple access patterns. However it suffers storage redundancy and large overhead in query evaluation [3].

An alternative approach [11] preserves the graph nature of RDF data by storing RDF graphs in an object-relational database. However, this separates the RDF schema and RDF primary data, which brings difficulties in evaluating queries containing both schema and data instances.The proposal of serializing RDF graph into XML trees to utilize existing XML technologies [9] focused on representing all RDF features such as blank node in XML, but pays less or no attention to the efficiency of RDF data storage and query evaluation. It either leads to XML data [11] with large redundancy or flat XML data that cannot take full advantage of XML query evaluation techniques. Mo Zhou and Yuqing Wu [8] proposed the two RDF-to-XML decomposition algorithms for the decomposition in two steps: (1) the schema-level decomposition which maps an RDF schema to a set of XML schemas and (2) the data-level decomposition which maps RDF data to a set of XML documents conforming to the XML schemas which brings inefficient in mapping RDF data to a set of XML documents conforming to the XML schemas in some applications.

## 3. RDF, UML, ODL and XML

1. **RDF -** The meaning of the Semantic Web is to allow everybody to publish interlinked machine-processable information with the ease of publishing a web page. RDF (Resource Description Framework) encrypts these meanings in the sets of triples that build meaningful webs about related things. RDF is a W3C recommended language for describing linked data of the Semantic Web. RDFS (RDF Schema) extends RDF vocabularies to define the structure of underlying RDF data. RDF can also be used to show the resources on the web which can’t be directly retrieved from the current web. Actually it is the technology which provides the descriptive knowledge in formal ways about web document’s domain, its contents and relationship between those contents. Actually it formally describes about contents of web document. It organizes formal description in terms of classes, properties and their instances.
2. **UML -** The UML is a standard language for writing software blueprints. The UML may be used to visualize, specify, construct, and document are artifacts of a software-intensive system. The UML is appropriate for modeling systems ranging from enterprise information systems to distributed Web-based applications. The goal is for UML to become a common language for creating models of object oriented computer software. UML defines a number of different kinds of diagrams. The kind of diagrams that this system uses is Class Diagram. A class diagram is a diagram that shows classes, interfaces and their relationships.
3. **ODL -** The ODL is a standardised way of describing objects in an object-oriented database environment which was defined by the ODMG. ODL specifies the logical schema for an object-oriented database. It is independent of any particular programming language. ODL is an extension of IDL, which is a component of CORBA. ODL is a kind of data definition language at conceptual level.
4. **XML -** As the W3C standard document format for writing and exchanging information on the Web, XML (eXtensible Markup Language) is mostly concerned about syntax. XML is textual language quickly gainging popularity for data representation and exchange on the Web. XML Schema is a standard for describing the structure of an XML document. Documents that conform to an XML Schema are called schema-valid. Nested, tagged elements are the building blocks of XML. Each tagged element has a sequence of zero or more attribute/value pairs, and a sequence of zero or more subelements. These subelements may themselves be tagged elements or they may be “tagless” segments of text data. XPath is a declarative query language for XML that provides simple syntax for addressing parts of an XML document.

Table 1. Mapping from RDF schema to UML class diagram

| No | RDF Schema | UML Calss Diagram |
|---|---|---|
| 1 | Class | Class |
| 2 | Type | InstanceOf |
| 3 | type | Type of Model Element |
| 4 | Property | Attribute |
| 5 | Property | Association |
| 6 | subClassof | Generalization |
| 7 | subPropertyof | Stereotyped dependency between 2 associations * |
| 8 | subPropertyof | Generalization between stereotyped classes* (named DAML property) |
| 9 | Comment | Note |
| 10 | Label | Name |
| 11 | seeAlso/isDefinedBy | Tagged value on a class and association* |
| 12 | Literal | Attribute type:string |
| 13 | value | Attribute value:value |
| 14 | domain | Class containing the attribute |
| 15 | domain | Source class of an association |
| 16 | range | Attribute type(primitive or class) |
| 17 | range | Target class of an association |

## 4. Mapping from RDF Schema to XML Schema

### 4.1 Mapping from RDF Schema to UML class diagram

RDF's vocabulary description language, RDF Schema, is a semantic extension of RDF. It provides mechanisms for describing groups of related resources and the relationships between these resources. RDF Schema vocabulary descriptions are written in RDF using the terms described in this document. These resources are used to determine characteristics of other resources, such as the domain and ranges of properties.The mapping from RDF Schema to UML class diagram are as follow:

**Example:**

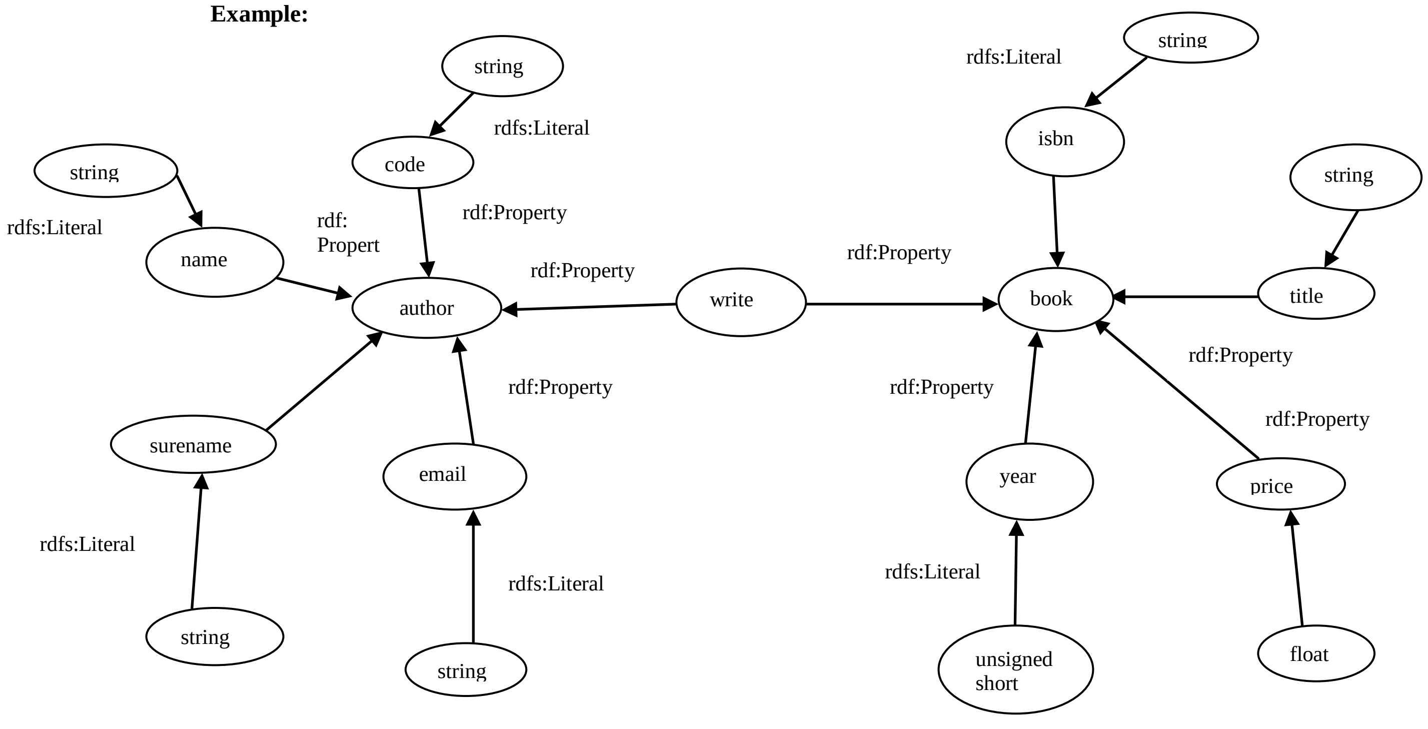

Figure 1. RDF Schema

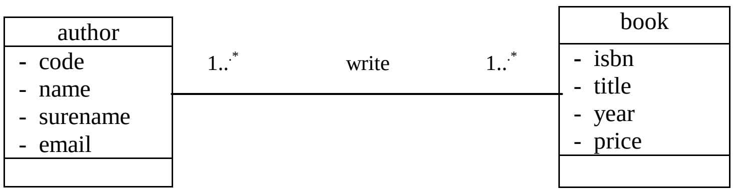

Figure 2. UML Class Diagram for RDF Schema

### 4.2 Mapping from UML class diagram to XML Schema using ODL
#### 4.2.1 Mapping from UML class diagram to ODL

The mapping from UML to ODL is followed:

1. A class declaration includes:
   - A name for the class
   - Optional key declaration(s)
   - Extent declaration = name for the set of currently existing objects of the class
   - An element declaration is an attribute, a relationship or a method.

2. Attributes are (usually) elements with a type that does not involve classes. *attribute <type> <name>;*

3. Relationships connect an object to one or more other objects of one class. relationship <type > <name>minverse <relationship>. The type of a relationship is either
   - Set–unordered collection without duplicates
   - Bag–unordered collection that may contain duplicates
   - List–ordered collection, all the same type
   - Array–dynamically sized ordered collection, locatable by position
   - Dictionary– unordered sequence of key-value pairs without duplicates

4. The extends keyword is used to represent Generatio.

5. ODL subclasses describe superclass with a colon and its name and subclass lists only the properties unique to it and

Also inherits its superclass' properties.

**Example:**

```
Class author
(extend authors key code)
{
     attribute string code;
     attribute string name;
     attribute string surname;
     attribute list <string>
email;
     relationship set <book>
write
                    Inverse
book::writtenby;
};
```

```
Class book
(extend books key isbn)
{
      attribute string isbn;
      attribute string title;
      attribute unsigned short
year;
      attribute float price;
      relationship set <author>
writtenby
                   Inverse
author::write;
};
```

ODL representation for the UML class diagram of Figure 2

### 4.2.2 Mapping from ODL to XML Schema

This mapping describes a set of seven transformation rules to convert a simple ODL database schema into XML-SCHEMA schema:

1. **Rule 1**: Since every XML document must have a root element, such an element has to be created with, for example, the database name. One should also associate with that root element an anonymous complex type that includes a special choice element with a *maxOccurs* attribute equal to *unbounded*.
2. **Rule 2**: Each top-level class in the ODL schema is converted into an element with the same name, which is included in the special *choice* element created above (rule 1). One should also associate with each of these elements a complex type with a special *sequence* element. The contents of each one of these special *sequence* elements are derived from the next rules.
3. **Rule 3**: The attributes with basic data types (*string*, *short*, *date*, *float*, etc.) are translated into atomic elements with the same name. These elements are included in the corresponding special *sequence* element that has resulted from rule 2, and their data types should, if possible, be the same, although users could indicate another data type.
4. **Rule 4**: Regarding *key* attributes, a special *attribute* element has to be declared as a sub-element of the element that has resulted from rule 1. This special element should also have a *name* attribute which value should be equal to the name of the class that the key attribute belongs to, plus the letter K (Key) appended to it.
5. **Rule 5**: Each relationship that contains the keywords *set*, *list* or *bag*, is converted into an element with the same name. This element is included in the special *sequence* element that has resulted from rule 2 and, since the cardinality of the relationship is greater than one, one should explicitly associate the attribute *maxOccurs* with it, with value *unbounded*. This element should include an anonymous complex data type with a special element *complexContent*. Moreover, this special element should include a special *restriction* element that contains an attribute with the name of the key attribute of the referred class. Additionally, a special *keyref* element has also to be declared as a sub-element of the element that has resulted from rule 1.
6. **Rule 6**: Each relationship that does not contain any of the keywords *set*, *list* or *bag*, is converted into an element with the same name. The translation is similar to that of the previous rule, with three exceptions: the attribute *maxOccurs* (with value *unbounded*) should not be associated with the created element.
7. **Rule 7**: Each attribute of a class that contains the keyword *list* in its definition is converted into an element with the same name, which is included in the special *sequence* element that has resulted from rule 2. This element should include an anonymous simple data type. Moreover, the simple data type should include a special *list* element with an *itemType* attribute, whose value could be indicated by users, or defined by default (for instance, string).

**Example:**

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"
attributeFormDefault="unqualified">
<xsd:annotation>
<xsd:documentation>XML Schema derived from an ODL
schema</xsd:documentation>
</xsd:annotation>
<xsd:element name="bibdb">
     <xsd:complexType>
      <xsd:choice maxOccurs="unbounded">
       <xsd:element name="author">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="code" type="xsd:string"/>
        <xsd:element name="name" type="xsd:string"/>
        <xsd:element name="surname" type="xsd:string"/>
        <xsd:element name="email">
          <xsd:simpleType>
             <xsd:list itemType="xsd:string"/>
          </xsd:simpleType>
       </xsd:element>
```

```
          <xsd:element name="write" maxOccurs="unbounded">
             <xsd:complexType>
              <xsd:complexContent>
                <xsd:restriction base="xsd:anyType">
                  <xsd:attribute name="isbn" type="xsd:string"/>
                < /xsd:restriction>
              </xsd:complexContent>
             </xsd:complexType>
           </xsd:element>
         </xsd:sequence>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="book">
         <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="isbn" type="xsd:string"/>
              <xsd:element name="title" type="xsd:string"/>
              <xsd:element name="year" type="xsd:unsignedShort"/>
              <xsd:element name="price" type="xsd:float"/>
              <xsd:element name="writtenby" maxOccurs="unbounded">
                 <xsd:complexType>
                   <xsd:complexContent>
                     <xsd:restriction base="xsd:anyType">
                       <xsd:attribute name="code" type="xsd:string"/>
                    </xsd:restriction>
                  </xsd:complexContent>
                </xsd:complexType>
             </xsd:element>
          </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
```

XML Schema for ODL representation

```
<?xml version="1.0" encoding="UTF-8"?>
<bibdb xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"
xsi:noNamespaceSchemaLocation="C:\ \Conf\SCCC
2002\Camara Ready\schema.xsd">
<author>
   <code>A1</code>
   <name>Michael</name>
   <surname>Stonebraker</surname>
   <title>PhD</title>
   <email>ms@hotmail.com</email>
  <write isbn="1-55860-452-9"/>
</author>
<book>
   <isbn>1-55860-452-9</isbn>
   <title>O-R DBMSs Tracking … Great Wave</title>
   <year>1999</year>
   <price>50</price>
  <writtenby code="A1"/>
</book>
</bibdb>
```

XML document

## 5. XPath QUERY LANGUAGE

XPath is designed for XML documents. It provides a single syntax that can use for queries, addressing and patterns. Fundamentally, an XPath is an expressing. Evaluating an XPath expression results in one of the following (1) A node set, (2) A Boolean, (3) A floating-point number, (4) A String of Unicode character. Specifically, identify constraints require the resultant node set to contain only elements or attributes. Fragment identifiers restrict the resultant node set to contain only elements.

## 6. PROPOSED FRAMEWORK

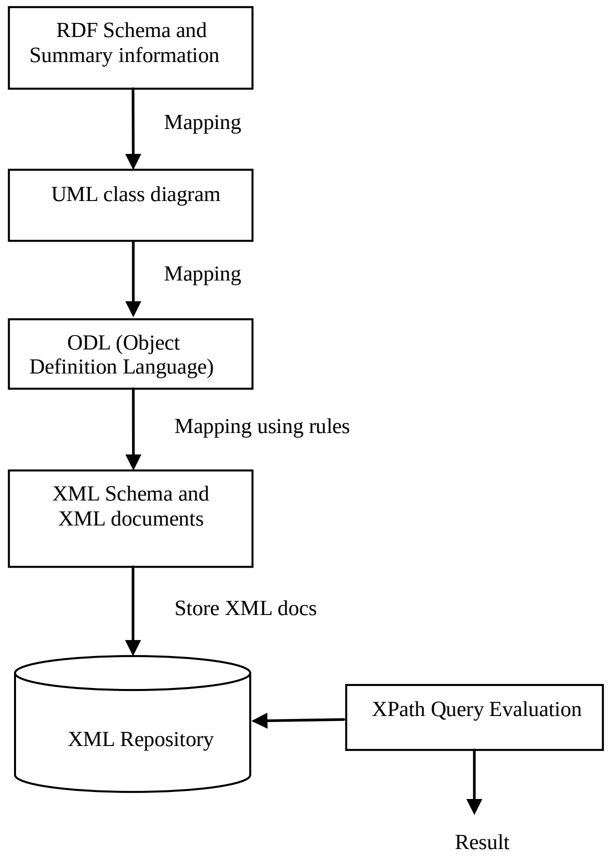

Figure 3 .Overview of the proposed framework

The detailed analysis of this system is given in section 4. First, observe the similarity between RDF and XML, in terms of data representation (e.g. using links to represent relationships among data instances) and query (e.g. tree pattern matching in XML and graph pattern matching in RDF), and propose to leverage the sophisticated storage management and query evaluation techniques of XML data repositories. RDF data are significantly different from XML data in syntax and data model: RDF data and schema are directed graphs with both nodes and edges labeled, while XML data are trees with only nodes labeled. The proposed system is designed to map from RDF schema to UML class diagram and from UML class diagram to ODL (Object Definition Language) and from ODL to XML schema and XML documents that stores in an XML repository and XPath queries to be evaluated against the XML documents using the latest XML query evaluation techniques. Although our work, as other RDF storage approaches, is syntax independent, the different between the data models brings substantial challenges to storing and querying RDF data using XML techniques, in transforming graphs into trees, and keeping storage efficiency. The XML-based RDF data management and query evalution approach which is scalable and capable of supporting high query performance for XPath queries.

## 7. CONCLUSION

To answer the increasing demands on RDF repository, carefully studied the existing RDF data management systems, identified the preferred properties of an RDF repository and proposed to take advantage of the latest XML data storage. This system identified the mapping from RDF data to UML class diagram and UML class diagram to XML document. In addition, this approach is efficient for time consuming in translation from RDF to XML documents for supporting Semantic Web applications in various domains.

## REFERENCES

[1] Artur Afonso de Sousa1, José Luís Pereira2 and João Álvaro Carvalho, "Mapping Rules to Convert from ODL to XML-SCHEMA." ,2002.

[2] C. Weiss, *et al.* "Hexastore: sextuple indexing for semantic web data management", *PVLDB,* 1(1):1008–1019, 2008.

[3] D. J. Abadi, *et al.* "Scalable Semantic Web Data Management Using Vertical Partitioning", In *VLDB*, 2007.

[4] F. Tian, D. DeWitt, J. Chen, and C. Zhang. "The Design and Performance Evaluation of Alternative XML Storage Strategies. Technical report, Univ. of Wisconsin, 2000.

[5] G. Klyne and J. Carroll. " Resource Description Framework (RDF): Concepts and Abstract Syntax." http://www.w3.org/TR/rdf-concepts/, 2003.

[6] J. J. Carroll, *et al.* "Jena: implementing the semantic web recommendations", 2004.

[7] J. Clark and S. DeRose. XML Path Language (XPath) Version 1.0. http://www.w3.org/TR/1999/REC-xpath-19991116, 1999.

[8] M. Zhou and W. Yuqing. "XML-Based RDF Data Management for Efficient Query Processing", 2010.

[9] Norman Walsh. Rdf twig: accessing rdf graphs in xslt. *In Proc. Extreme Markup Languages, 2003*.

[10] R. Bourret. XML and Databases. http://www.rpbourret.com/xml/XMLAndDatabases.htm, 1999-2005.

[11] S. Alexaki, *et al*. The ICS-FORTH RDFSuite: Managing Voluminous RDF Description Bases. In *SemWeb*, 2001.

[12] S. Bischof, S. Decker, K. Thomas, N. Lopes, A. Polleres. "Mapping Between RDF and XML with XSPARQL", April 2011.

[13] S. Battle."Gloze: XML to RDF and back again", http://www.hpl.hp.com/personal/steve-battle.

[14] T. Bray, J. Paoli, C. Sperberg-McQueen, and E. Maler. Extensible Markup Language (XML) 1.0 (Second Edition). http://www.w3.org/TR/2000/REC-xml-20001006, 2000.

## BIOGRAPHY OF AUTHOR

I received B.C.Sc, B.C.Sc(Hons:) and M.C.Sc degrees in Computer Science from University of Computer Studies, Yangon(UCSY) in 2005,2006 and 2009 respectively. During 2007-2009, I served as a tutor in Computer Application Department of University of Computer Studies, Mawlamyine. From 2009 to 2010, I attened PH.D coursework in UCSY. Now, I'm doing my research which is related with Database Management System.