J. Basic. Appl. Sci. Res., 3(3)xxxx-xxx, 2013

ISSN 2090-4304
Journal of Basic and Applied Scientific Research
www.textroad.com

Graphical Ontology and Validation Process for University Fee Structure

Sana Rizwan, Dr. Javaid Sikandar Mirza

Department of Computer Science
COMSATS Institute of Information Technology
Lahore, Pakistan

ABSTRACT

Manipulation of various species of OWL for ontology building and the validation of ontology results using the SPARQL for dynamic structures are the important tasks in web semantics. Direct growth for semantic modeling case supports to RDF, RDFS, RDF PLUS and OWL, for that purpose ontology development is done after schema building. The inter relationship between different components, makes classes, sub classes, individuals, properties and association between them. We have modeled an ontology containing fee structure of COMSATS. In this ontology we have identified various classes, properties, individuals and forms, and then generate graphs that show the relationship between the different components of schema generation. Graphs are the ontology key representation that identifies the value-able information via visual effects. For the validation process in dynamic structure, query using SPARQL will show the results that fetch out from the semantic ontology. Developed ontology is used to answer queries of students i.e. provide me "Registration Charges" of BSCS (Bachelor of Sciences in Computer Science) program of Semester 1 in Islamabad Campus or what are pre-requisites of MSTE (Masters of Sciences in Telecommunication Engineering) program in Lahore Campus or what are the credit hours of MCS (Masters of Computer Science) in Abbotabad Campus and Attock Campus etc. Results are compiled and assessed correctly by SPARQL query that will work with RDF/OWL.

KEYWORDS: Semantic Modeling, Ontology Building, Querying through SPARQL/XQuery, Inter-operatible Graphical Design, RDF/RDFS/OWL Modeling, SPARQL Protocol, Dynamic Semantic Constraints.

1. INTRODUCTION

COMSATS University has a number of undergraduate programs at its seven campuses. Due to large number of departments and programs, it is difficult for prospective students wishing to enroll to search the fee structure. Usually students get exasperated to find that the desired information is not available or inappropriately written.[1] Moreover traditional university containing fee structure web sites are not interactive, they cannot handle the synonym values as the user can use in the natural language with versatile cases. Beside this user have not enough time to search required information in many hyperlinks generated by different search engines. The problem of difficulty in fee structure search can be solved with RDF/OWL modeling. Open source ontology editor "Protégé" is used for ontology modeling. Query answering is supported by SPARQL (SPARQL Protocol and RDF Query Language). Data from different data sources like XML, HTML define referential integrity, delineate schema and the SPARQL clients fetch the results from ontology by querying through SPARQL or XQuery [20]. Ontologies support interoperability between various systems and problems of interoperability can be solved by controlling the semantics. RDF modeling has an advantage that the information from various university resources can be merged as shown in Figure 1.

***Corresponding Author:** Sana Rizwan, Department of Computer Science, COMSATS Institute of Information Technology, Lahore, Pakistan. Email: sana@ciitlahore.edu.pk Cell 92-300-4201200

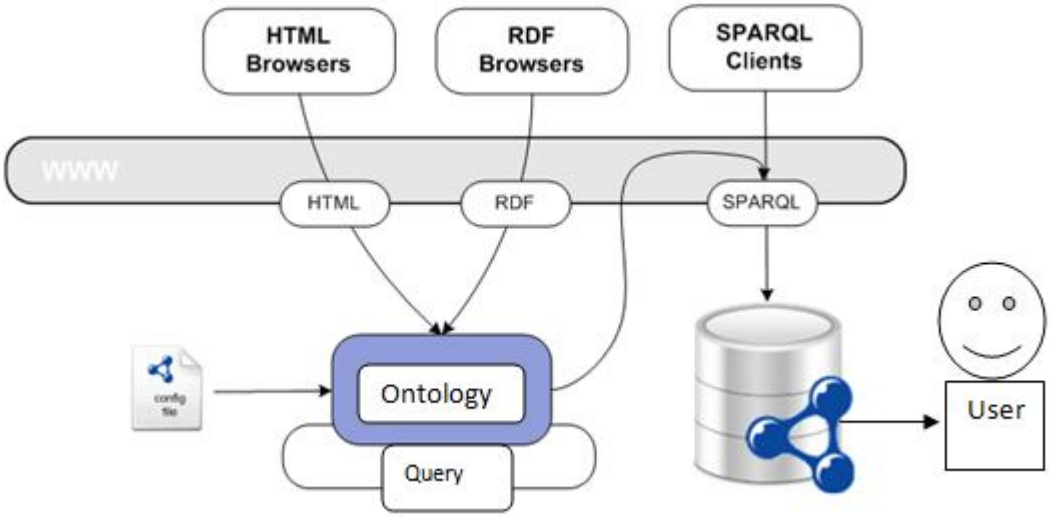


Figure 1: Clients query using SPARQL

Semantic web services is basically the alternated approach to traditional web, where user can get only accurate result of their queries that depends upon the keywords searching, shown in Figure 3, as contrast to multiple records in the form of search engine as shown in Figure 2. Therefore our queries have to be very precise, exact and correct.

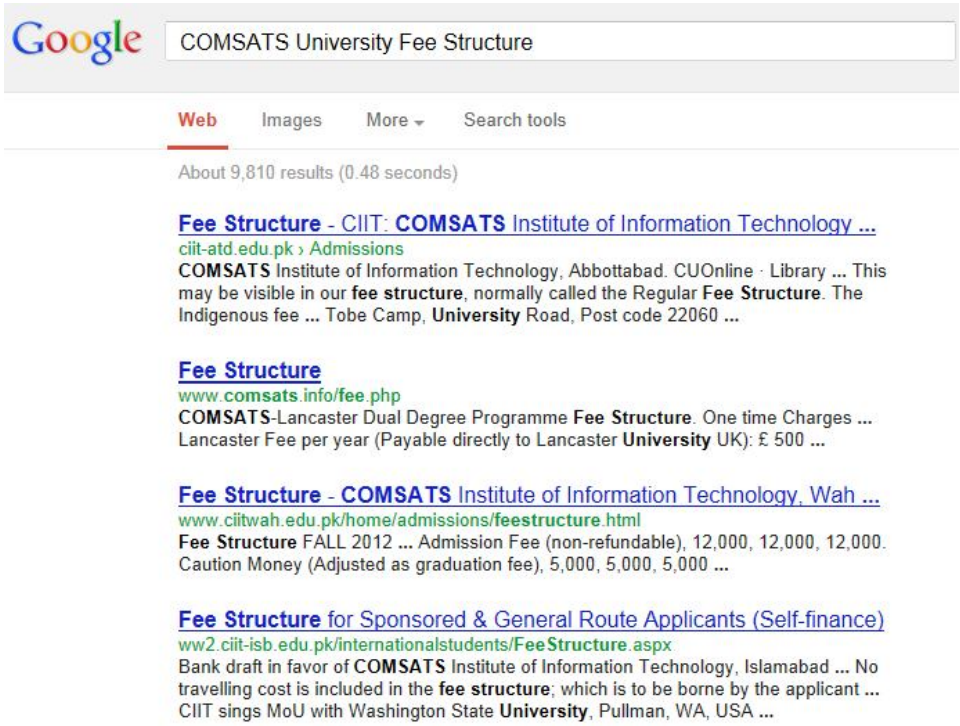


Figure 2: Searching COMSATS University Fee Structure in Traditional Web

Figure 3: Searching COMSATS University Fee Structure in Semantic Web

Sharing of information among the different data sources classifies the variety of data available, which will be accessible in classes, subclasses and associated properties. They depend on each other to share and reuse that information from the existing ontologies as well as from new ontologies that are not developed yet. So one way is to merge that noval approach of ontology module extraction process [20] and applied queries to proof them.[2] Other way is to widen your own ontology then introduce an abstract graph model for graph module that is independent of the language and express the ontology inheritance. The question arises what will be the data sources to build a new ontology, it can be XML, HTML, and SGML etc. Then how to translate XML file to RDF, RDF specifies RDFS flexibility, whereas illustrate its OWL functionality [19]. Another example was in front of us about news bulletin, the whole scenario developed for that specified crate [20]. We worked and applied the techniques with modified properties like synonyms, union etc, that are much superior. As referenced to our previous research paper [23], where we build fee ontology using RDF, RDFS and OWL, to validate this work we have to be more specific and detailed about ontology, then querying it and find out accurate results. In our university fee structure ontology validation process, firstly, nature of ontology is build uniquely that is not developed earlier [23], secondly, the validation process is proved in this article using SPARQL queries and their related graphs. In future, fee structure of other universities can also be integrated in the developed fee model. More over concatenation of multiple ontologies and find out integrated results [22].

The Scientific contributions of this paper are:

1. Goal of the endeavor is to build a unified information medium that is both understandable for people and computers and that can be used for the automatic deduction of meaningful inferences.
2. Ontology will be generated after normalization procedure, therefore the raw data is not the part of the ontology that's why it is unique data.
3. Building of a unique fee ontology structure that can be applied for any university and the parameters can be modified according to the university demands.
4. Conversion of XML table formats to XML schema and then to RDF/RDFS/OWL.
5. Building of ontology graphs that express their ontology hierarchy and articulate their detailed attributes.
6. Validation process to verifiy the ontology building is the major aim in our scenerio. Validation is done by organize queries that will retrieve the correct, exact and unique data. One more advantage for that purpose is synonym property in which the functionality of semantic web is that it doesn't depends upon keyword based search. Mapping of multiple synonyms to each other for getting one perfect answer is the inimitable work. Synonym property is the major to differentiate semantic web with traditional web. For-example, add synonym property in our fee structure ontology, by which a user can enter synonyms words to query a particular field and get its values. For example Dual_Degree_Program has synonyms like DDP, Lancaster Program, Multi Degree, Dual Program, Dual degree, LanCom Degree etc as shown in Figure 4. A user can search by any of the names that are associated with it as synonyms, by adding text field in the user interface.

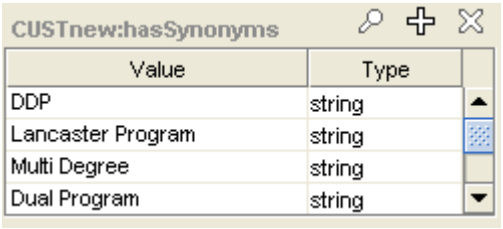


Figure 4: HasSynonyms Property

METHODOLOGY

XML and XML Schema Files

This is the XML file that is develop from a relational table elaborated in previous research paper [23] after passing by normalization process. Due to the lesser space we have pasted only a small instance [3][18][21]. For building of XML file we have to be very careful about the classes, sub classes, individuals, objects, forms, domain/ranges, properties and their parameters as shown in Figure 5(a).

```
<?xml version="1.0" ?>

<knowledge_base
        xmlns="http://protege.stanford.edu/xml"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
        xsi:schemaLocation="http://protege.stanford.edu/xml
http://protege.stanford.edu/xml/schema/protege.xsd">

        <class>
                <name>http://www.w3.org/2002/07/owl#Thing</name>
                <type>http://www.w3.org/2002/07/owl#Class</type>
                <own_slot_value>
                        <slot_reference>:ROLE</slot_reference>
                        <value value_type="string">Abstract</value>
                </own_slot_value>
                <own_slot_value>
                        <slot_reference>http://www.w3.org/1999/02/22-rdf-syntax-ns#type</slot_reference>
                        <value value_type="class">http://www.w3.org/2002/07/owl#Class</value>
                </own_slot_value>
                <template_slot>http://www.w3.org/2002/07/owl#differentFrom</template_slot>
                <template_slot>http://www.w3.org/2002/07/owl#sameAs</template_slot>
                <template_slot>http://www.w3.org/2002/07/owl#versionInfo</template_slot>
                <template_slot>http://www.w3.org/1999/02/22-rdf-syntax-ns#type</template_slot>
                <template_slot>http://www.w3.org/1999/02/22-rdf-syntax-ns#value</template_slot>
                <template_slot>http://www.w3.org/2000/01/rdf-schema#comment</template_slot>
                <template_slot>http://www.w3.org/2000/01/rdf-schema#isDefinedBy</template_slot>
                <template_slot>http://www.w3.org/2000/01/rdf-schema#label</template_slot>
                <template_slot>http://www.w3.org/2000/01/rdf-schema#member</template_slot>
                <template_slot>http://www.w3.org/2000/01/rdf-schema#seeAlso</template_slot>
                <template_slot>http://protege.stanford.edu/plugins/owl/protege#SLOT-
CONSTRAINTS</template_slot>
                <template_slot>http://www.w3.org/2003/11/swrl#body</template_slot>
                <template_slot>http://www.w3.org/2003/11/swrl#head</template_slot>
                <template_slot>http://www.w3.org/2003/11/swrl#arguments</template_slot>
                <template_slot>http://www.w3.org/2003/11/swrl#builtin</template_slot>
                <template_slot>http://www.w3.org/2003/11/swrl#argument1</template_slot>
                <template_slot>http://www.w3.org/2003/11/swrl#classPredicate</template_slot>
                <template_slot>http://www.w3.org/2003/11/swrl#propertyPredicate</template_slot>
                <template_slot>http://www.w3.org/2003/11/swrl#dataRange</template_slot>
                <template_slot>http://www.w3.org/2003/11/swrlb#args</template_slot>
                <template_slot>http://www.w3.org/2003/11/swrlb#minArgs</template_slot>
                <template_slot>http://www.w3.org/2003/11/swrlb#maxArgs</template_slot>
                <template_slot>http://www.w3.org/2003/11/swrl#argument2</template_slot>
         </class>
        <class>
                <name>http://www.w3.org/2002/07/owl#Class</name>
                <type>http://www.w3.org/2002/07/owl#Class</type>
                <own_slot_value>
                <slot_reference>http://www.w3.org/1999/02/22-rdf-syntax-ns#type</slot_reference>
                <value value_type="class">http://www.w3.org/2002/07/owl#Class</value>
                </own_slot_value>
                <superclass>http://www.w3.org/2000/01/rdf-schema#Class</superclass>
         </class>
        <class>
                <name>http://protege.stanford.edu/plugins/owl/protege#ExternalClass</name>
                <type>http://www.w3.org/2000/01/rdf-schema#Class</type>
                <own_slot_value>
                <slot_reference>http://www.w3.org/1999/02/22-rdf-syntax-ns#type</slot_reference>
                <value value_type="class">http://www.w3.org/2000/01/rdf-schema#Class</value>
                </own_slot_value>
                <superclass>http://www.w3.org/2000/01/rdf-schema#Class</superclass>
```

```
 </class>
<class>
        <name>http://www.w3.org/2002/07/owl#UnionClass</name>
        <type>http://www.w3.org/2000/01/rdf-schema#Class</type>
        <own_slot_value>
        <slot_reference>http://www.w3.org/1999/02/22-rdf-syntax-ns#type</slot_reference>
        <value value_type="class">http://www.w3.org/2000/01/rdf-schema#Class</value>
        </own_slot_value>
        <superclass>http://www.w3.org/2002/07/owl#LogicalClass</superclass>
 </class>
<class>
        <name>http://www.w3.org/2002/07/owl#DatatypeProperty</name>
        <type>http://www.w3.org/2002/07/owl#Class</type>
        <own_slot_value>
        <slot_reference>http://www.w3.org/1999/02/22-rdf-syntax-ns#type</slot_reference>
        <value value_type="class">http://www.w3.org/2002/07/owl#Class</value>
        </own_slot_value>
        <superclass>http://www.w3.org/1999/02/22-rdf-syntax-ns#Property</superclass>
        <template_facet_value>
        <slot_reference>http://www.w3.org/2002/07/owl#equivalentProperty</slot_reference>
        <facet_reference>:VALUE-TYPE</facet_reference>
        <value value_type="class">http://www.w3.org/2002/07/owl#DatatypeProperty</value>
        </template_facet_value>
 </class>
……..
…….
……
```

Figure 5(a) Creation of XML file

Generate detailed description and build XML schema by using ALTOVA XMLSpy. XML file is spawn as shown in Figure 5(a), whereas XML Schema is also created in ALTOVA XMLSpy version 2013 as shown in Figure 5(b). Furthermore RDF file is converted from XML schema by using ALTOVA and open the RDF/OWL file in protégé for auxiliary processing [15].

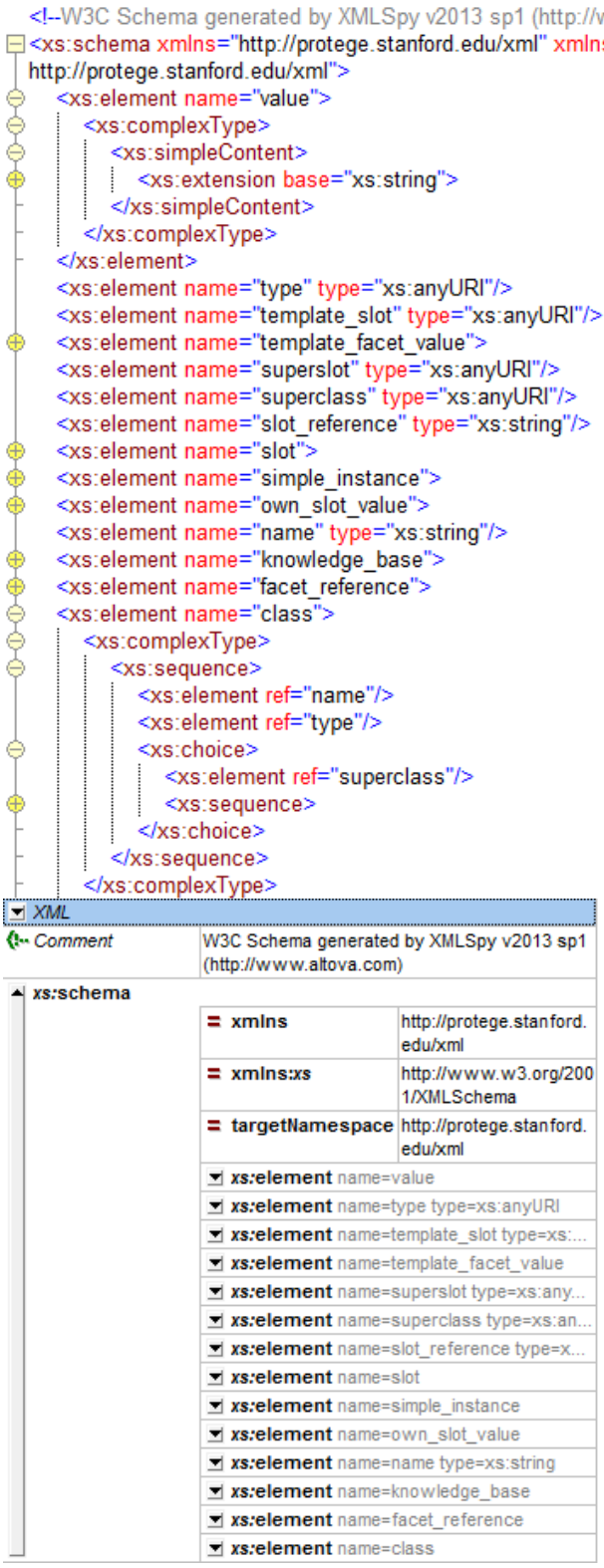


Figure 5(b) Building of XML Schema

Fee onto Description

We enhanced our ontology to multiple campuses of COMSATS, their multiple departments and multiple degrees they have offered. To define the sub and super classes, taxonomic constructor for classes is rdfs:subClassOf.

As the super class is COMSATS with hieracial sub classes are Administration and Campuses, moreover inherited sub classes of administration is admissions with the related sub classes are Prerequist, Programs, Semesters and Subjects [24]. Program class having the BS, MS, Masters and Phd Sub classes, whereas BS class have Degree_Style sub class with associated Dual_Degree program and Local_Degree_Program sub classes. Same as the case with the class inheritance followed in Subject and Campuses classes that are described in Figure 6 and Figure 7.

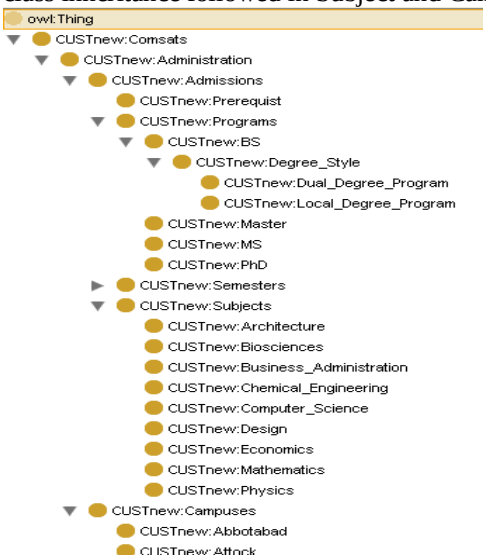


Figure 6: Hierarchical representation of classes and sub classes

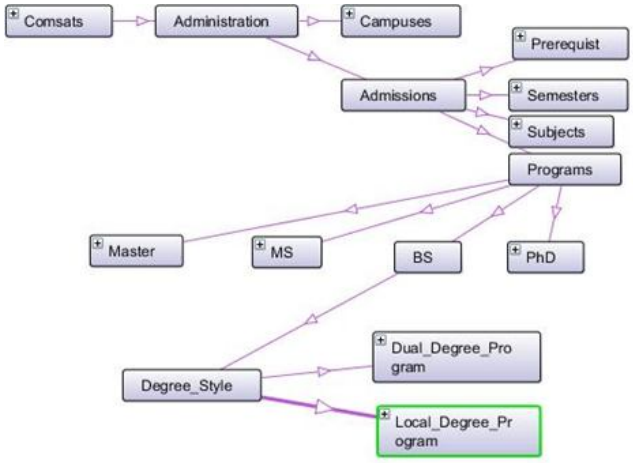


Figure 7: Graphical representation of classes and sub classes

Concept of tuples, triples and variations of restriction property also used for the development of ontology that identifies the individuals their data type and object type properties on their forms. These restriction properties are defined as domain and range associations shown in Figure 8.

```
<owl:DatatypeProperty rdf:ID="hasAdmissionFee">
  <rdf:type rdf:resource="&owl:FunctionalProperty"/>
  <rdfs:domain>
     <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Programs"/>
        <owl:Class rdf:about="#Semesters"/>
        </owl:unionOf>
     </owl:Class>
</rdfs:domain>
      <rdfs:range rdf:resource="&xsd:int"/>
      <rdfs:subPropertyOf rdf:resource="#hasOneTimeCharges"/>
</owl:DatatypeProperty>
       <owl:DatatypeProperty rdf:ID="hasCreditHours">
```

```
<owl:ObjectProperty rdf:ID="hasPrerequist">
    <rdf:type rdf:resource="&owl:FunctionalProperty"/>
    <rdfs:domain>
    <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Programs"/>
        <owl:Class rdf:about="#Semester1"/>
        </owl:unionOf>
    </owl:Class>
    </rdfs:domain>
  <rdfs:range rdf:resource="#Prerequist"/>
</owl:ObjectProperty>
```

Figure 8: Use of Restriction Property

For our sample BS domain, we create these root classes: Programs, Admissions, Administration, and COMSATS.

```
<owl:Class rdf:ID="Comsats">
<owl:Class rdf:ID="Programs">
<owl:Class rdf:ID="Admissions">
<owl:Class rdf:ID="Administration">
<owl:Class rdf:ID="BS">
```

To define the sub and super classes, taxonomic constructor for classes is rdfs:subClassOf. These classes are more accurate, specific as compared to the general classes (super classes), for that we can say if A is a subclass of B then every instance of A is also an instance of B . We can use transitive relationship for that in which we can say if A is a subclass of B and B is a subclass of C then A is a subclass of C. Here in the example shown below "administration" is subclass of "COMSATS"; it is inheriting all the properties of COMSATS. On the other hand Admissions is the subclass of Administration, the resultant after using the transitive property is, Admissions also the subclass of COMSATS and inherits all the properties of COMSATS [13].

```
<owl:Class rdf:ID="Administration">
    <rdfs:subClassOf rdf:resource="#Comsats"/>
</owl:Class>
<owl:Class rdf:ID="Admissions">
    <rdfs:subClassOf rdf:resource="#Administration"/>
</owl:Class>
<owl:Class rdf:ID="Architecture">
    <rdfs:subClassOf rdf:resource="#Subjects"/>
</owl:Class>
<owl:Class rdf:ID="Attock">
    <rdfs:subClassOf rdf:resource="#Campuses"/>
</owl:Class>
```

Class definition has two parts: a name introduction or allusion and number of restrictions, whereas "is used by" is a restriction property that includes a "range" [12][17]. On the other side imposes a restriction on the objects to which the property can be applied is the *domain* property. Obviously class definition restricts the instances of the defined class whereas it belongs to the intersection of the restrictions as identifies in owl:equivalentClass. If we want to deal data of different campuses and want to change versatile placement attributes than we have to use labels. Label is like a comment and contributes nothing to the logical interpretation of ontology having the tag for label is rdfs:label [13]. RDF/OWL file will be created after developing the schema and class with subclass hieratical representation will be formulated in Protégé as shown in Figure 9.

- <rdf:Property rdf:ID="**EquivalentClass**">

  <rdfs:label>EquivalentClass</rdfs:label>
- <rdf:Property rdf:ID="**EquivalentProperty**">

  <rdfs:label>EquivalentProperty</rdfs:label>

  <rdfs:subPropertyOf rdf:resource="&rdfs;subPropertyOf"/>

  </rdf:Property>
- <rdf:Property rdf:ID="**disjointWith**">

  <rdfs:label>disjointWith</rdfs:label>
- <rdf:Property rdf:ID="**sameIndividualAs**">

  <rdfs:label>sameIndividualAs</rdfs:label>
- <rdf:Property rdf:ID="**differentFrom**">

  <rdfs:label>differentFrom</rdfs:label>
- <rdf:Property rdf:ID="**sameAs**">

  <rdfs:label>sameAs</rdfs:label>
- <rdfs:Class rdf:ID="**AllDifferent**">

  <rdfs:label>AllDifferent</rdfs:label>
- <rdf:Property rdf:ID="**distinctMembers**">

  <rdfs:label>distinctMembers</rdfs:label>[2]

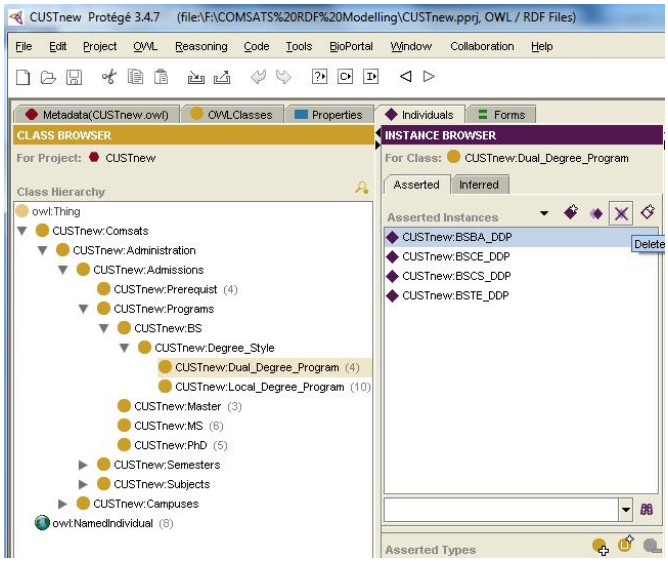


Figure 9 Asserted Individuals

Ontological Graphs

OWL is a vocabulary extension of RDF Semantics. Therefore RDF graph forms an OWL Full ontology, reason to define an RDF graph by OWL having a given approach to the graph by RDF. OWL Full ontologies can be arbitrary RDF content, which is treated in a consistent way with its treatment by RDF [11][14]. OWL suggests more relationships to certain RDF triples. OWL DL and OWL Lite inherits the RDF vocabulary but have some restrictions on its use. That's why RDF documents will generated in OWL Full instead of OWL DL or Lite. OWL graphs of the different sub classes with their individuals as shown in 10; we have displayed Local_Degree_Program sub class from BS class having Degree_Style intermediate class. This Local_Degree_Program contains 10 individuals and can be editing as the scaling conditions, same as with Dual_Degree_Program.

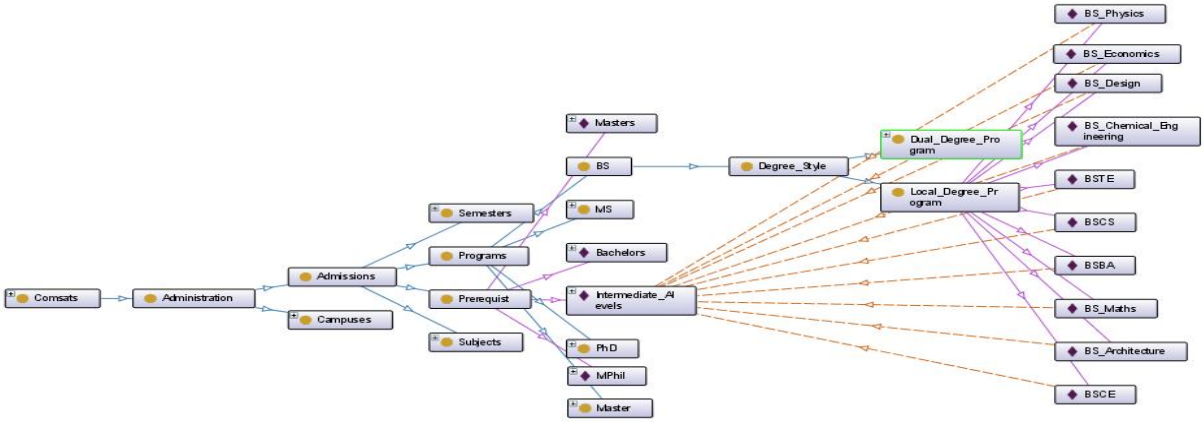


Figure 10 Classes and Sub Classes' Ontology Graphical Representation

Figure 11 shows MS sub class that is connected with Programs class. It has 6 individuals related to different subjects like MS-Maths, MS_Physics, MSEE, MSCS, MSCE and MSBA. Same graphs are builder by PhD and Masters Sub class.

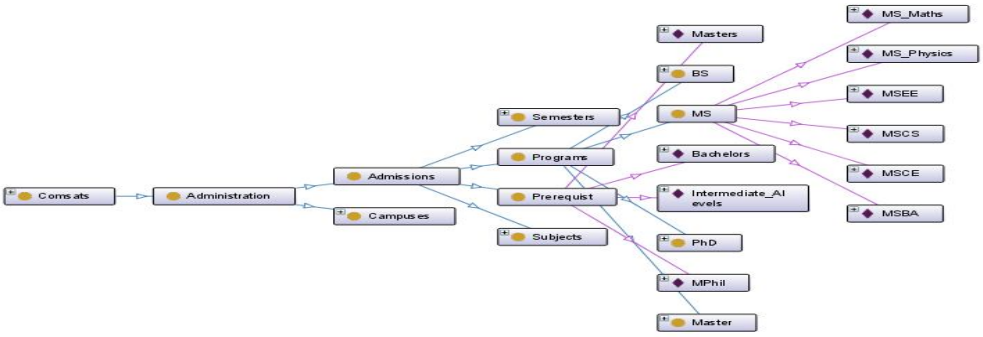
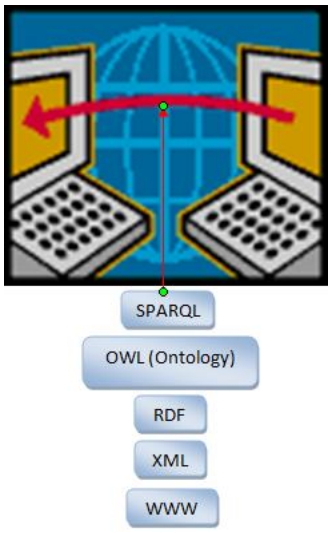


Figure 11 MS Ontology OWL Graph

With affection to that all the classes, sub classes, individuals, properties and forms are inter-dependent, associated and correlated with each other via different properties like object type property, data type property, transitive property, symmetric property, functional property, inverse of property, inverse functional property and synonym property etc [7].

Protocol and Architecture

Query language used for OWL is DAML and upgraded to OWL-QL and XQuery. DAML directly targets to those users whose focuses on answering query on the semantic web. RDF/ RDFS query language (SPARQL) usually query on semantic web style. It emphasis to query schema structure as compared to the data instance as shown in Figure 12 and Figure 13 [5]. That's why we can say SPARQL works as the middleware between RDF schema and the Client interface.



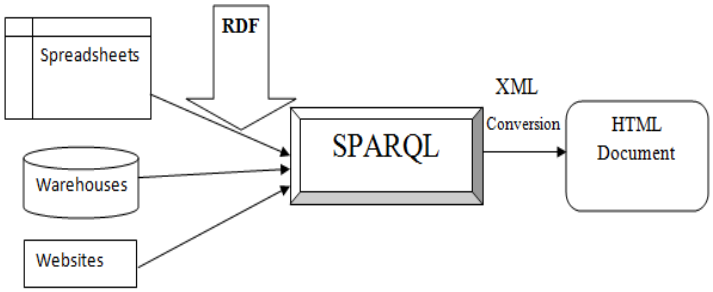


Figure 12 Architecture

Figure 13 SPARQL as middleware

In our ontology, we prefer SPARQL query language. SPARQL is better to perform with the XML, relational data, spreadsheet etc. We broaden the SPARQL by crucial the semantics of SPARQL queries underneath the entailment of RDF, RDFS and OWL. Planned extensions are new versions of the SPARQL 1.1 which is presently being developed as a part of the W3C equivalence process of SPARQL 1.1. SPARQL is not only the query language although it is protocol too. SPARQL queries works between multiple data sources or varies the combinations of these sources. SPARQL protocol inclined queries between server and client program sites and bring results by SPARQL queries via SPARQL protocol [4][8][9]. SPARQL used to utter queries from different data sources as shown in Figure 14. SPARQL shows queries like addition, extraction, sub queries, subclasses, valued expressions, extensible testing etc and making their result very positive as designed in the RDF/OWL graphs. SPARQL Protocol contains SPARQL Query interface which contains query operation. WSDL 2.0 [WSDL 2] is embedded with SPARQL Protocol abstractly as likely HTTP and SOAP that implements its interface, operations, types and faults.

WSDL library or programming language framework will facilitate for the concatenation of WSDL2 with RDF schema query. SPARQL query sending request and receiving response with the help of SPARQL protocol, this message passing is between client and servers. SPARQL protocol illustrates in two ways [22];

1. Conceptual interface autonomous of tangible recognition, execution, and strap to other protocol.
2. Binding with HTTP and SOAP of this interface [17].

The In-out message pattern operation will perform and explains the two ways communication, this prototype consists of two messages. First to indicate by a reference component that is interface message label is 'In', which defines its direction received from some node 'N' [6]. Secondly to indicate by a reference component that is interface message label is 'Out', which defines its direction received from some node 'N'[10].

Queries and their Results

SPARQL Queries on Fee Structure

In our previous paper [23], we used protégé 3.4.7, extension to that "SPARQL query panel" is available to implement using SPARQL by which we can check queries and sort out the results. In the Reasoning tab of Protégé editor, we can select "Open SPARQL query panel" and run our queries. Below is a SPARQL query from our Fee Ontology to search the laboratory fee of BSBA_DDP program. And the output is shown in Figure 14.

```
SELECT ?object
WHERE {http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#BSBA_DDP
http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#hasLaboratoryFee
?object}
```

Results

object
8000

Figure 14 Query and Output

Figure 15 shows a protégé editor having a class browser, containing classes and sub classes, instance browser having the asserted instances with their types, and individual browser having many data type and object type properties [22]. In these properties red labeled properties are the aggregation properties having the result with the combination of more than one property.

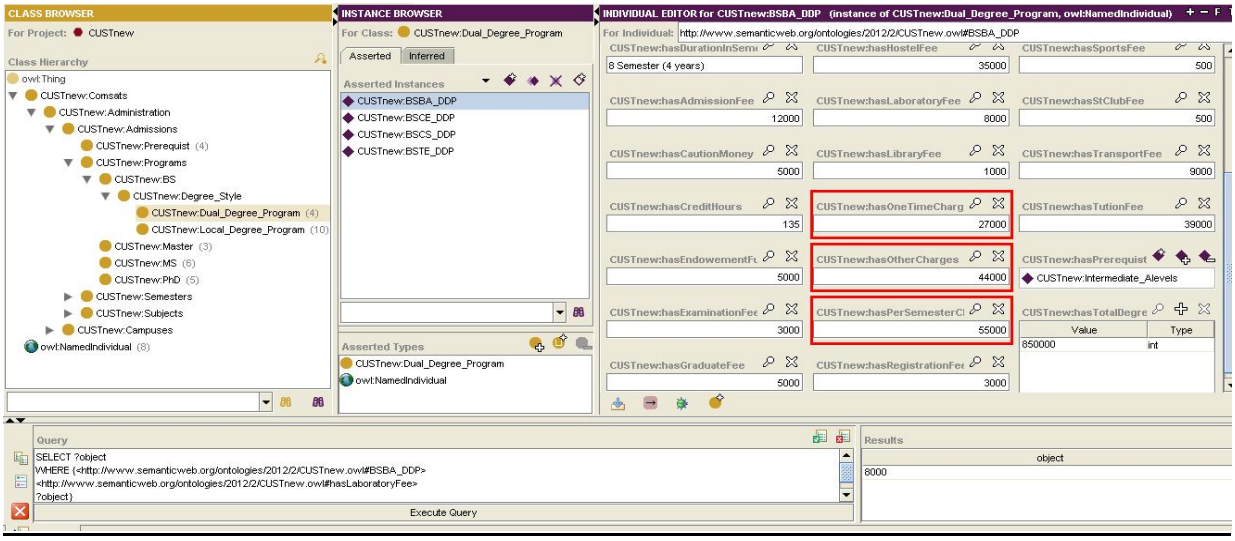


Figure 15 Aggregated Property Values

The bundle of queries for fee ontology regarding object type, data type and annotation properties are given below.

- SELECT ?object
  WHERE {http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#BSBA_DDP
  http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#hasLaboratoryFee
  ?object}
- SELECT ?object
  WHERE {http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#BSBA_DDP
  http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#hasAdmissionFee
  ?object}
- SELECT ?object
  WHERE {http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#BSBA_DDP
  http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#hasCreditHours
  ?object}
- SELECT ?object
  WHERE {http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#BSBA_DDP
  http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#hasDurationInSemester
  ?object}
- SELECT ?object
  WHERE {http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#BSBA_DDP
  http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#hasPerSemesterCharges
  ?object}
- SELECT ?object
  WHERE {http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#BSBA_DDP
  http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#hasOtherCharges
  ?object}
- SELECT ?object
  WHERE {http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#BSBA_DDP
  http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#hasOneTimeCharges
  ?object}
- SELECT ?object
  WHERE {http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#BSBA_DDP
  http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#hasPrerequist
  ?object}
- SELECT ?object
  WHERE {http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#BSBA_DDP
  http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#hasTotalDegreeExpense
  ?object}

  .....

Select Query Form

Prefix identifies the URL related to URI of our fee ontology, as an example rdf, rdfs, owl etc are prefixes, which stapled with the ontology meta-data. Finding the objects from the related sub classes of the associated subjects, this meta-data is fetched by the rdfs language by the help of RDFS prefix [22].

Use of 'SubClassOf' in Query

Below two different queries can be used to find the subject (sub class) from the related object (class). In both the queries we are getting the sub classes as an subject related to every class, whereas classes are taking as objects. In the first query all the subject are retrieving those are related to COMSATS class. Where as in the second query, retrieving exactly those subjects that are related to Admission subclass.

```
PREFIX Comsats:
<http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#Comsats>
SELECT ?subject ?object
WHERE { ?subject rdfs:subClassOf ?object }
```

Results

subject	object
CUSTnew:Subjects	CUSTnew:Admissions
CUSTnew:Wah	CUSTnew:Campuses
CUSTnew:Computer_Science	CUSTnew:Subjects
CUSTnew:Semester3	CUSTnew:Semesters
CUSTnew:Semesters	CUSTnew:Admissions
CUSTnew:Design	CUSTnew:Subjects
CUSTnew:Economics	CUSTnew:Subjects
CUSTnew:Programs	CUSTnew:Admissions
CUSTnew:Lahore	CUSTnew:Campuses
CUSTnew:Local_Degree_Program	CUSTnew:Degree_Style
CUSTnew:Semester4	CUSTnew:Semesters
CUSTnew:Prerequist	CUSTnew:Admissions
CUSTnew:Business_Administration	CUSTnew:Subjects
CUSTnew:Semester1	CUSTnew:Semesters
CUSTnew:Architecture	CUSTnew:Subjects
CUSTnew:Semester6	CUSTnew:Semesters
CUSTnew:Vehari	CUSTnew:Campuses
CUSTnew:Abbotabad	CUSTnew:Campuses
CUSTnew:Sahiwal	CUSTnew:Campuses
CUSTnew:Biosciences	CUSTnew:Subjects
CUSTnew:Chemical_Engineering	CUSTnew:Subjects
CUSTnew:Semester8	CUSTnew:Semesters
CUSTnew:Administration	CUSTnew:Comsats
CUSTnew:MS	CUSTnew:Programs
CUSTnew:Islamabad	CUSTnew:Campuses
CUSTnew:PhD	CUSTnew:Programs
CUSTnew:Dual_Degree_Program	CUSTnew:Degree_Style

Figure 16 'SubClassOf' Query

```
Prefix Comsats:<http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#>
SELECT ?subject ?object
WHERE {?subject rdfs:subClassOf Comsats:Admissions }
```

Results

subject	object
CUSTnew:Subjects	
CUSTnew:Semesters	
CUSTnew:Programs	
CUSTnew:Prerequist	

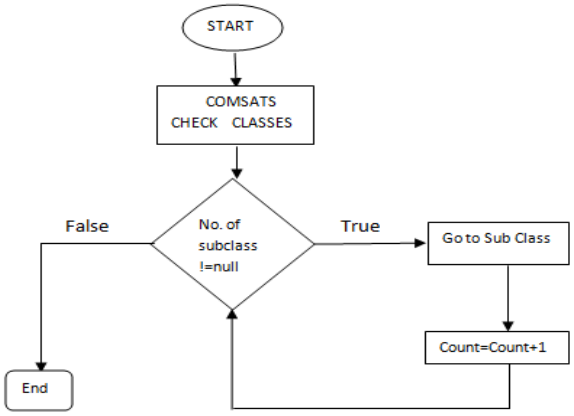


Figure 17 Another 'SubClassOf' Query Style

Use of 'ORDERBY' in Query

Another way to write this query is to use OrderBy object (classes) by sorted it through subject (subclasses) by value. The query returns all the classes and attached subclasses as the result shown in the Figure 18 from ontology.

```
PREFIX Comsats: <
http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#Comsats >
SELECT ?subject ?object
WHERE {?subject rdfs:subClassOf ?object}
ORDER BY ?subject
```

Results

subject	object
CUSTnew:Prerequist	CUSTnew:Admissions
CUSTnew:Programs	CUSTnew:Admissions
CUSTnew:Semesters	CUSTnew:Admissions
CUSTnew:Subjects	CUSTnew:Admissions

Figure 18 'OrderBy' Query

Use of "DISTINCT' in Query

Using DISTINCT in SPARQL query will display the only discrete result and result shown in Fig 15. Distinct is a specialized keyword that can retrieve only those subclasses related by the called object.

```
PREFIX Comsats: <
http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#Comsats >
SELECT DISTINCT ?subject
WHERE {?subiect rdfs:subClassOf ?object}
ORDER BY ?subject
```

subject
CUSTnew:Abbotabad
CUSTnew:Administration
CUSTnew:Admissions
CUSTnew:Architecture
CUSTnew:Attock
CUSTnew:BS
CUSTnew:Biosciences
CUSTnew:Business_Administration
CUSTnew:Campuses
CUSTnew:Chemical_Engineering
CUSTnew:Computer_Science
CUSTnew:Degree_Style
CUSTnew:Design
CUSTnew:Dual_Degree_Program
CUSTnew:Economics
CUSTnew:Islamabad
CUSTnew:Lahore
CUSTnew:Local_Degree_Program
CUSTnew:MS
CUSTnew:Master
CUSTnew:Mathematics
CUSTnew:PhD
CUSTnew:Physics
CUSTnew:Prerequist
CUSTnew:Programs
CUSTnew:Sahiwal
CUSTnew:Semester1
CUSTnew:Semester2

Figure 19 'DISTINCT' Query

Use of 'subPropertyOf' in Query

By the use of 'subPropertyOf' shows the resultant of those objects type properties which are chosen after aggregating multiple property values as shown in Figure 20. This convention utilizes ORDERBY clause. As an example it is displaying the object type properties i.e. hasOneTimeCharges, hasPerSemesterCharges etc is used with various subjects.

```
SELECT  ?object
WHERE {?subject rdfs:subPropertyOf ?object}
ORDER BY ?object
```

Results

object
CUSTnew:hasOneTimeCharges
CUSTnew:hasOneTimeCharges
CUSTnew:hasOneTimeCharges
CUSTnew:hasOneTimeCharges
CUSTnew:hasOtherCharges
CUSTnew:hasOtherCharges
CUSTnew:hasPerSemesterCharges
CUSTnew:hasPerSemesterCharges
CUSTnew:hasPerSemesterCharges
CUSTnew:hasPerSemesterCharges
CUSTnew:hasPerSemesterCharges
CUSTnew:hasPerSemesterCharges
CUSTnew:hasPerSemesterCharges

Figure 20 'SubPropertyOf' Query

Use of 'DISTINCT' in Query with type keyword

The SPARQL keyword '***a***' is a shortcut for the common predicate rdf:type, giving the class of a resource. List all classes in a dataset with '***type***' keyword having the variety of classes [22], their sub classes and relationship properties, these relationships are getting by the type declaration in Figure 21. In this particular query, we are getting the classes and their subclasses with their associated relationships i.e. FunctionalProperty, Datatype Property, Object Property and their allied individuals.

```
PREFIX rdf: <http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#>
SELECT DISTINCT ?type
WHERE {
?s a ?type.
}
```

Results

type
owl:Class
CUSTnew:MS
owl:FunctionalProperty
owl:DatatypeProperty
CUSTnew:PhD
CUSTnew:Local_Degree_Program
CUSTnew:Master
CUSTnew:Prerequist
owl:ObjectProperty
owl:NamedIndividual
CUSTnew:Dual_Degree_Program
owl:Ontology

Figure 21 'DISTINCT' Type Query

Use of 'Domain' in Query

Domain identifies the boundary specification of a particular object, object contains cardinality, properties and their parameters a shown in Figure 22. Whereas in the second query we have defined lower and upper bound of the domains containing the object type, data type and annotation properties as shown in Figure 23.

```
PREFIX rdf: <http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#>
SELECT DISTINCT ?domain
WHERE {
?s a ?domain.
}
```

Figure 22 'Domain' Query

```
PREFIX Comsats: <http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#>
SELECT distinct ?x ?y
WHERE {?x <http://www.w3.org/2000/01/rdf-schema#domain>
?y} order by ?y
```

Results

x	y
CUSTnew:hasCautionMoney	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasRegistrationFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasPerSemesterCharges	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasTutionFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasOtherCharges	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasLaboratoryFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasPrerequist	CUSTnew:Programs or CUSTnew:Semester1
CUSTnew:hasLibraryFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasOneTimeCharges	CUSTnew:Programs or CUSTnew:Semester1
CUSTnew:hasTransportFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasDurationInSemester	CUSTnew:Programs or CUSTnew:Semesters
CUSTnew:hasSportsFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasCreditHours	CUSTnew:Programs or CUSTnew:Semesters
CUSTnew:hasEndowementFund	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasGraduateFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasAdmissionFee	CUSTnew:Programs or CUSTnew:Semesters
CUSTnew:hasExaminationFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasStClubFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasHostelFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasTotalDegreeExpense	CUSTnew:Programs

Figure 23 'Domain' Query with Multiple Variables

Use of 'Range' in Query

Range classifies the versatile results of properties having data types like int, boolean, string, char, float etc as shown in Figure 24.

```
PREFIX Comsats: <http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#>
SELECT distinct ?x ?y
WHERE {?x <http://www.w3.org/2000/01/rdf-schema#range>
?y} order by ?y
```

Results

x	y
CUSTnew:hasPrerequist	CUSTnew:Prerequist
CUSTnew:hasAdmissionFee	int
CUSTnew:hasCautionMoney	int
CUSTnew:hasCreditHours	int
CUSTnew:hasEndowementFund	int
CUSTnew:hasExaminationFee	int
CUSTnew:hasGraduateFee	int
CUSTnew:hasHostelFee	int
CUSTnew:hasLaboratoryFee	int
CUSTnew:hasLibraryFee	int
CUSTnew:hasOneTimeCharges	int
CUSTnew:hasOtherCharges	int
CUSTnew:hasPerSemesterCharges	int
CUSTnew:hasRegistrationFee	int
CUSTnew:hasSportsFee	int
CUSTnew:hasStClubFee	int
CUSTnew:hasTotalDegreeExpense	int
CUSTnew:hasTransportFee	int
CUSTnew:hasTutionFee	int
CUSTnew:hasDurationInSemester	string

Figure 24 'Range' Query

Finding the 'Instances' through Query

The instances of different sub classes are related to each other randomly with multi- variety scopes. For example instance of one class is associated with the instance of another class [22]. Both of the instances bind with each other with the object and subject labels. Figure 25(a)/(b) shows the result of the above queries. The class individuals will

be retrieved according to the property "hasPrerequist". The query would be "What will be prerequisites for the classes and subclasses built in entire ontology?" In the Figure 25(b) "What will be the admission fee of BSCS?", the exact result will be retrieved by running the following query.

```
Prefix Comsats:<http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#>
SELECT ?subject ?object
WHERE { ?subject
<http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#hasPrerequist>
?object}
```

Results

subject	object
CUSTnew:BSBA_DDP	CUSTnew:Intermediate_Alevels
CUSTnew:MS_Maths	CUSTnew:Masters
CUSTnew:MSBA	CUSTnew:Masters
CUSTnew:MS_Physics	CUSTnew:Masters
CUSTnew:BS_Economics	CUSTnew:Intermediate_Alevels
CUSTnew:PhD_EE	CUSTnew:MPhil
CUSTnew:MCS	CUSTnew:Bachelors
CUSTnew:PhD_MS	CUSTnew:MPhil
CUSTnew:PhD_IS	CUSTnew:MPhil
CUSTnew:BS_Physics	CUSTnew:Intermediate_Alevels
CUSTnew:BSTE_DDP	CUSTnew:Intermediate_Alevels
CUSTnew:BSCS	CUSTnew:Intermediate_Alevels
CUSTnew:BSBA	CUSTnew:Intermediate_Alevels
CUSTnew:PhD_CS	CUSTnew:MPhil
CUSTnew:BSCS_DDP	CUSTnew:Intermediate_Alevels
CUSTnew:PhD_Physics	CUSTnew:MPhil
CUSTnew:BS_Maths	CUSTnew:Intermediate_Alevels
CUSTnew:MSEE	CUSTnew:Masters
CUSTnew:BS_Design	CUSTnew:Intermediate_Alevels
CUSTnew:BS_Chemical_Engineering	CUSTnew:Intermediate_Alevels
CUSTnew:BS_Architecture	CUSTnew:Intermediate_Alevels
CUSTnew:BSCE_DDP	CUSTnew:Intermediate_Alevels
CUSTnew:MBA_1.5	CUSTnew:Bachelors
CUSTnew:BSTE	CUSTnew:Intermediate_Alevels
CUSTnew:MBA_3.5	CUSTnew:Bachelors
CUSTnew:MSCE	CUSTnew:Masters
CUSTnew:MSCS	CUSTnew:Masters
CUSTnew:BSCE	CUSTnew:Intermediate_Alevels

Figure 25(a) Show All Instances by SPARQL

```
Prefix Comsats:<http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#>
SELECT ?subject ?object
WHERE { <http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#BSCS>
<http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#hasAdmissionFee>
?object}
```

subject	object
	12000

Figure 25(b) Value of the Property Instance

Use of 'UNIONOF' and 'Members' in Query with 'List' prefix

UNIONOF query works with multi variable as expressed in this code i.e. 'x', 's', 'l', 'm' and identifies the list of the member associates regarding these as shown in Figure 26(a)/(b) . List prefix is declared like Prefix list:http://jena.hpl.hp.com/ARQ/list#. The object type and data type properties will be retrieved by concatenation of different alias, that are members and having their domains.

```
Prefix Comsats: <http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#>
Prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
Prefix owl:<http://www.w3.org/2002/07/owl#>
Prefix list:<http://jena.hpl.hp.com/ARQ/list#>
SELECT distinct ?x ?m
WHERE { ?x rdfs:domain ?s. ?s owl:unionOf ?l. ?l list:member ?m.
}
```

Results

x	m
CUSTnew:hasPerSemesterCharges	CUSTnew:Semesters
CUSTnew:hasPerSemesterCharges	CUSTnew:Programs
CUSTnew:hasLibraryFee	CUSTnew:Semesters
CUSTnew:hasLibraryFee	CUSTnew:Programs
CUSTnew:hasRegistrationFee	CUSTnew:Semesters
CUSTnew:hasRegistrationFee	CUSTnew:Programs
CUSTnew:hasDurationInSemester	CUSTnew:Programs
CUSTnew:hasDurationInSemester	CUSTnew:Semesters
CUSTnew:hasTransportFee	CUSTnew:Semesters
CUSTnew:hasTransportFee	CUSTnew:Programs
CUSTnew:hasCreditHours	CUSTnew:Programs
CUSTnew:hasCreditHours	CUSTnew:Semesters
CUSTnew:hasAdmissionFee	CUSTnew:Programs
CUSTnew:hasAdmissionFee	CUSTnew:Semesters
CUSTnew:hasEndowementFund	CUSTnew:Semesters
CUSTnew:hasEndowementFund	CUSTnew:Programs
CUSTnew:hasHostelFee	CUSTnew:Semesters
CUSTnew:hasHostelFee	CUSTnew:Programs
CUSTnew:hasTutionFee	CUSTnew:Semesters
CUSTnew:hasTutionFee	CUSTnew:Programs
CUSTnew:hasPrerequist	CUSTnew:Programs
CUSTnew:hasPrerequist	CUSTnew:Semester1
CUSTnew:hasStClubFee	CUSTnew:Semesters
CUSTnew:hasStClubFee	CUSTnew:Programs
CUSTnew:hasSportsFee	CUSTnew:Semesters
CUSTnew:hasSportsFee	CUSTnew:Programs
CUSTnew:hasLaboratoryFee	CUSTnew:Semesters
CUSTnew:hasLaboratoryFee	CUSTnew:Programs

Figure 26(a) 'UNIONOF' and 'Member' Query

```
PREFIX COMSATS: <http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
Prefix owl:<http://www.w3.org/2002/07/owl#>
Prefix list:<http://jena.hpl.hp.com/ARQ/list#>
SELECT distinct ?x ?y
WHERE {?x <http://www.w3.org/2000/01/rdf-schema#domain> ?y.
?y <http://www.w3.org/2002/07/owl#unionOf> ?z }
order by ?y
```

Results

x	y
CUSTnew:hasCautionMoney	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasRegistrationFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasPerSemesterCharges	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasTutionFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasOtherCharges	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasLaboratoryFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasPrerequist	CUSTnew:Programs or CUSTnew:Semester1
CUSTnew:hasLibraryFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasOneTimeCharges	CUSTnew:Programs or CUSTnew:Semester1
CUSTnew:hasTransportFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasDurationInSemester	CUSTnew:Programs or CUSTnew:Semesters
CUSTnew:hasSportsFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasCreditHours	CUSTnew:Programs or CUSTnew:Semesters
CUSTnew:hasEndowementFund	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasGraduateFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasAdmissionFee	CUSTnew:Programs or CUSTnew:Semesters
CUSTnew:hasExaminationFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasStClubFee	CUSTnew:Semesters or CUSTnew:Programs
CUSTnew:hasHostelFee	CUSTnew:Semesters or CUSTnew:Programs

Figure 26(b) 'UNIONOF' Query with List Prefix

Use of 'FILTER', 'OPTIONAL' and 'BOUND' in Query

Query Language -- SPARQL endow with OPTIONAL clause and the clause allows using information during the graph pattern designing as the availability, on the contrast it will not eliminate the solutions. The basic definition of graph pattern is to find out that particular set of correlation subsist in it or not. The relationship between subject, predicate and object is there as it is graph pattern constraints [22]. To work positively SPARQL offers FILTER operations to bind the extra constraints. According to the open scenario when we want to find missing data, OPTIONAL clause will handle it expertly. In this query, we concatenate OPTIONAL clause with BOUND keyword because the merging result do not declare particular relationship [16].

```
PREFIX COMSATS: <http://www.semanticweb.org/ontologies/2012/2/CUSTnew.owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
Prefix owl:<http://www.w3.org/2002/07/owl#>
Prefix list:<http://jena.hpl.hp.com/ARQ/list#>
SELECT DISTINCT *
WHERE {
 ?x Program:BS ?y .
 {
 OPTIONAL  {
  ?y a ?type .
 } FILTER (! BOUND (?type ))
 } UNION {
 ?y a Program:BS type .
 }
}
```

DISCUSSION

Data gathering from different data sources by a finest extraction filter and that data sources having the data in versatile formats and conversion of that data in required format. Enhancing the ontology to different campuses of COMSATS as well as merging of fee structure ontologies of different universities. Therefore we can resolve heterogeneity between various university fee structures. In the upcoming work we can deploy it on client server architecture where it binds server with SPARQL protocol. In the future, we can add advanced queries to perform semantic search.

Conclusion

Keeping detailed study of the fee structure of COMSATS University in mind, ontology for university fee structure has been developed. Moreover, classes, properties (object, data, and annotations), domain, range, functional, inverse, non-functional, transitive, symmetric etc have been carefully identified. For data arrangement, taxonomy and management all the major technologies that are used for the solution of web semantics have been described and reached a final solution that RDF/OWL is a main feature for the solution of semantics web. It stores the data in subject, object and predicate form by using RDF/OWL file with the combination of RDFS that adds more construct to organize classes and their properties in various formats. OWL gives a better approach to SPARQL and has greater flexibility to enhance the schema regarding the RDF graph. We used SPAQL, semantic query language for the Web, to query the ontology. By using SPARQL query, fee ontology is used to answer the queries of students i.e. what are full charges for BSCS_DDP program or what are one time charges for PhD_EE program. Semantic modeling of university fee structure helps the students in identifying the fee structure of a particular program and supports the university administration without hiring new people for this task. We accumulate that syntax of SPARQL query language is similar to SQL query language. Only major difference is SQL captures the data from tables but SPARQL seizes data from graph. Fetching procedure from graph is done by pattern matching method. OWL builds the classes and associates their properties as RDF and RDFS and contains less constructs than OWL. OWL gains the benefit that SPARQL works with it because it has more constructs and SPARQL have a capability to join and extend more graphs and more results. User interface transfer the query to the RDF document sources via Jena APIs using SPARQL. Interface to query the ontology is developed in Java that displays the user questions and

also publishes data on the web and provides an interface that accepts query and return results in standard XML layout.

Acknowledgment

Though only our name appears as the authors of this paper, a great person has contributed to its production. We own our gratitude to him, who has made this paper possible and because of whom our experience has been one that we will cherish forever. Our deepest gratitude is to our guiders, Dr. Tabbasum Naz and other domain researchers. We have been amazingly fortunate to have excellent researchers who gave us the freedom to explore on our own and at the same time the guidance to recover when our steps faltered.

REFERENCES

[1] Liyang Yu. (2011). In: *A developer's guide to the semantic web,*Heidelberg; New York: Springer.
[2]Sven Groppe. (2011). In: *Data management and query processing in semantic web databases*, Berlin ; Heidelberg ; New York : Springer.
[3] Bob DuChame. (2011). In: *Learning Sparql,*Oreilly & Associates
[4] John Hebeler, Mattew Fisher, Ryan Blace, Andrew Perez-Lopez, Mike Dean. (2009). In: Semantic Web programming.
[5] Toby Segaran; Colin Evans; Jamie Taylor. (2009). In: *Programming the Semantics Web*, O' Reilly Media, Inc.
[6] Herbert Schildt . (2002). In: *Java 2 : The Complete Reference, Fifth Edition*, Mc Graw-Hill/ OSBORNE, Brandon A. Nordin.
[7] Grigoris Antoniou and Frank van Harmelen. (2004). In: *A Semantic Web Primer*, MIT Press Cambridge, Massachusetts London, England.
[8] A. Gómez-Pérez, M. Fernández-López, Oscar Corcho. (2004). In: *Ontological Engineering: With examples from the areas of Knowledge Management, E-Commerce and the Semantic Web*, Springer
[9] Dean Allemang, Jim Hendler , (2011). In: *Semantic Web for the working ontologist : Effective Modeling in RDFS and OWL*, Denise E. M. Penrose Morgan , Kaufmann in Elsevier
[10] Li Ding, Pranam Kolari, Zhongli Ding and Sasikanth Avancha. (2007). In: *Using Ontologist in the Semantic Web: A Survey*, Maryland Baltimore County.

[11] Berners-Lee, T., J. Hendler, and O. Lassila. (2001). In: Semantic web, a new form of web content that is meaningful to computers will unleash a revolution of new possibilities,
[12] Michael C. Daconta Leo J. Obrst Kevin T. Smith .(2009). In: *The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management*, Wiley Publishing, Inc.
[13] James Bailey, Franc, Ois Bry, Tim Furche, Sebastian Schaffert. (2008). In: *Semantic Web Query Languages*
[14] "OWL Web Ontology Language Guide", W3C Recommendation 10 February (2004).
[15] Kendall Grant Clark, Lee Feigenbaum, Elias Torres. (2008). In: *SPARQL Protocol for RDF-- W3C Recommendation 15 January 2008.*
[16]"The future of the Web is Semantic: *Ontologies form the backbone of a whole new way to understand online data "* by developer works IBM
[17] "Semantic Web Architecture (Ontologies and Semantic web)", http://www.obitko.com/tutorials/ontologies-semantic-web/semantic-web-architecture.html
[18] Paul Doran, Valentina Tamma, Luigi Iannone, CIKM(2007) 16th ACM Conference ."Ontology Module Extraction for Ontology Reuse: An Ontology Engineering Perspective", page 61-70
[19] Michael Grobe, SIGUCCS (2009). "RDF, Jena, SparQL and the Semantic Web", page 131-138
[20] Hafiz Hammad Rubbani. (2007). In: *Semantic Web Solutions: Thesis report in IT University of Copenhagen*
[21] Jonathan Hayes. (2004). In: *A Graph Model for RDF: Diploma Thesis*, Technische Universit¨at Darmstadt Universidad de Chile.
[22] Sana Rizwan, 2012.Semantic Modeling of COMSATS University Fee Structure, M.S. thesis, COMSATS Institute of Information Technology, Lahore Pakistan.
[23] J.S.Mirza, Tabbsum Naz, Sana Rizwan, Imran Latif . (2011). In: *Semantic Modeling of a University Fee Structure*, ICAMS 2011.
[24]Natalya F. Noy and Deborah L. McGuinness. (2009). In: *Ontology Development 101: A Guide to Creating Your First Ontology*.