Journal of Information Engineering and Applications
ISSN 2224-5782 (print) ISSN 2225-0506 (online)
Vol 2, No.3, 2012

www.iiste.org

RROVT: A Proposed Visualization Tool for Semantic Web Technologies

Fagbola Temitayo* Olabiyisi Stephen , Adigun Abimbola

Department of Computer Science & Engineering, Ladoke Akintola University of Technology,
PMB 4000, Ogbomoso, Oyo State, Nigeria
* E-mail of the corresponding author: cometoty@yahoo.com

Abstract

Visualization is the graphical or semi-graphical representation of information that aids human comprehension of and reasoning about that information. Visualization tools are critically important for creating, querying, visualizing and validating Semantic Web Data. Semantic Web, an efficient infrastructure that enhances visibility of knowledge on the web, is often used more specifically to refer to the formats and technologies that enable it. These technologies include RDF, RDFS and OWL. However, lack of robust and efficient tools to visualize, analyze and represent these technologies within time and space constraint remains a big challenge. In this study, semantic web technologies and their visualization tools were reviewed. RROVT (RDF, RDFS, OWL Visualization Tool), a tool to evaluate and represent formal description of concepts, terms and relationships of data models within a given knowledge domain as well as manage time and space complexities for publishing contents of the semantic web more efficiently is developed and proposed. Performance of RROVT was investigated on samples of semantic web documents implementing RDF, RDFS and OWL technologies. The proposed tool showed remarkable improvement over the existing tools as it aids a better comprehension of the syntax and semantics of semantic web technologies investigated in this study.

Keywords: Semantic Web, Visualization Tool, Semantic Web Technologies

1. Introduction

The semantic web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation (Berners-Lee et al, 2001)**.** It is an efficient infrastructure that enhances visibility of knowledge on the web and is often used more specifically to refer to the formats and technologies that enable it (W3C, 2011). These technologies include RDF, RDFS and OWL which are used to improve retrieval, terminology, development and data mash-ups. RDF is the first layer that forms the Semantic Web. RDF is a data model for objects (resources) and relations between them. It provides simple semantics for data models, and these data models can be represented in XML syntax. RDF Schema is a vocabulary for describing properties and classes of RDF resources, with a semantics for generalization-hierarchies of such properties and classes (McGuinness & van Harmelen, 2004). On top of the basic RDF model is layered an extensible, object-oriented type system known as RDFS (Brickley & Guha, 2000). While RDFS is being used to represent the web resources, RDF is for representing the knowledge resources on the web and uses the web identifier URI (Uniform Resource Identifier) to identify the resources (Sanjay et al., 2010). The expressivity of RDF and RDFS are very limited. As a result, OWL was designed to add more vocabulary for describing properties and classes: among others, relations between classes (e.g. disjointness), cardinality (e.g. "exactly one"), equality, richer typing of properties and characteristics of properties (e.g. symmetry), and enumerated classes (Sabou, 2006). However, semantically enriched information becomes more useful if adequately visualized. Suvodeep (2010) identified visualization technologies as means to comprehend, understand and explore data. The author further stated

that observing patterns and anomalies via visualization tools help users to understand issues and take informed decisions. Several tools for visualizing semantic web technologies have been proposed. The ISX display navigates and visualizes the discovered entities and relationships contained in the source data. RDF Graph Visualization Tool (RDF Gravity) visualizes directed graphs built in RDF and OWL, and provides a simple but powerful visualization of RDF graph structures which has the ability to filter out and visualize specific parts or fragments of RDF Graphs. RacerPro shows the hierarchical structure of the classes and can be used to inspect the knowledge-base and view the hierarchy of concept names or roles. It mainly shows the taxonomy and role hierarchy respectively. OWLViz visualizes all or part of the asserted and inferred class hierarchies. However, most of these visualization tools lack expressiveness, generality and scalability for effective user interaction and conceptualization. This makes visualization, analysis and representation of semantic web technologies a big challenge. On the other hand, it is time consuming to create and publish contents of the semantic web with different formats for human viewing and machines due to its high dimensionality. To address these challenges, RROVT was developed and proposed in this study. The computation time and space constraints are managed by indexing the table structures contained in the database. The performance of RROVT was investigated on samples of semantic web documents implementing RDF, RDFS and OWL technologies.

2.0 Literature Review & Conceptual Understanding

This section discusses semantic web, semantic web technologies and the visualization tools for the semantic web technologies considered in this study.

2.1 Semantic Web

The Semantic Web is an evolution of the current web that provides meanings to web contents to enable their intelligent processing by computers (Berners-lee et al., 2001). Semantic Web is a vision: the idea 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 (W3C, 2011). It encompasses efforts to build a new world wide web architecture that augments content with formal semantics thereby producing content suitable for automated systems to consume, as opposed to content intended for human consumption (Berners-Lee et al, 2001). It is a graph-theoretic model of taxonomies, partonomies and ontology mappings. Semantic Web creates an artificial intelligence (AI) application which will make web content meaningful to computers, thereby unleashing a revolution of new abilities to support machine-processing capabilities that will automate web applications and services (Sanjay et al., 2010). It is a concept of how computers, people and the web can work together more effectively than is possible now (Thomas, 2001). Semantic Web extends the classical Web by providing it with a semantic structure of web pages to give support to human and artificial agents to understand the content. It further provides an environment where software agents can navigate through Web documents and execute sophisticated tasks (Ig et al., 2008). The main purpose of the semantic web is to drive the evolution of the current web by enabling users to find, share and combine information more easily.

2.2 Semantic Web Technologies

Semantic web technologies are used to improve retrieval, terminology, development and data mash-ups. The semantic web technologies considered include the RDF, RDFS and OWL.

2.2.1 Resource Description Framework (RDF)

RDF is the first layer that forms the Semantic Web. It belongs to a family of World Wide Web consortium specification originally designed as a metadata data model (Kalyanpur, 2006) such that the learning object metadata is stored in the learning object repository as RDF statements. RDF uses URIs to identify and locate resources and information on the Web. Pathmeswaran & Ahmed (2009) explained that RDF Schema is a modelling language for defining and describing classes of resources in the RDF model. Three concepts are fundamental in RDF including subject (S), predicate (P) and Object (O). A subject is a resource to be referred to on the Web; a predicate is a property describing the resource and an object is a value of the property. Thus, the triplet <S, P, O> describes the same information and forms an RDF graph or statement (Wilkinson et al., 2003). The meaning of a RDF statement is that subject S has property P with value O. In a RDF statement, S and P are uniform resource identifiers (URIs), while O is either a URI or a literal value (Wilkinson et al., 2003). An RDF graph or statement is graphically represented with two nodes and one arc; where, the arc is the property that describes the resource and the nodes at both sides of the arc, the resource and property's value respectively (Jean et.al, 2011).

2.2.2 RDF Schema (RDFS)

On top of the basic RDF model is layered an extensible, object-oriented type system known as RDF Schema (Brickley & Guha, 2000). RDFS is used to represent the web resource while RDF is for representing the knowledge resources on the web and uses the web identifier URI to identify the resources (Sanjay et al., 2010). RDFS extends RDF and is a vocabulary for describing properties and classes of RDF-based resources, with semantics for generalized-hierarchies of such properties and classes. RDFS is a set of primitives to describe lightweight ontologies in RDF using the RDF model and syntax while the RDF employs the ontologies to type resources and relations (Kalyanpur, 2006).

RDFS allows us to:

- name and declare a vocabulary (name resource types and binary relation types called properties);
- specify the signature of properties (type of the domain i.e. type of the subject and type of the range i.e. type of the object);
- specify the (multiple)-inheritance links between the types of classes (`subClassOf`);
- specify the (multiple)-inheritance links between the types of properties (`subPropertyOf`);
- provide labels and comments in natural language to document and display these primitives.

RDF Schema is a primitive ontology language. It offers certain modelling primitives with fixed meaning. Key concepts of RDF Schema are class, subclass relations, property, subproperty relations, domain and range restrictions (Grigoris Antoniou & Frank van Harmelen, 2004).

2.2.3 OWL

The expressivity of RDF and RDFS are very limited. RDF is roughly limited to binary ground predicates while RDFS is roughly limited to a subclass hierarchy and a property hierarchy, with domain and range definitions of these properties. Hence, OWL was developed. The Web Ontology Language OWL is a semantic markup language for publishing and sharing ontologies on the World Wide Web. It uses the XML/RDF exchange syntax (McGuinness & Van Harmelen, 2004). OWL extends RDFS with additional

predicates to characterize properties and classes.OWL adds more vocabulary for describing properties and classes: among others, relations between classes (e.g. disjointness), cardinality (e.g. "exactly one"), equality, richer typing of properties and characteristics of properties (e.g. symmetry), and enumerated classes.

2.3 Visualization Tools for RDF, RDFS and OWL Technologies

2.3.1 ISX Tool

This tool is a visualization and organization tool developed to display, navigate and visualize the discovered entities and relationships contained in the source data. It incorporates entity extraction technology which categorizes incoming data while ontologies provide information organization. Ontology hierarchy navigation and ontology element partial phrase matching are used to build queries. Query results display the relationships of the entities extracted from the documents and allows the user to navigate between data nodes (Vincent Wolowski, 2006). ISX is applying Semantic Web technology to address the visualization and organizational needs of today's analyst. However for ISX, as the amount of raw data available to analysts increase, the task of managing and exploiting the knowledge inside of this information space becomes increasingly difficult (Vincent Wolowski, 2006).

2.3.2 RDF Graph Visualization Tool

RDF Graph Visualization Tool (RDF Gravity) is a tool for visualizing directed graphs built in RDF and OWL. The tool provides a powerful visualization of RDF graph structures and characterized with the ability to filter out and visualize specific parts or fragments of RDF graphs.

Its main features are:

- Graph Visualization (Renderers/ Zoom/ Selection)
- Filters (Global, Local, Namespace, Instance level) enabling specific views on a graph.
- Full text Search/ RDQL queries
- Visualizing multiple RDF files

However, the drawback of this tool is that certain inner relationships and subclasses information cannot be analyzed, visualized or perfectly represented (W3C, 2011).

2.3.3 RacerPro

RacerPro is a tool that is used to show the hierarchical structure of the classes. It can be used to inspect the knowledge-base and to see the hierarchy of concept names or roles. It mainly shows the taxonomy and role hierarchy. Racerpro also suffers from the inability to represent certain inner relationships and subclasses information in a domain (Volker et al., 2010).

2.3.4 OWLViz (OWL Visualization)

OWLViz enables the class hierarchies in an OWL Ontology to be viewed and incrementally navigated, allowing comparison of the asserted class hierarchy and the inferred class hierarchy. OWLViz integrates with the Protege-OWL plugin, using the same color scheme so that primitive and defined classes can be distinguished, computed changes to the class hierarchy can be clearly seen, and inconsistent concepts are

highlighted in red. OWLViz makes it possible to view all or part of the asserted and inferred class hierarchies. However, unless the ontology that is being edited has been classified at least once, the classes in the inferred hierarchy will be displayed without any edges connecting (Vincent Wolowski, 2006).

3.0 Materials and Method

3.1 Methodology

A machine reader is designed to read and scan through RDF, RDFS and OWL documents. RDF, RDFS and OWL documents serve as the input to the system. After the machine has read through the input, tokens will be generated depending on the language type (i.e. RDF, RDFS or OWL). A database is designed to store the tokens for future use. The system identifies keywords in RDF, RDFS and OWL and stores them in the database. The tokens generated are analyzed. The system then maps the keywords with the tokens to identify the objects, classes, subclasses, properties and relationships. The system analyzes the classes, subclasses and the relationships that exist among objects. It then represents graphically, the taxonomy hierarchy of the classes and subclasses as well as their relationship and properties.

3.2 Components of the Proposed RROVT Model

The following components are designed and implemented: A Scanner, a tokenizer, a database, an analyzer and a visualizer.

3.2.1 RROVT Scanner

The RDF, RDFS or OWL documents are input into the scanner part of the system. The scanner accepts these inputs and read through it. The scanner verifies the input for correctness by checking whether the input document is in the right syntax and format it should be. The scanner then determines whether or not the input should be accepted and processed. If accepted, the documents are processed and then passed on to the next stage.

3.2.2 RROVT Tokenizer

This is the component that is responsible for tokenization of the string input that is taken from the scanner. After the scanner has accepted the RDF, RDFS or OWL documents, tokenization is the next stage. Tokenization is the process of breaking a stream of text up into words, phrases, symbols, or other meaningful elements called tokens. The list of tokens becomes input for further processing such as parsing or text mining. The set of delimiters (the characters that separate tokens) are specified at creation time and on a per-token basis.

The String Tokenizer behaves in the following ways:

- A flag is set to determine when to separate a substring from the string input based on a set of delimiters.
- If the flag is true, delimiter characters serve to separate tokens. A token is a maximal sequence of consecutive characters that are not delimiters.

- A delimiter is first used to break the string input into substring .The delimiter used separates the tokens into classes. Each substring thus represents a class in RDF, RDFS and OWL documents.

- Each substring is further broken into tokens to identify the subclasses in each of the classes and also the properties and other components in each class. Another set of delimiter is further used to do this.

A String tokenizer object internally maintains a current position within the string to be tokenized. A token is returned by taking a substring of the string that was used. The pseudo code illustrating the tokenizer is as follows:

3.2.3 RROVT Database

After tokenization, a database system is required to store the tokens generated, thus forming a component of the proposed system. The database system is responsible for the storage of the tokens taken by the scanner. This relational database is developed on phpMyAdmin platform running on MYSQL engine and served by WAMP server. A connection is made from the java application to MYSQL database through a connection string snippet. Before inserting data into the database, if-clauses are used to specify which part of the tokens will be inserted into the database tables and which table of the relational database it will be inserted into. Identification of classes and their subclasses with their relationship is done and they are inserted into appropriate tables. The tokens to be put in the database are then formatted to remove some unwanted characters and spaces. The algorithm below is used to insert classes and subclasses with their relationships into the table using the SQL.

3.2.4 RROVT Analyzer

The analyzer maps the keywords with the tokens to identify the objects, classes, subclasses, properties and relationships. The system then analyzes the classes, subclasses and the relationships that exist among them. The analyzer queries the database to identify the subclasses of a particular class and the relationships among them. It also queries the database to know if there are some classes that are related to some other classes and investigate their properties. The pseudo code below was used:

3.2.5 RROVT Visualizer

This is the last component of the proposed system. It is the last stage of all the processes. The visualizer displays the results generated by the analyzer methods in a graphical form. It illustrates conceptuality the taxonomy hierarchy of the classes and subclasses as well as their relationship and properties. The package also provides mouse clicking events. The major classes are first displayed. Clicking on these classes, the subclasses are displayed in a tree fashion and including their relationship and properties. The following algorithm is used to identify and conceptualize the vertices and edges in the graph.

3.3 Programming Tools

The visualization tool was developed using JDK 1.6.0 and a WAMP server running phpMyAdmin. JDK1.6.0 was used for the development of the front-end environment interfaces, phpMyAdmin was used to create the database and MySQL engine was used to store and manage the data. WAMP SERVER was used to serve the pages locally. Necessary query operations were carried out using MySQL.

4.0 Implementation, Results & Discussion

This section presents the RRO visualization tool for semantic web technologies including RDF, RDFS and OWL.

4.1 RROVT System Testing

4.1.1 The Input Reader

It accepts the OWL, RDF or RDFS document into the text area in the Scanner. This is done either by typing directly into the text area or by copying the document from somewhere and paste inside the text area.

4.1.2 The Database

The WAMP server is the local server used in the implementation. The WAMP server is composed of Apache and MYSQL database. It also has database administration tool, phpMyadmin, to administer and manage the database. The tokens generated are stored inside the database. They are also retrieved from the database whenever necessary.

4.1.3 RROVT tested using a sample semantic web document, newspaper.owl as input

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xml:base="http://www.owl-ontologies.com/Ontology1299666367.owl">
    <owl:Ontology rdf:about=""/>
    <owl:Class rdf:ID="Employee">
        <rdfs:subClassOf>
            <owl:Class rdf:ID="Person"/>
        </rdfs:subClassOf>
        <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Employee</rdfs:label>
    </owl:Class>
    <owl:Class rdf:about="#Person">
        <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Person</rdfs:label>
    </owl:Class>
    <owl:Class rdf:ID="Personals_Ad">
        <rdfs:subClassOf>
            <owl:Class rdf:ID="Advertisement"/>
        </rdfs:subClassOf>
        <rdfs:label
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Personals_Ad</rdfs:label>
    </owl:Class>
    <owl:Class rdf:ID="Content_Layout">
        <rdfs:label
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Content_Layout</rdfs:label>
        <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Defines a
rectangle of space on a page</rdfs:comment>
        <rdfs:subClassOf>
            <owl:Class rdf:ID="Layout_info"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Billing_Chart">
        <rdfs:label
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Billing_Chart</rdfs:label>
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Layout_info"/>
        </rdfs:subClassOf>
    </owl:Class>
```

```
<owl:Class rdf:ID="Article">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Article</rdfs:label>
    <rdfs:subClassOf>
        <owl:Class rdf:ID="Content"/>
    </rdfs:subClassOf>
        </owl:Class>
<owl:Class rdf:ID="Standard_Ad">
    <rdfs:subClassOf>
        <owl:Class rdf:about="#Advertisement"/>
    </rdfs:subClassOf>
    <rdfs:label
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Standard_Ad</rdfs:label>
</owl:Class>
<owl:Class>
    <owl:unionOf rdf:parseType="Collection">
        <rdf:Description rdf:about="http://www.w3.org/2002/07/owl#Thing"/>
        <owl:Class rdf:about="#Person"/>
        <owl:Class rdf:about="#Advertisement"/>
    </owl:unionOf>
</owl:Class>
<owl:Class rdf:ID="Director_Supervision_Relation">
    <rdfs:subClassOf>
        <owl:Class rdf:ID="Manager_Supervision_Relation"/>
    </rdfs:subClassOf>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Director Supervision
Relation</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="Organization">
    <rdfs:label
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Organization</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="Columnist">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Columnist</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Employee"/>
    <rdfs:subClassOf>
        <owl:Class rdf:about="#Author"/>
    </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Author">
        </rdf:RDF>
```

This output displays classes, their subclasses, their relationships and properties at any level. A class, Standard_Ad is a subclass of Advertisement and Advertisement is a subclass of Contents. Personal_Ad is a subclass of Advertisement and Advertisement is a subclass of Contents. Article is a subclass of content. A columnist is a subclass of class Author. It goes on and on like that.

4.1.4 RROVT tested using a sample semantic web document, pizza.rdfs as input

```
<?xml version="1.0"?>
<rdf:RDF xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.rdfs#"
xmlns:swrlb="http://www.w3.org/2003/11/swrlb#" xmlns:swrl="http://www.w3.org/2003/11/swrl#"
xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#" xmlns:owl="http://www.w3.org/2002/07/owl#"
```



```
        <owl:imports rdf:resource="http://protege.stanford.edu/plugins/owl/protege"/>
        <protege:defaultLanguage
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">en</protege:defaultLanguage>
    </owl:Ontology>
    <owl:Class rdf:ID="DeepPanBase">
        <owl:disjointWith>
            <owl:Class rdf:ID="ThinAndCrispyBase"/>
        </owl:disjointWith>
        <rdfs:subClassOf>
            <owl:Class rdf:ID="PizzaBase"/>
        </rdfs:subClassOf>
        <rdfs:label xml:lang="pt">BaseEspessa</rdfs:label>
    </owl:Class>
    <owl:Class rdf:ID="VegetarianTopping">

    <owl:Class rdf:ID="CajunSpiceTopping">
        <owl:disjointWith>
            <owl:Class rdf:ID="RosemaryTopping"/>
        </owl:disjointWith>
        <rdfs:label xml:lang="pt">CoberturaDeCajun</rdfs:label>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:someValuesFrom>
                    <owl:Class rdf:ID="Hot"/>
                </owl:someValuesFrom>
                <owl:onProperty>
                    <owl:FunctionalProperty rdf:ID="hasSpiciness"/>
                </owl:onProperty>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Class rdf:about="#HerbSpiceTopping"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="CheeseyVegetableTopping">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#VegetableTopping"/>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Class rdf:about="#CheeseTopping"/>
        </rdfs:subClassOf>

    <owl:Class rdf:ID="HotGreenPepperTopping">
        <rdfs:label xml:lang="pt">CoberturaDePimentaoVerdePicante</rdfs:label>
        <rdfs:subClassOf>
            <owl:Class rdf:ID="GreenPepperTopping"/>
        </rdfs:subClassOf>
        <rdf:subClassOf>
            <rdfs:Restriction>
                <rdfs:someValuesFrom>
                    <rdfs:Class rdf:about="#Hot"/>
```

```
                    </rdfs:someValuesFrom>
                    <rdfs:onProperty>
                        <rdfs:FunctionalProperty rdf:about="#hasSpiciness"/>
                    </rdfs:onProperty>
                </rdfs:Restriction>
            </rdf:subClassOf>
            </rdfs:Class>
```

This result is showing classes, subclasses and their relationships at any level. For example, a class CajunSpiceTopping is a subclass of HerbSpiceTopping and HerbSpiceTopping is a subclass of Pizza Topping. CheeseVegetableTopping is a subclass of both Cheese Topping and Vegetable Topping. Both Cheese Topping and Vegetable Topping have a subclass of Pizza Topping.

4.1.5 Sample semantic web document, printer.rdf tested on RROVT.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3 .org/1999/02/22-rdf-syntax-nsi"
xrnlns:rdf="http://www.w3.org/2000/01/rdf #"
xmlns:xsd="http://www.w3 .org/2001/XMLSchemaI" xmlns:owl ="http://www.w3.org/2002/07/owl"
xmlns="http://www.cs.vu.nh/..frankh/spool/printer.rdf">
<rdf:Ontology rdf:about=""/>
<rdf:versionlnfo>
</rdf:versionlnfo>
<rdf:Ontology>
<rdf:Class rdf:ID="product">
<rdf:subClassOf>
<rdf:Class rdf:ID="Printer"/>
</rdf:subClassOf>
</rdf:Class>
<rdf:Class rdf:ID="padid">
<rdf:subClassOf>
<rdf:Class rdf:ID="Printer"/>
</rdf:subClassOf>
<rdf:label>Device</rdf:label>
<rdf:subClassOf rdf:resource="#product">
</rdf:Class>
<rdf:Class rdf:ID="hpproduct">
<rdf:subClassOf rdf:resource="#product">
</rdf:Class>
<rdf:Class rdf:ID="printer">
<rdf:subClassOf rdf:resource="#personalprinter">
</rdf:Class>
<rdf:Class rdf:ID="hpseries">
<rdf:subClassOf rdf:resource="#hplaserjet">
</rdf:Class>
```


<rdf:Class rdf:ID="1100series">

<rdf:subClassOf rdf:resource="#1100xi">

</rdf:Class>

<rdf:Class rdf:ID="hplaserjet">

<rdf:subClassOf rdf:resource="#hpprinter">

<rdf:Class rdf:ID="hpprinter">

<rdf:subClassOf rdf:resource="#1100series">

</rdf:Class>

</rdf:Class>

</rdf:RDF>

5.0 Conclusion & Future Works

In this paper, a critical review of the semantic web, its technologies and their visualization tools was carried out. The need for a more robust and efficient visualization tool for these technologies was identified and discussed. RROVT, a visualization tool for RDF, RDFS and OWL documents is developed and proposed. The tool can evaluate and represent formal description of concepts, terms and relationships of data models within a given knowledge domain as well as manage time and space complexities for publishing contents of the semantic web more efficiently. The computation time and space constraints are managed by indexing the table structures contained in the database. This approach facilitated search and retrieval of required information from the table structures. Performance of RROVT was investigated on samples of semantic web documents implementing RDF, RDFS and OWL technologies. While RROVT showed remarkable improvements over the existing visualization tools, future research work should focus on the development of the visualization tool functionalities such as visualization of the union classes and disjoint classes in semantic web documents and distinguish between the properties such as symmetric properties ,transitive properties etc.

References

Berners-Lee T., Hendler J., Lassila O. (2001). The Semantic Web. *Scientific American*, 284, 35-43.

Brickley & Guha (2000), "Resource Description Framework (RDF) Schema Specification 1.0", W3C Candidate Recommendation, World Wide Web Consortium, Cambridge (MA); available on-line as http://www.w3.org/TR/rdfschema/.

Grigoris Antoniou & Frank van Harmelen (2004). A Semantic web Primer. The MIT Press Cambridge, Massachusetts, pp 30-150, 2004.

Ig Ibert Bittencourt, Seiji Isotani, Evandro Costa, Riichiro Mizoguchi (2008). Research directions on Semantic Web and education. Scientia Interdisciplinary Studies in Computer Science 19(1): 60-67.

Jean Vincent, Fonou-Dombeu & Magda Huisman (2011). Combining Ontology Development Methodologies and Semantic Web Platforms for E-government Domain Ontology Development. International Journal of Web & Semantic Technology (IJWesT) Vol.2, No.2, Pp 12-25.

Kalyanpur, A. (2006). “Debugging and Repair of OWL Ontologies, ” *PhD Thesis*, Faculty of Graduate School, University of Maryland, USA, pp.1-3.

McGuinness, D.L. & Van Harmelen, F. (2004). *OWL Web Ontology Language Overview. W3C Recommendation*. The MIT Press.

Pathmeswaran, R. & Ahmed, V. (2009). The Built & Human Environment Review, Volume 2, Special Issue 1, SWmLOR: Technologies for Developing Semantic Web based Mobile Learning Object Repository.

Sanjay Kumar Malik, Nupur Prakash & S.A.M Rizvi (2010). Developing an University Ontology in Education Domain using Protégé for Semantic Web. International Journal of Engineering Science and TechnologyVol. 2(9), 4673-4681.

Sabou M. (2006). "Building Web Service Ontologies, " *PhD Thesis*, Dutch Graduate School for Information and Knowledge Systems, Netherlands, p. 1&17.

Suvodeep Mazumdar (2010). Visualization of Large Datasets using Semantic Web Technologies.

Thomas R. (2001). Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition*, 5(2),199-220.

Vincent Wolowski (2006). Seminar Visualization of Information, University Hagen, Department of Computer Science, Chair of Internet Applications of Prof.Dr.Hemmje, Seminar Visualization of Information, pp6-45 ,Dipl.Inf.G.Jäschke.

W3C 2011. Semantic Web Activity. Retrieved Nov. 26, 2011.

K. Wilkinson, C. Sayers, H. Kuno and D. Reynolds (2003). "Efficient RDF Storage and Retrieval in Jena2," *In proceedings of the First International Workshop on Semantic Web and Databases (SWDB)*, Berlin, Germany, pp.131-150.

Volker Haarslev, Kay Hidde, Ralf Moller & Michael Wessel (2010). The RacerPro Knowledge Representation and Reasoning System. www.semantic-web-journal.net. Retrieved 09-02-2012.

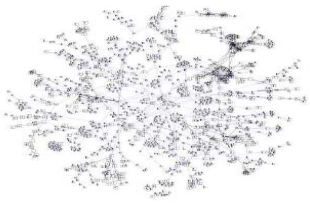

Figure 1: ISX DISPLAYING GRAPHS OF AN OWL DOCUMENT

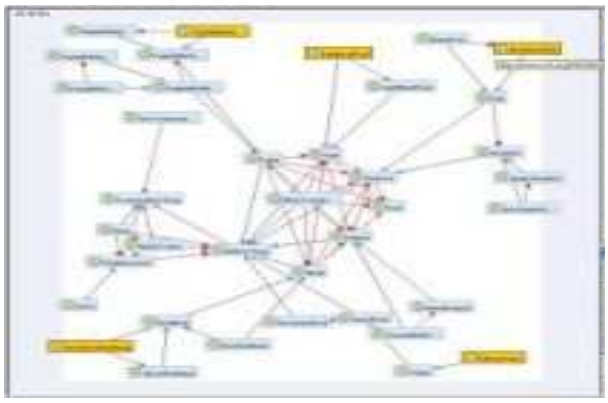

Figure 2: RDF GRAPH DISPLAYING GRAPHS OF A DOCUMENT

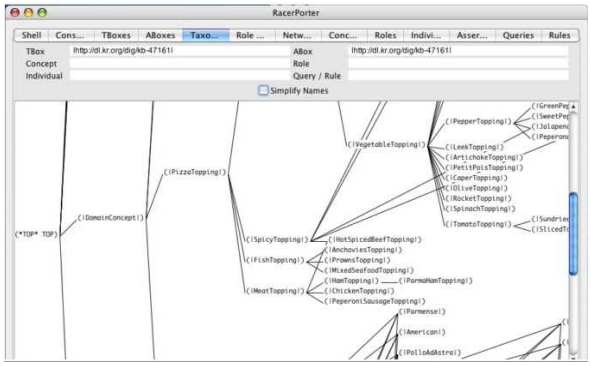

Figure 3: RACERPRO DISPLAYING GRAPHS OF AN OWL DOCUMENT

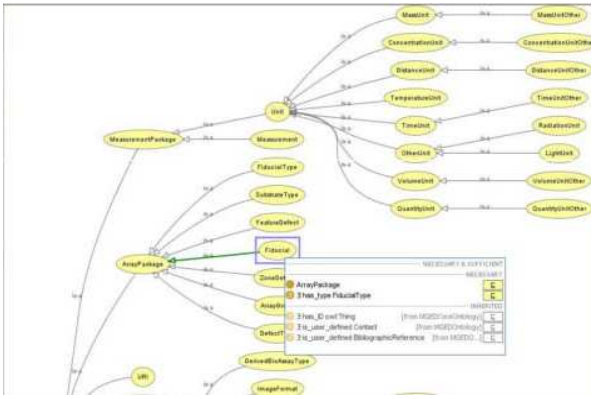

Figure 4: OWLVIZ DISPLAYING GRAPHS OF AN OWL DOCUMENT

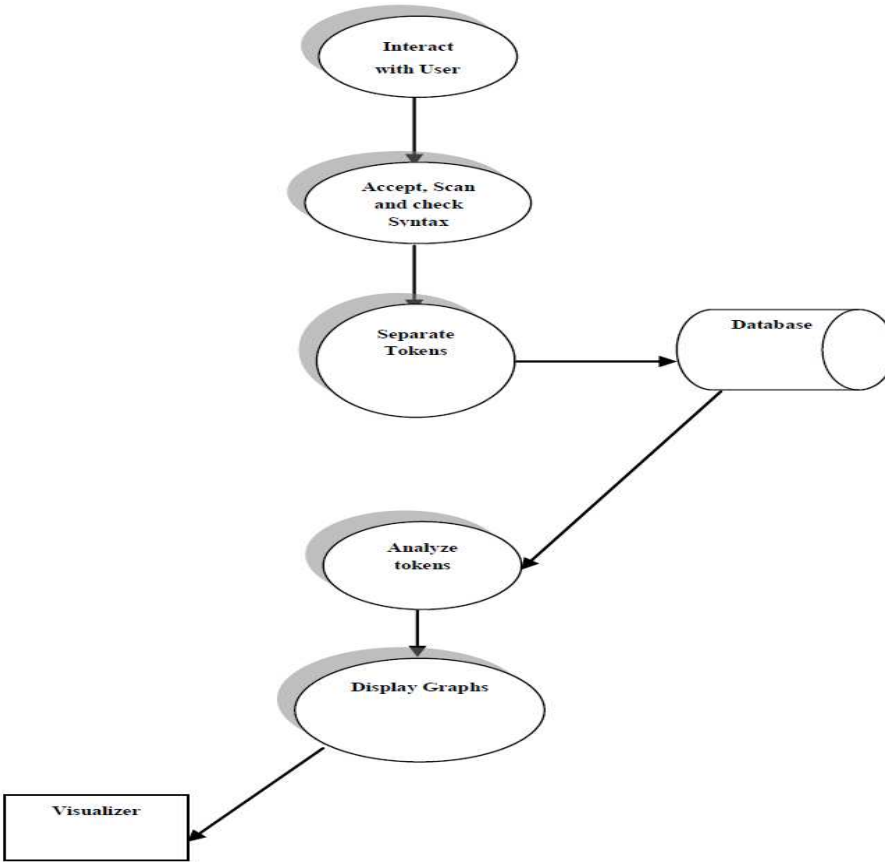

Figure 5: System Flow Diagram for the Proposed RRO Visualization Tool


```
1   Set  substring counter to zero
2   Prompt the user to enter the string
3   Set delimiter to equal owl: Class
4   If delimiter is equal to owl: Class, go to 7
5   If delimiter is equal to RDF: Class, go to 8
6   If delimiter is equal to RDFS: Class, go to 9
7  Increment the owl class counter, go to 10
8  Increment the RDF class counter, go to 10
9  Increment the RDFS class counter, go to 10
10   assign a class label to string and break the string to substring
11  Set array substring to the substrings generated
12   Set substring counter to zero
13    While substring counter less than the number of substring
14   Set delimiter to equal =, <,'>
15    If delimiter is equal to =, <,'> break the substring to tokens
16   Set array tokens to the tokens generated
17   Output the next tokens
18    Send Tokens into database
19  Add one to the substring counter
```

Figure 6: Part of the pseudo code for the tokenizer

```
1    Set counter equal zero
2    Identify class as owl, RDF, RDFS
3    Set identifier1 to equal OWL: Class
4    Set identifier2 to equal OWL: SubClass
5    Set identifier3 to equal OWL: Property
6    Set identifier1 to equal RDF: Class
7   Set identifier2 to equal RDF: SubClass
8    Set identifier3 to equal RDF: Property
9    Set identifier1 to equal RDFS: Class
10   Set identifier2 to equal RDFS: SubClass
11   Set identifier3 to equal RDFS: Property
12   While counter less than the number of substring
13   If tokens is equal to identifier1
14   put next token into the database
15   If tokens is equal to identifier2
16   put next token into the database
17   If tokens is equal to identifier3
```

Figure 7: Part of the pseudo code for token insertion to the database


```
1   Initialize counter to zero
2   Click to get the class name
3    While    counter less than the number of substring
4   Select class from database
5   If class equal class name
6   Print Class
7   Print Subclass
8   Print Properties
9   Add one to the counter
```

Figure 8: Part of the pseudo code for the analyzer

```
1 Initialize counter to zero
2 Click to get the class name
 3 While counter less than the number of substring
4 If class equal class name then
 5  Draw vertexes
6  Insert class in vertex
7  Draw edge
8  Insert properties in vertex
9  Draw vertexes
10  Insert subclass in vertex
11   Add one to the counter
```

Figure 9: The pseudo code for the RROVT Visualizer:

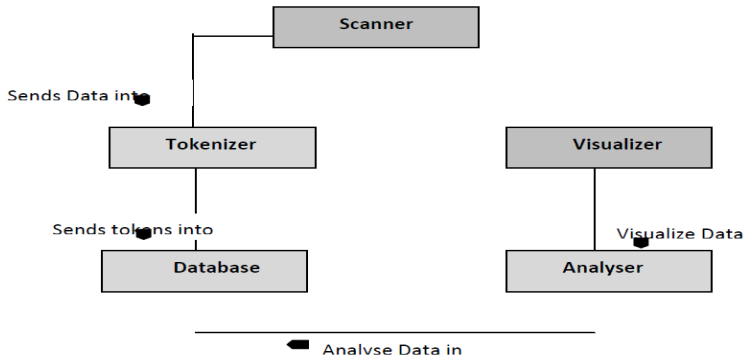

Figure 10: Class Diagram for the RROVT System Model

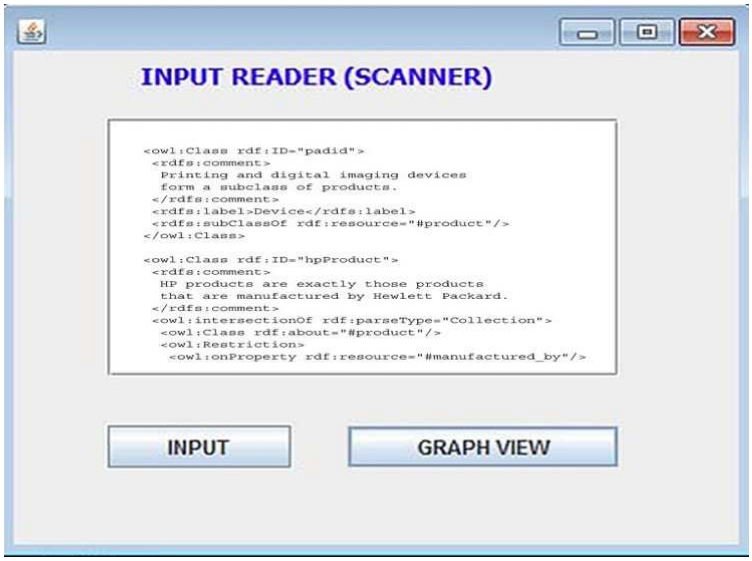

Figure 11: RROVT Scanner

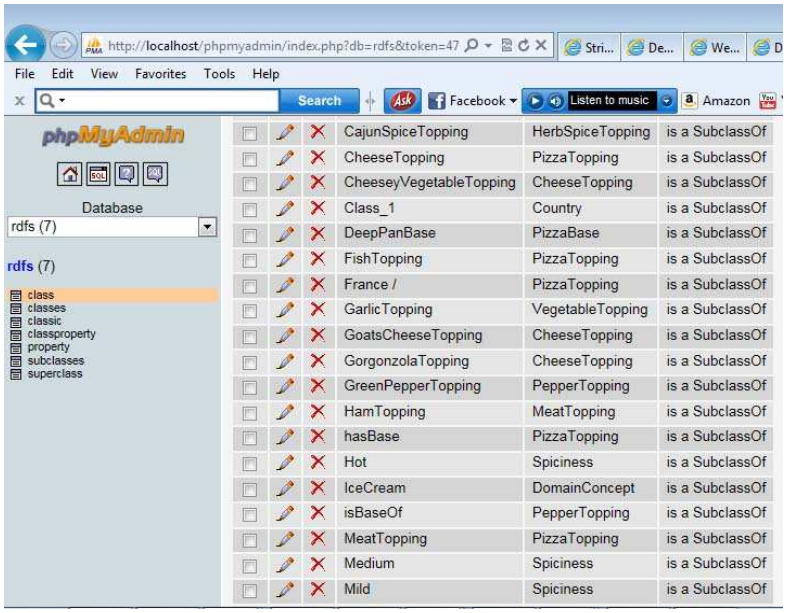

Figure 12: Part of the RROVT Database Structure

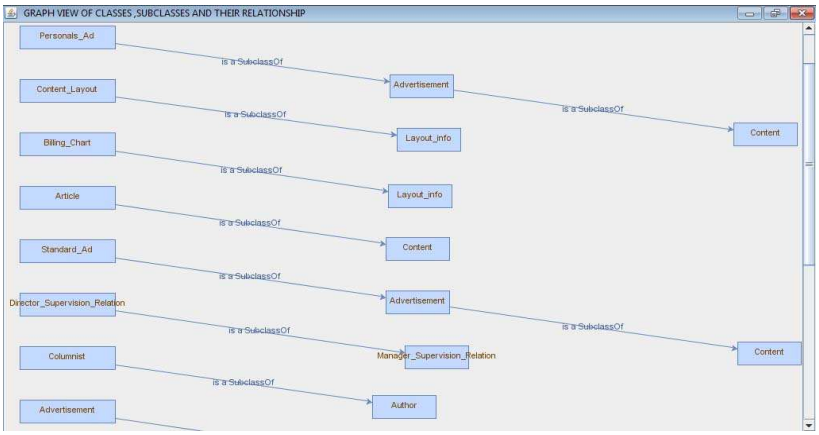

Figure 13: Part of the RROVT output for newspaper.owl

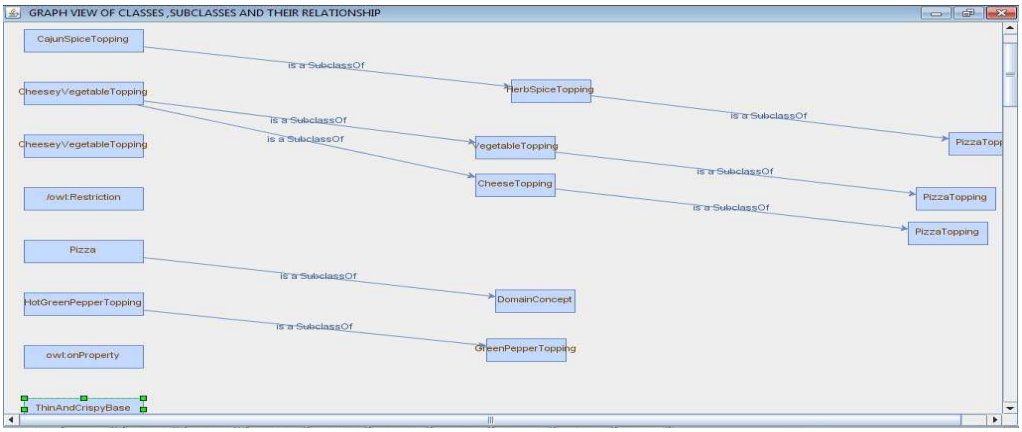

Figure 14: Part of the RROVT output for pizza.rdfs

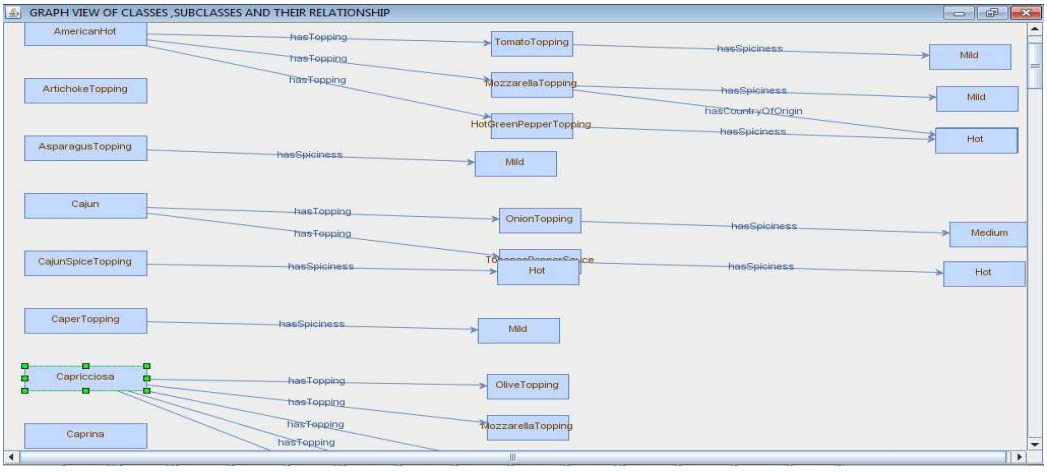

Figure 15: An output showing the relationships between classes in pizza.rdfs

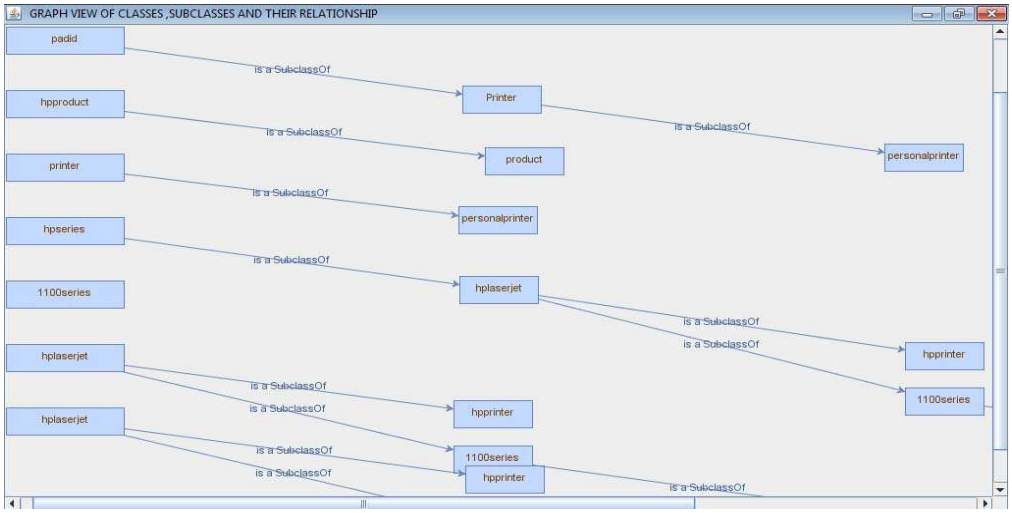

Figure 16: A part of the output of printer.rdf