Study and Constructing RDF model for a well formatted Valid XML document

1.B.Hemanth Kumar,
Associate., Prof., Dept., of Information Technology,
RVR & JC College of Engineering,
Guntur, A.P. India

2. Prof. M. Surendra Prasad Babu,
Chairman BOS, Dept., of CS & SE
AU college of Engineering,
Andhra University,
Visakhapatnam, A.P, India.

Abstract: In recent years semantic web is one of the technologies to make the WWW as machine-understandable. The Resource Description Framework (RDF) proposed by W3C is used for describing metadata about (Web) resources. The RDF data model can be represented as entity-relationship or class diagrams, as it is based upon the idea of making statements about resources (in particular Web resources) in the form of subject-predicate-object expressions. These expressions are known as *triples* in RDF terminology. The Resource Description Framework (RDF) is a language for representing web information in a minimally constraining, extensible, but meaningful way. XML is the basis to represent RDF models on semantic Web. In business applications data can be represented in Xml which describes the structure of data. But to increase the efficiency with respect to data searching and retrieval the xml data can be represented in RDF format. All the elements and attributes of an xml can be defined in RDF format. In this paper we are representing how to define RDF elements for a well formatted and valid XML document.

Keywords-Semanticweb, XML,DTD,,RDF

I. INTROCDUCTION

The Semantic web is a concept of having data on the web, defined and linked in a way that it can be used by machines not just for display purposes, but for automation, integration, and reuse of data across various applications. As per the W3C recommendations semantic web will be build on RDF. Ontology is the central component of the Semantic web used to define the data models in a specific domain. Resource Description Framework (RDF)[1] is a language to describe these data models that are available on semantic web[2]. RDF is the underlying unified data model for representing semantics. The data model and XML serialization syntax is used for describing resources both on and off the Web. RDF makes use of unique identifiers (URI, Uniform Resource Identifier) for describing metadata. URIs are used to describe things, also called resources, which could represent people, places, documents, images, databases, etc. All RDF applications adopt a common convention for identifying these things. A subset of URI, the Uniform Resource Locator or URL, is concerned with the location and retrieval of resources, while URI is a unique identifier for things or resources that we describe but that may not necessarily be retrievable. However, RDF provides a consistent, standardized way to describe and query internet resources, from text pages and graphics to audio files and video clips. RDF[3] is based on three core concepts: resources, properties, and statements. A resource is the subject or the object of an RDF description. A property describes a resource; its value is either a literal or a reference to another resource. A statement combines the property of a resource with a value. It can be represented as a subject, a predicate, and an object, where subject and object may be URIs and predicate is the property linking the object to the subject. The subject denotes the resource, and the predicate denotes traits or aspects of the resource and expresses a relationship between the subject and the object.

Figure 1 :General form of an RDF statement

In this section we are giving a brief introduction about semantic web technologies, section 2 explains how you can represent data in xml format, how these xml elements can be defined in RDF format is given in section 3 fallowed by some examples with graphical representation and then conclusions.

II. XML REPRESENTATION OF DATA

XML was proposed by W3C to describe and structure data for Web applications using a simple, flexible, and standards-based format. An XML[4] document is a set of data entities where each entity is made up of a number **of** data ***elements.*** Data elements can be nested hierarchically and can have ***attributes*** to indicate how the contents **of** the element should be interpreted. Similar to HTML documents, through programming interfaces such as DOM, the elements of an XML document and their attributes can be constructed and manipulated by scripts as structured variables. In particular, the structured XML document can be passed back and forth as messages between computer programs for interchanging information. This enables data to be migrated, exchanged, and shared across the Web. Thus, XML is anticipated to lead the evolution of the Web technology. The data interactions introduced by XML documents can be more complicated than those introduced by HTML documents. With XML technology, programmers can integrate elements into a single XML document from multiple data sources**.** Thus, all the integrated elements must be identified and analyzed in order to understand the data interactions of an XML document. Moreover, XML document elements can be created dynamically. Development of the World Wide Web has reached a revolutionary new phase - the Second-Generation Web, **also** known as the Semantic or XML-based Web[5]. This next-generation Web will be founded on XML (Extensible Markup Language) rather than on HTML. XML is a metalanguage that permits a set of users to create its own mark-up language for describing the contents of Web documents. The use of XML has many advantages, since an XML file contains not only data (like HTML) but also metadata - structural and semantic information about that data. In this sense, an XML document is very similar to a database, and the Second-Generation Web in general will look like one big database. Since XML is a platform and application independent language, the visual and logical presentation of XML data to the end user will generally be defined not by a browser, but by an application distributed together with specific XML documents. This means that the same information can be presented in many different forms, including visually rich ones.

A. XML

XML provides a mechanism for structuring a document, so that it can be exchanged and manipulated easily. It is easy to move the XML structured information from one place to another, or from one application to another application. In the real world applications the computer systems and the databases may contain incompatible data formats. However XML data is stored in plain text format; so it is possible to maintain hardware-independent and software independent data. Different applications can share data easily. XML reduces the complexity of exchanging data over incompatible systems. It is well known that XML is the solution to bring structure to unstructured content. XML makes easy to upgrade new operating system, new applications, or new browsers, without losing data. With XML, the data can be available to all kinds of "reading machines" like handheld computers, voice machines, news feeds, etc. XML documents have data elements, markup declarations (instructions for the XML parser)[6], and processing instructions (for the application program that is processing the data in the document). In XML, one can defines a new nested tag to provide more info about the content of a tag. Attributes of an element can reduce the complexity of the element. Moreover, XML-based data exchange is easy to implement for Web Services.

B. DTD

A DTD is made up of three main building blocks: ELEMENT, ATTLIST, and ENTITY. ELEMENT is the main building block of XML documents. In the DTD[7], XML elements are declared with an ELEMENT. An element definition has the following syntax: *<!ELEMENT element-name (element-content)> element-content* may be EMPTY, or data type, or sequences of children. Because ELEMENT is used to describe elements of a document and each element can contain children elements, the function of these elements is like a class in a structure program. ATTLIST provides extra information about elements. Following is a general syntax of an ATTLIST element: *<!ATTLIST element-name attribute-name attribute type default-value> element-name* is the name of element for which we declare an attribute. *Attribute-name* is a name of the attribute we want to declare. *Attribute-type* is a data type and *default-value* specifies default value of the attribute. Finally, ENTITY is used to define a shortcut for a common text in XML. Its syntax is as follows: *<!ENTITY name definition>*. Besides these there are some declarations in DTD, such as CDATA, PCDATA, #REQUIRED, #IMPLIED, etc. Their purpose is to declare the data type or the displaying conditions of elements or attributes in the document.

III. RESOURCE DESCRITION FRAME WORK (RDF)

RDF is a simple language that provides a flexible mechanism for describing web resources and relationships among them. RDF has three concepts: resource, property, and syntax. A resource is a substance that can be specified with Web identifiers (called Uniform Resource Identifiers, or URIs). A property is a resource that can hold names, possibly those of other resource. Syntax is a combination of resources, properties and values. In syntax, a property plays the role of the subject, a resource function the predicate, and a value the object. The syntax constitutes one record where the fact was described. So RDF provides a simple tuple model, <S, P, 0>, to express all knowledge. The interpretation of this statement is that subject S has property P with value 0, where S and P are resource URIs and 0 is either a URI or a literal value. RDF Schemas (RDFS) is the vocabulary description language of RDF, and it "extends" RDF by introducing a set of distinguished resources into the language, and help RDF define properties, kinds, and relationships of resources in RDF documents.

IV. CONSTRUCTING RDF

A GUI[8] component has been developed to generate an RDF structure for the well formatted xml file. Figure-2 shows an option to create a new RDF structure when click the button “Create RDF STRUCTURE”. The first input for this designed component is the number of elements of each record of the xml file as shown in fig-3. It generates the fields as shown in fig-4. Give element names and data type as shown in the figure-4. Then click Save and create”, it creates the fields and ready to accept data for the given elements as shown in figure-5. Enter the data as shown in figure-5 and click “Add data” , the data is saved. Now click “Generate RDF” button as shown in the figure-6. It generates an RDF structure for the saved data.

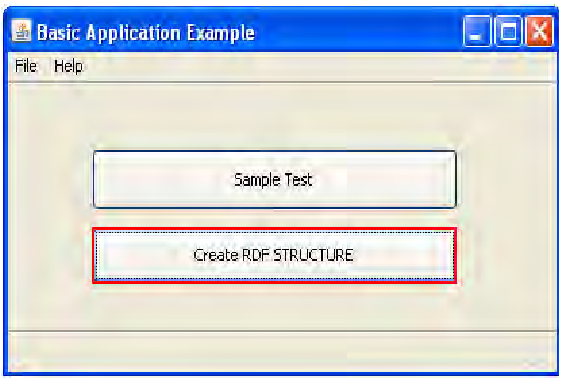

Figure -2: Create new RDF structure

Figure -3: Create Fields

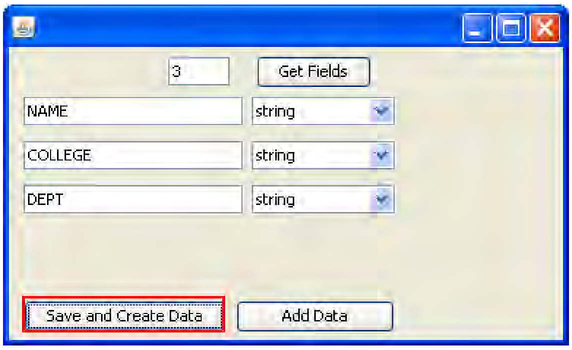

Figure -4: Enter Field Names

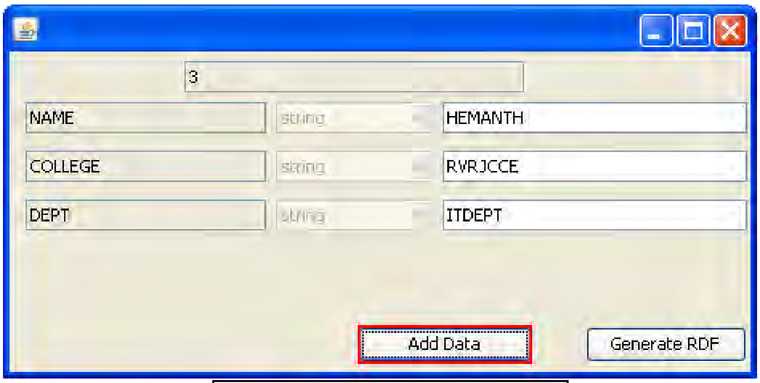

Figure -5: Enter Field Values

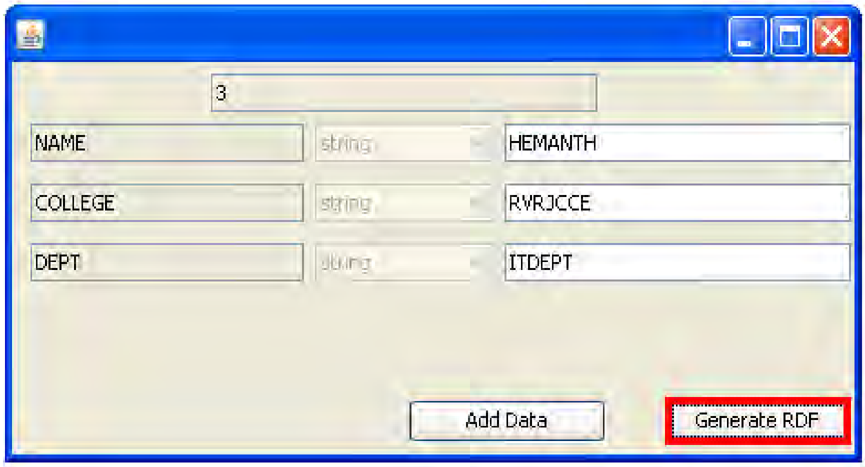

Figure -6: Generate RDF structure

V. RESULTS

Here we have generated an RDF structure for the given xml file. Figure-7: itdept.xml is a well formatted valid xml file and Figure -8 shows the corresponding Document Type Definition (DTD) .

```
<?xml version='1.0'?>
<!DOCTYPE student SYSTEM
'itdept.dtd'>
<itdept>
<record>

<NAME>HEMANTH</NAME>
<COLLEGE>RVRJCCE</COLLEGE>
<DEPT>ITDEPT</DEPT>

</record>
</itdept>
```

Figure -7 : Itdept xml data

```
<!ELEMENT itdept (record+)>
<!ELEMENT record (NAME,COLLEGE,DEPT)>
<!ELEMENT NAME (#PCDATA)>
<!ELEMENT COLLEGE (#PCDATA)>
<!ELEMENT DEPT (#PCDATA)>
```

Figure-8 : DTD for Itdept xml data

Constructing RDF structure for the above xml data.

Step 1: Select "*Create RDF Structre*" as shown in figure-2.

Step 2: Enter number of fields 3 and select "*Getfields*" as shown in figure-3.

Step 3: Enter field names (NAME, COLLEGE, DEPT) and type of the data (string) and then select "*Saveandcreatedata*" as shown in figure-4.

Step 4: Enter field values NAME: HEMANTH, COLLEGE: RVRJCCE, DEPT: ITDEPT and then select "*AddData*" as shown in figure-5. It saves the given data.

Step 5: After saving the data select "*GenerateRDF*" as shown in figure-6. It generate RDF as shown in figure-9.

The generated RDF can be represented as Triples shown in Table -1 and can be represented as Graphical model as shown in figure-10.

```
<?xml version="1.0" ?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:it="http://www.rvrjc.ac.in/it#">
<rdf:Description
rdf:about="http://rvrjc.ac.in/it/ITDEPT">
  <it:NAME>HEMANTH</it:NAME>
  <it:COLLEGE>RVRJCCE</it:COLLEGE>
  <it:DEPT>ITDEPT</it:DEPT>
</rdf:Description>
</rdf:RDF>
```

Figure-9: Generated RDF for the itdept xml data

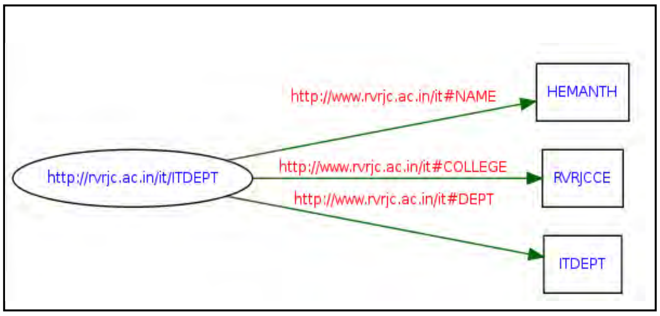

Figure-9: Graphical Model

Table : 1. Triple format

Number	Subject	Predicate	Object
1	http://rvrjc.ac.in/it/ITDEPT	http://www.rvrjc.ac.in/it#NAME	“HEMANTH”
2	http://rvrjc.ac.in/it/ITDEPT	http://www.rvrjc.ac.in/it#COLLEGE	“RVRJCCE”
3	http://rvrjc.ac.in/it/ITDEPT	http://www.rvrjc.ac.in/it#DEPT	“ITDEPT”

VI. CONCLUSIONS

As per the results shown by the above experiments, to improve the efficiency of the system with respect to data extraction, it is always preferable to maintain the data in XML format than RDB. Applications which will process only text data can directly interact with xml data sets. As per W3C recommendations, in Semantic web the data will be represented in RDF format which follows the XML syntax. So the RDF structure can be implemented in the Semantic web systems. Web services are dynamic technologies that will increasingly help to integrate independent systems, provided by business partners participating in multiple enterprises. We can build new databases based on open collaboration and standards (XML Web Services) that are accessible and interoperable and promote the use of XML Web Services[9][10]. The performance of all the dynamic web services depends up on the response time. By minimizing the response time we can increase the performance of a system. All the experiments summarized in the paper are purely dependent up on the system configuration, the processor and the type of the application programs currently executing in the system.

REFERENCES

[1] Ora Lassila, Ralph R. Swick, “Resource Description Framework (RDF) Model and Syntax Specification” W3C recommendation, Feb. 1999, http://www.w3.org/TR/1999/REC-Rdf-Syntax-19990222.
[2] Tim Berners-Lee, James Hendler, Ora Lassila, “The Semantic Web”, Scientific American Degital Magazine , Vol:284, No:5, May 2001, pages 29-37
[3] Stefan Decker, Prasenjit Mitra, and Sergey Melink, “Semantic Web:*An RDF Tutorial*”, IEEE INTERNET COMPUTING, NOVEMBER • DECEMBER 2000, pages 68-73.
[4] T. Bray, J. Paoli, C.M.Sperberg, McQueen, E. Maler, “XML 1.0 Extensible Markup Language (XML) 1.0 (Second Edition)”, W3C Recommendation, 6 October 2000, http://www.3.org/TR/2000/REC-xml-20001006
[5] Stefan Decker and Sergey Melink, “The Semntic Web: Roles of XML and RDF” IEEE INTERNET COMPUTING, SEPTEMBER • OCTOBER 2000, pages 63-74
[6] Wei Zhang and Robert A. van Engelen “An Adaptive XML Parser for Developing High-Performance Web Services” 978-0-7695-3535-7/08 © 2008 IEEE
[7] Jan Hegewald, Felix Naumann, Melanie Weis, “XStruct: Efficient Schema Extraction from Multiple and Large XML Documents” Proceedings of the 22nd International Conference on Data Engineering Workshops (ICDEW'06), 2006 IEEE, computer society of India.
[8] B.Hemanth kumar , Prof. M.Surendra Prasad Babu ,” Design and Development of a New JAVA Based Conversion Tool from Relational Database to XML Datasets” International Journal of Computer Science and Information Technologies, Vol. 3 (6), 2012,5439-5446.
[9] Chen Zhifeng, “Research on the Application of XML in Web Services”, 2009 IEEE International Conference on Multimedia Information Networking and Security, Wuhan, China, Vol:1, pages 680-683.
[10] Dr. Vladimir Tosic, Dr. Patrick C. K. Hung “Quality of Service Specification and Management for XML Web Services” 0-7695-2408-7/05 © 2005 IEEE
[11] J2EE Design Patterns By William Crawford, Jonathan Kaplan Publisher: O'Reilly Media
[12] Enterprise Applications in Java and J2EE Publisher: O'Reilly Media
[13] Java Web Services by David Chappell and Tyler Jewell
[14] XML Schema Part 0: Primer, W3C Working Draft, “http://www.w3.org /TR/xmlschema-0/”.
[15] http://w3schools.com/xml/xml_http.asp
[16] http://www.oracle.com/technetwork/java/javase/jdbc/index.html
[17] Cay Hostman, John Wiley&sons, “Big Java “ Pearson.