A group housing society ontology in Swoop 2.3 Beta 4 and Protege 3.4.4

Sanjay Kumar Malik*

University School of Information Technology,
GGS Indraprastha University,
Sector 16C, Dwarka,
New Delhi-110075, India
E-mail: sdmalik@hotmail.com
*Corresponding author

S.A.M. Rizvi

Department of Computer Science,
Jamia Millia Islamia University,
Jamia Nagar, New Delhi, India
E-mail: Samsam_rizvi@yahoo.com

Abstract: Ontology creation using various editors like Protégé and SWOOP plays a significant role in Semantic Web. In this paper, we illustrate the creation of a housing society ontology viz, 'Jawaharlal CGHS ontology' using Swoop and Protégé, the most commonly used ontology editors. We are using Swoop 2.3 Beta 4 editor where super class and sub class hierarchy and instances have been presented with the query retrieval process using 'Fly the MotherShip' option in advanced tab and show the 'general concept inclusion axiom', 'class expression table' for relations in classes and subclasses. Then, the same ontology is created using Protégé 3.4.4 and finally illustrates using the OWL code to obtain ontology class hierarchy (subject/object relationship) in Protege by executing a SPARQL query.

Keywords: Semantic Web; ontology; Swoop; Protégé; query retrieval; class hierarchy; SPARQL.

Reference to this paper should be made as follows: Malik, S.K. and Rizvi, S.A.M. (2014) 'A group housing society ontology in Swoop 2.3 Beta 4 and Protege 3.4.4', *Int. J. Autonomic Computing*, Vol. 2, No. 1, pp.21–38.

Biographical notes: Sanjay Kumar Malik has more than 15 years of experience in academics and industry in India and abroad (Dubai and USA) and is presently working as an Assistant Professor in the University School of IT, GGS Indraprastha University, Delhi. He holds an MCA and MTech (IT), presently pursuing his PhD from Indraprastha University, published several research papers in national/international journals/conferences of repute, and attended conferences in USA.

S.A.M. Rizvi holds a PhD in Computer Science and Engineering and is presently working as an Associate Professor in the Department of Computer Science, Jamia Millia Islamia, New Delhi, India, having more than 26 years of experience in India and abroad. He is an expert in software engineering, and has published numerous papers in the field of software engineering, MIS, mathematical modelling, bioinformatics and web-based applications.

1 Introduction

The new generation web known as Semantic Web refers to an intelligent web where search is performed meaningfully and efficiently and aims at automated processing on web through machine-understandable metadata where semantic web agents could utilise metadata and ontologies to carry out its tasks (Alesso and Smith, 2006). Berners-Lee defines the Semantic Web as "a web of data that can be processed directly and indirectly by machines". The Semantic Web is an extension of the current web in which information is given a well-defined meaning, better enabling computers and people to work in cooperation (Berners-Lee et al., 2001). "The 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 using it in various applications (Lambrix, 2005).

To make Semantic Web happen, ontology is an important technology which is one of the key components of Semantic Web layered architecture and which is an effective way to represent knowledge and refers to formal specification of a shared conceptualisation (Gruber, 1995). It refers to understanding the concepts of a domain as well as helps the machine to interpret the definitions of concepts in the domains and also the relations between them (Chandrasekran et al., 1999). Ontology usage is the heart of Semantic Web which involves domain conceptualisations consisting of a set of concepts, their definitions, properties and the relationships between them. According to Gruber (1995) "An ontology is an explicit specification of a conceptualization". Ontologies are needed because of various reasons such as (Noy and McGuinness, 2000):

a sharing common understanding of structure of information among people or software agents

b analysing a domain knowledge and separating it from operational knowledge

c enabling reuse of domain knowledge

d making domain assumptions explicit.

Ontology construction (design and development) is one of most significant issue which involves a series of well-defined steps like: check anomalies, define/determine scope/instances/facets/relations, build taxanomy, enumerate terms and concept exploration (Antoniou and Harmelen, 2008). Among the several ontology issues, ontology creation is the most critical one which has been addressed here. There are various ontology editors/tools available for the design and development of an ontology like Protégé (http://protege.stanford.edu/), Swoop (http://www.mindswap.org/2004/SWOOP/), etc. Protégé is widely used open source Java based tool and Swoop is an ontology editor designed specifically for rapid and easy browsing and development of web ontologies. It creates primary identifiers for ontologies, classes, properties, and individuals to support hyper textual navigation through and between ontologies.

This paper focuses on Swoop and Protégé usage for ontology development. We illustrate the creation of 'Jawaharlal Housing Society' ontology using Swoop 2.3 Beta 4 editor and Protégé 3.4.4. First, using Swoop presents super class and sub class hierarchy of Jawaharlal CGHS ontology, creating LHS and RHS axiom in GCI, displaying the class expression using 'Show Class Expression Table' sub-option of 'Advanced' tab and 'Management Committee' class expression, cropcricle obtained using 'Fly The MotherShip' option for searching 'Member', editing and annotating OWL entities, code snippets, ontology statistics with observations. Second, using Protégé shows ontology class hierarchy and executing SPARQL [Simple Protocol and RDF (Resource Description Framework) Query Language] to obtain output showing subject and object relationship (Malik and Rizvi, 2012).

2 Swoop and Protégé ontology editors

Although, Protégé is the most commonly used ontology editor/tool but Swoop is also used as and when required.

Table 1 Brief comparison of Protégé and Swoop

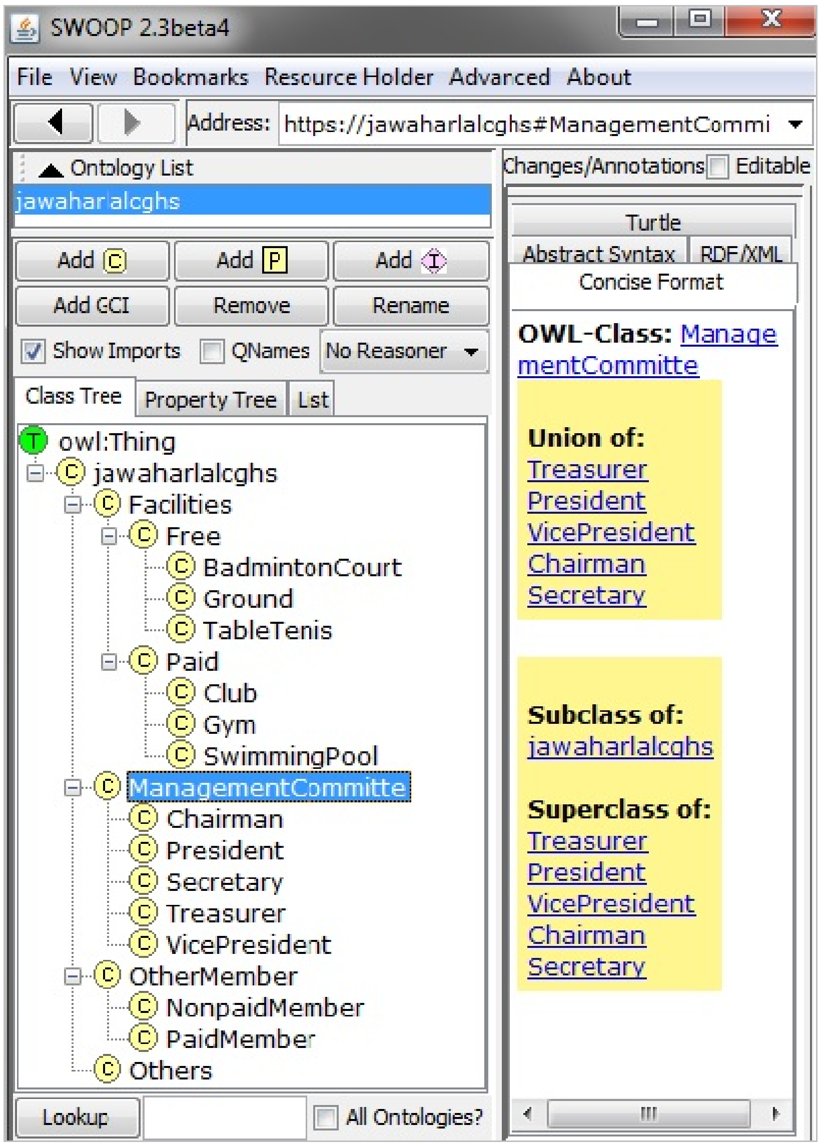
Protégé: ontology editor (http://protege.stanford.edu/)	Swoop: ontology editor (http://mindswap.org/2004/swoop/)
Developed by Stanford Center for Biomedical Research at the Stanford University	Developed by MIND Lab at University of Maryland, College Park
Functionalities like browsing, editing, checking of existing ontologies , creating new ontologies	Functionalities: browsing, editing, creating new ontologies, multiple ontology support
Free, open-source platform which can be customised or extended	Open source, (user friendly) browser like, inline editing with HTML renderer.
It provides a suite of tools for constructing domain models and knowledge-based applications with ontologies. Support the creation, visualisation, and manipulation of ontologies in various representation formats	An ontology browser and editor, designed specifically for use with OWL and directly supporting the use of web-based 'cultural metaphors' – that is, based on the way people are used to interact with documents and data in current web applications
Context words: Java-based, extensible, plug-and-play environment, frames editor, OWL editor, debug, check, good for tests, graphical visualisations, KB framework, formats: RDF(S), OWL, XML schema; open source.	Context words: hypermedia-based, web browser (look and feel) - hyperlink based navigation, OWL ontology editor, inline editing, visualisation of the class hierarchy, multiple ontology support (browsing, mapping, comparison), collaborative annotation support
Training required	Requires no special training

3 Ontology creation using Swoop

Consider a housing society (Jawaharlal CGHS) with various classes and subclasses whose ontology is to be created using Swoop as shown in Figure 1.

By clicking on 'New Ontology' option in 'File' tab of menu bar of Swoop, we start creating the new ontology which is presented as below showing a few classes and subclasses. Figure 1 shows the sub class and super class hierarchy of Jawaharlal CGHS ontology. When we select 'Concise Format' of 'Management Committee' class, we notice that 'Management Committee' is the sub-class of 'Jawaharlal CGHS' and super-class of 'Treasurer', 'President', 'Vice President', 'Chairman', and 'Secretary' as shown in Figure 1.

Figure 1 Super class and Sub class hierarchy of Jawaharlal CGHS ontology (see online version for colours)



```
owl:Thing
  jawaharlalcghs
    Facilities
      Free
        BadmintonCourt
        Ground
        TableTenis
      Paid
        Club
        Gym
        SwimmingPool
    ManagementCommitte
      Chairman
      President
      Secretary
      Treasurer
      VicePresident
    OtherMember
      NonpaidMember
      PaidMember
    Others
```

OWL-Class: ManagementCommitte

Union of:
Treasurer
President
VicePresident
Chairman
Secretary

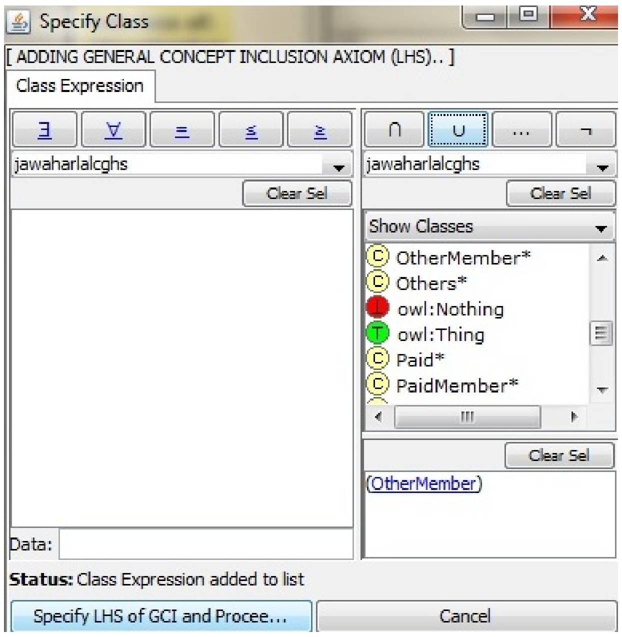
Subclass of:
jawaharlalcghs

Superclass of:
Treasurer
President
VicePresident
Chairman
Secretary

3.1 *Adding general concept inclusion axiom*

Ontology is the set of class, property and individual names that are used in axioms. By using this option, we establish the semantic between classes, sub-classes, properties, and instances in the ontology or between ontologies. This can be done by clicking the ‘Add GCI’ option in Figure 1. In Figure 2, we can select the classes, sub-classes, data-types, and individual as left hand side (LHS) and then select the same for right hand side (RHS). It will generate the class expression for selected classes as shown below:

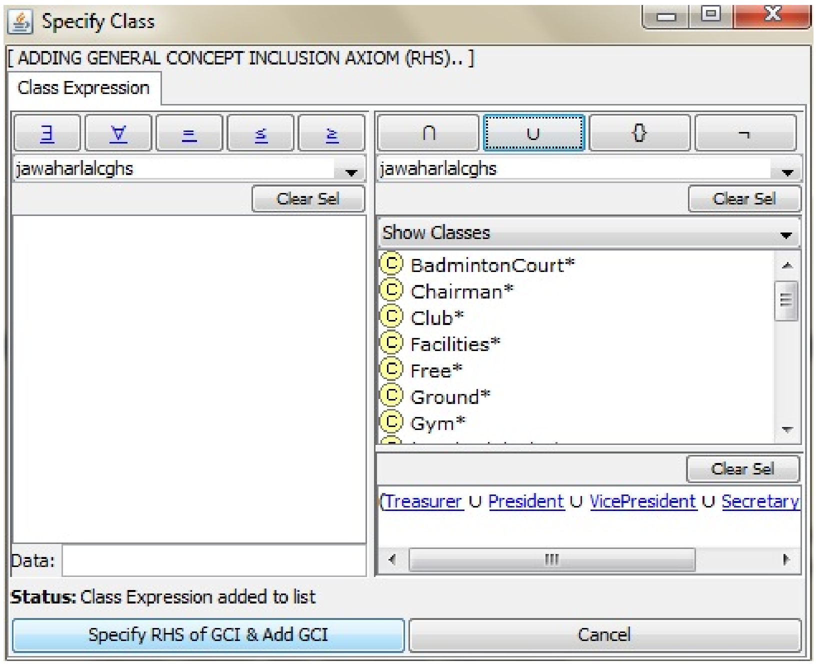
Figure 2 Creating LHS axiom in GCI (see online version for colours)



Status: Class Expression added to list

In Figure 2 'Other Member' class is selected as LHS by clicking on 'U' (union symbol) and after clicking on the 'Specify LHS of GCI and Proceed' then following Figure 3 will be obtained.

Figure 3 Creating RHS axiom in GCl (see online version for colours)



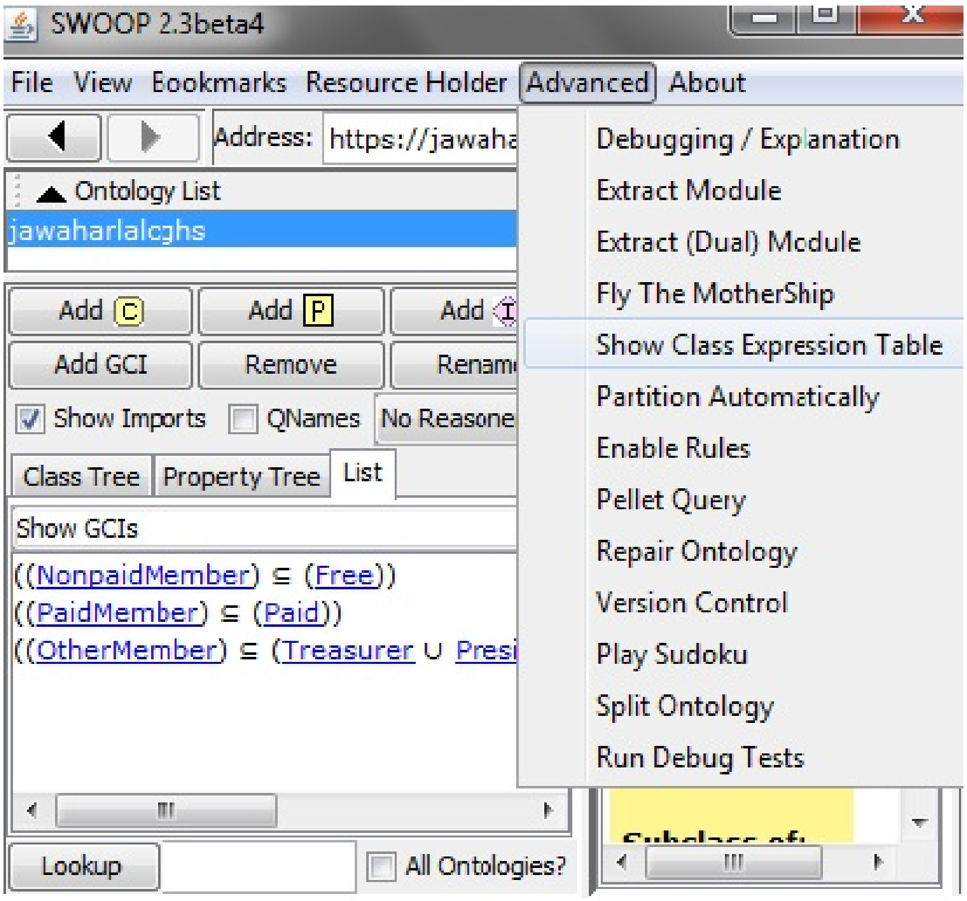
Status: Class Expression added to list

In Figure 3, ‘Treasurer’, ‘President’, ‘Vice-President’, ‘Chairman’, ‘Secretary’ class is selected and union of all the classes are created by clicking on ‘U’. Now, by clicking on ‘Specify RHS of GCI & Add GCI’, class expression is generated. It shows that all union of ‘Treasurer’, ‘President’, ‘Vice-President’, ‘Chairman’, ‘Secretary’ classes are the sub set of ‘Other Member’ class. Similarly, other options can be used and displayed in Table 1.

3.2 *Class expression table*

To display ‘Class Expression Table’ using the ‘Show Class Expression Table’ option of ‘Advanced’ tab as shown in Figure 4.

Figure 4 Displaying the class expression using ‘Show Class Expression Table’ sub-option of ‘Advanced’ tab (see online version for colours)



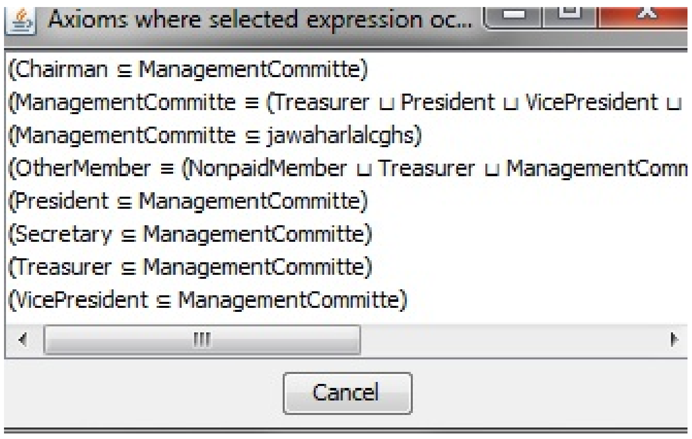
((NonpaidMember) ⊆ (Free))
((PaidMember) ⊆ (Paid))
((OtherMember) ⊆ (Treasurer ∪ Presi

When we select this option, it will show the table with the column class expression, occurrence, depth, and score. In the row, all the classes will be listed with their respective values. Now, if we select one of the classes e.g., ‘Management Committee’ class, then it will show the axioms generated by GCI with respect to the class ‘Management Committee’, as shown in Figure 5.

Table 2 Class expression table

Constructor	*DL Syntax*	*Example*	*FOL syntax*
intersectionOf	$C_1 \cap \ldots \cap C_n$	Human ∩ male	$C_1(x) \wedge \ldots \wedge C_n(x)$
unionOf	$C_1 \cup \ldots \cup C_n$	Doctor ∪ lawyer	$C_1(x) \vee \ldots \vee C_n(x)$
complementOf	$\neg C$	¬Male	$\neg C(x)$
one of	$\{x_1\} \cup \ldots \cup \{x_n\}$	{john} ∪ {mary}	$x = x_1 \vee \ldots \vee x = x_n$
allValuesFrom	$\forall P.C$	∀hasChild.Doctor	$\forall y.P(x,y) \rightarrow C(y)$
someValuesFrom	$\exists P.C$	∃hasChild.Lawyer	$\exists y.P(x,y) \wedge C(y)$
maxCardinality	$\leqslant nP$	1⩽hasChild	$\exists^{n} y.P(x,y)$
minCardinality	$\geqslant nP$	2⩾hasChild	$\exists^{n} y.P(x,y)$

Figure 5 ‘Management Committee’ class expression (see online version for colours)



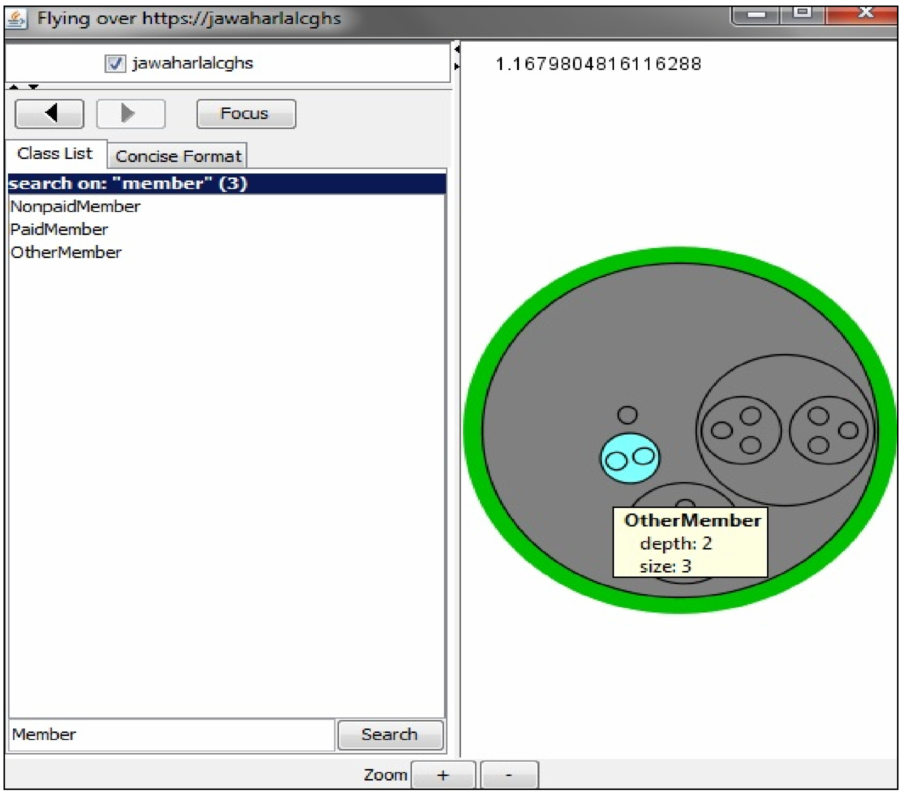
(Chairman ⊑ ManagementCommitte)
(ManagementCommitte ≡ (Treasurer ⊔ President ⊔ VicePresident ⊔
(ManagementCommitte ⊑ jawaharlalcghs)
(OtherMember ≡ (NonpaidMember ⊔ Treasurer ⊔ ManagementComn
(President ⊑ ManagementCommitte)
(Secretary ⊑ ManagementCommitte)
(Treasurer ⊑ ManagementCommitte)
(VicePresident ⊑ ManagementCommitte)

In this Figure 5, shows that class ‘management committee’ is equivalent to the union of ‘Treasurer’, ‘President’, ‘Vice-President’, ‘chairman’, ‘Secretary’ classes. Class ‘chairman’, ‘Secretary’, ‘President’, ‘Vice-President’, Treasurer’ are sub-class of class ‘Management Committee’ shown individually. It is also shown that ‘Management Committee’ class also comes under the ‘Other Member’ class but it is not the sub-class of Class ‘Other Member’.

3.3 Fly The Mothership

Now we consider the query retrieval process to show that how we run the query using the ‘Fly The MotherShip’ option of ‘Advanced’ tab as shown in Figure 6.

Figure 6 Cropcricle obtained using ‘Fly The MotherShip’ option for searching ‘member’ (see online version for colours)



1.1679804816116288

search on: "member" (3)
NonpaidMember
PaidMember
OtherMember

OtherMember
depth: 2
size: 3

Member

When we select the ‘Fly The MotherShip’ for searching ‘member’, then three result is obtained depicted in following Cropcircle diagram. It also shows Semantic ‘Paid Member’, and ‘Nonpaid Member’ as sub classes of ‘Other Member’ class.

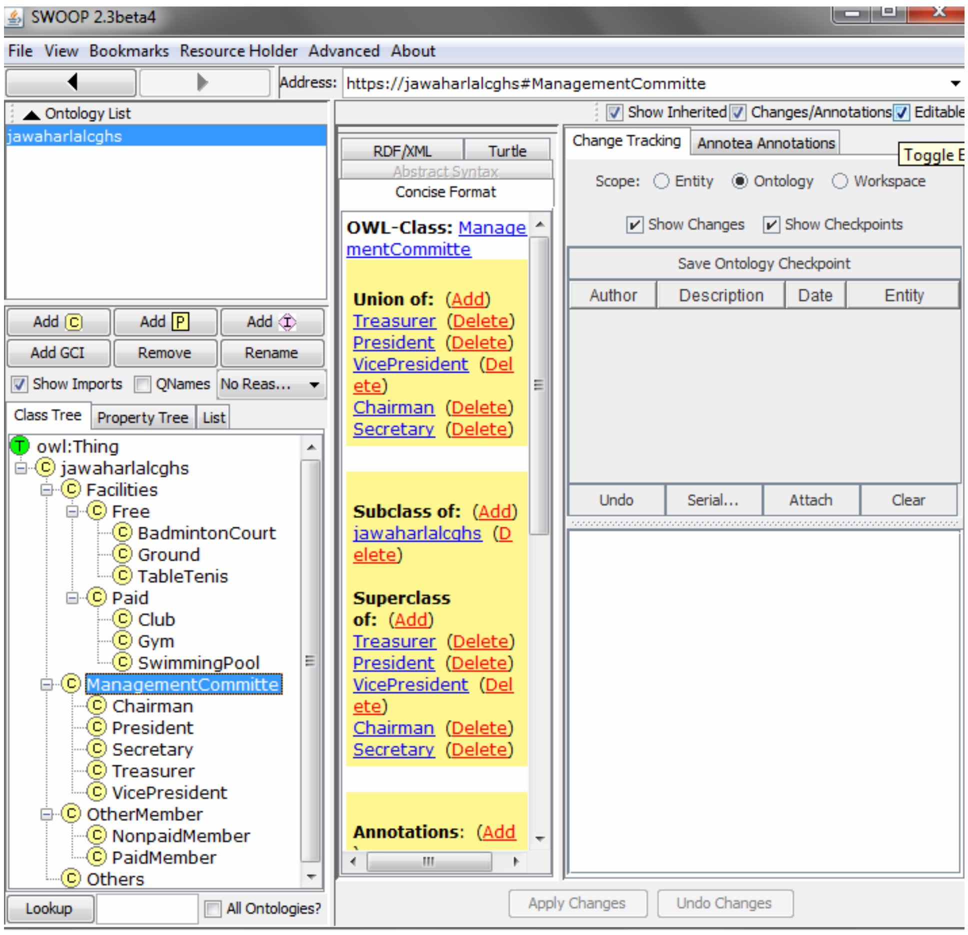
3.4 *Partition automatically*

When we click ‘Partition Automatically’ which is a sub-option of ‘Advanced’ tab, Figure 4, then ask for to ‘Continue?’, after clicking ‘yes’ it will transform an OWL ontology to into an e-connection. All super classes will be separated into respective ontology.

3.5 *Editing*

Editing OWL entities in a multiple ontology often difficult task but Swoop is an easily editable editor. It can be done by selecting the check box ‘Editable’. This option might give negative output and also too much editing may create ambiguity in ontology.

Figure 7 Editing OWL entities in Swoop (see online version for colours)



Address: https://jawaharlalcghs#ManagementCommitte

[✓] Show Inherited [✓] Changes/Annotations [✓] Editable

Scope: ( ) Entity [✓ Ontology] ( ) Workspace

[✓] Show Changes [✓] Show Checkpoints

[✓] Show Imports [ ] QNames

[ ] All Ontologies?

jawaharlalcghs

- owl:Thing
  - jawaharlalcghs
    - Facilities
      - Free
        - BadmintonCourt
        - Ground
        - TableTenis
      - Paid
        - Club
        - Gym
        - SwimmingPool
    - ManagementCommitte
      - Chairman
      - President
      - Secretary
      - Treasurer
      - VicePresident
    - OtherMember
      - NonpaidMember
      - PaidMember
    - Others

OWL-Class: ManagementCommitte

Union of: (Add)
Treasurer (Delete)
President (Delete)
VicePresident (Delete)
Chairman (Delete)
Secretary (Delete)

Subclass of: (Add)
jawaharlalcghs (Delete)

Superclass of: (Add)
Treasurer (Delete)
President (Delete)
VicePresident (Delete)
Chairman (Delete)
Secretary (Delete)

Annotations: (Add

Note: Concise format view

There are also some parameters which are debatable like:

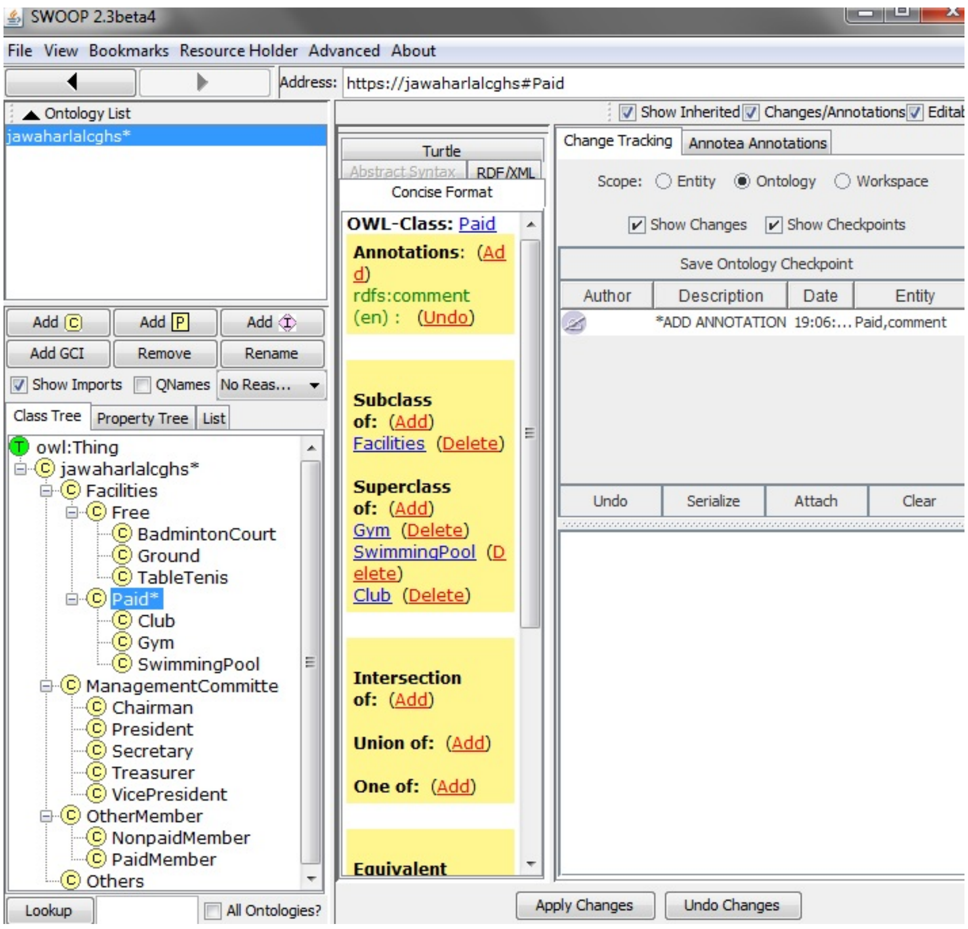
- scope of a change (should editing be restricted to the local ontology alone or can the imported ontology be directly or indirectly altered as well)
- the types of changes allowed (i.e., atomic vs. composite change strategies)
- at which level changes are made (in the abstract representations or directly in the source code)
- how to display the effects of changes before they are committed (direct vs. inferred effects on related entity definitions).
- The degree of rollback possible (for how long changes can be ‘undo’).

3.6 Annotations

When browsing or building ontologies that live on the web, it is almost as important to have information about the ontologies as it is to have the ontologies themselves. OWL allows for the associating of variously structured information with its core entities, e.g., classes and properties.

Swoop supports the editing and display of textual or HTML formatted comments, and of photos and other multimedia (both via HTML and independently) as part of ontologies. Since OWL ontologies can reference and import other ontologies, one can separate annotations about ontologies from the core ontologies themselves. The Annotea framework takes this idea and provides both specific RDF based, extensible annotation vocabulary, and a protocol for publishing and finding out-of-band annotations. Swoop uses the Annotea framework (Kahan et al., 2001) as the basis of collaborative ontology development.

Figure 8 Annotating OWL entities – ‘Paid’ of class (see online version for colours)



Address: https://jawaharlalcghs#Paid

Scope: ○ Entity ● Ontology ○ Workspace

☑ Show Changes ☑ Show Checkpoints

Author	Description	Date	Entity
	*ADD ANNOTATION	19:06:...	Paid,comment

OWL-Class: Paid

Annotations: (Add)
rdfs:comment
(en) : (Undo)

Subclass of: (Add)
Facilities (Delete)

Superclass of: (Add)
Gym (Delete)
SwimmingPool (Delete)
Club (Delete)

Intersection of: (Add)

Union of: (Add)

One of: (Add)

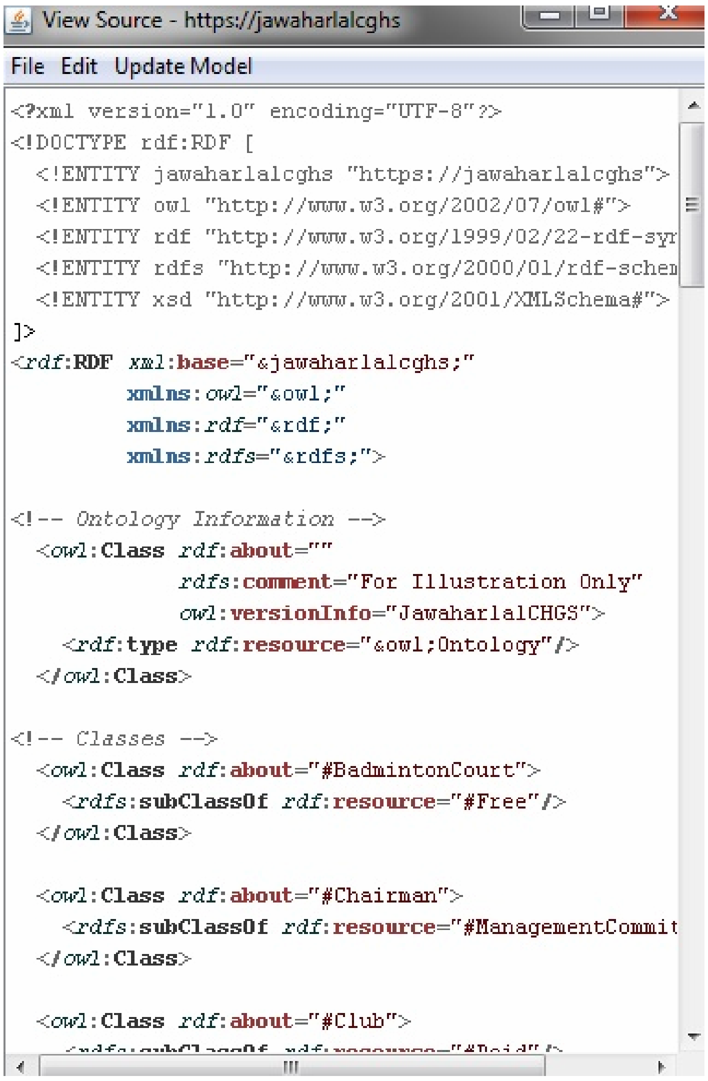
Equivalent

Annotea support in Swoop is provided via simple plug in whose implementation is based on the standard W3C Annotea protocols and uses the default Annotea RDF schema to specify annotations. Any Public Annotea Server can then be used to publish and distribute the annotations created in Swoop. The default annotation types (comment, advice, example, etc.) seem to be an adequate base for human oriented ontology annotations.

4 Code Snippet

RDF/XML code may be generated easily by clicking the ‘Source-RDF/XML’ which is sub-option of ‘View’.

Figure 9 Code Snippet (see online version for colours)

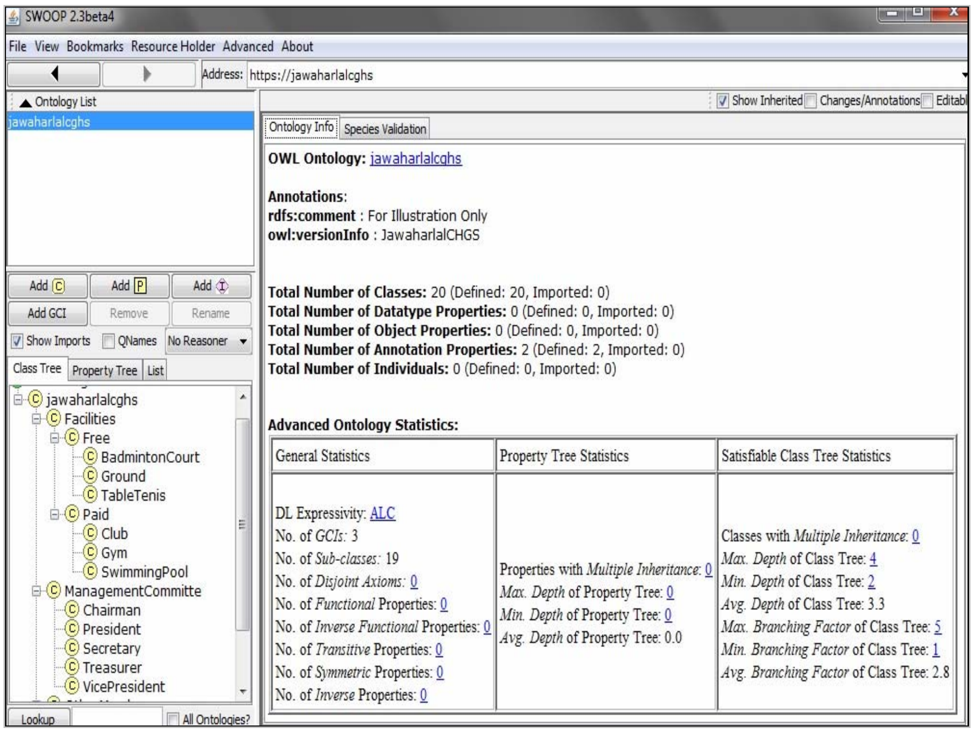


```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE rdf:RDF [
  <!ENTITY jawaharlalcghs "https://jawaharlalcghs">
  <!ENTITY owl "http://www.w3.org/2002/07/owl#">
  <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syr
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schen
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">
]>
<rdf:RDF xml:base="&jawaharlalcghs;"
         xmlns:owl="&owl;"
         xmlns:rdf="&rdf;"
         xmlns:rdfs="&rdfs;">

<!-- Ontology Information -->
  <owl:Class rdf:about=""
             rdfs:comment="For Illustration Only"
             owl:versionInfo="JawaharlalCHGS">
    <rdf:type rdf:resource="&owl;Ontology"/>
  </owl:Class>

<!-- Classes -->
  <owl:Class rdf:about="#BadmintonCourt">
    <rdfs:subClassOf rdf:resource="#Free"/>
  </owl:Class>

  <owl:Class rdf:about="#Chairman">
    <rdfs:subClassOf rdf:resource="#ManagementCommit
  </owl:Class>

  <owl:Class rdf:about="#Club">
    <rdfs:subClassOf rdf:resource="#Paid"/>
```

5 Ontology information statistics

The following observations are obtained from the ‘Jawaharlal CHGS ontology statistics’ which comprise of ‘General Statistics’, ‘Property Tree Statistics’ and ‘Satisfiable Class Tree Statistics’ headed by ‘Advanced Ontology Statistics’. Similarly, other relative statistics may also be analysed.

Figure 10 ‘Jawaharlal CGHS ontology’ statistics (see online version for colours)



jawaharlalcghs
- Facilities
  - Free
    - BadmintonCourt
    - Ground
    - TableTenis
  - Paid
    - Club
    - Gym
    - SwimmingPool
- ManagementCommitte
  - Chairman
  - President
  - Secretary
  - Treasurer
  - VicePresident

OWL Ontology: jawaharlalcghs

Annotations:
rdfs:comment : For Illustration Only
owl:versionInfo : JawaharlalCHGS

Total Number of Classes: 20 (Defined: 20, Imported: 0)
Total Number of Datatype Properties: 0 (Defined: 0, Imported: 0)
Total Number of Object Properties: 0 (Defined: 0, Imported: 0)
Total Number of Annotation Properties: 2 (Defined: 2, Imported: 0)
Total Number of Individuals: 0 (Defined: 0, Imported: 0)

Advanced Ontology Statistics:

General Statistics	Property Tree Statistics	Satisfiable Class Tree Statistics
DL Expressivity: ALC No. of *GCIs*: 3 No. of *Sub-classes*: 19 No. of *Disjoint Axioms*: 0 No. of *Functional* Properties: 0 No. of *Inverse Functional* Properties: 0 No. of *Transitive* Properties: 0 No. of *Symmetric* Properties: 0 No. of *Inverse* Properties: 0	Properties with *Multiple Inheritance*: 0 *Max. Depth* of Property Tree: 0 *Min. Depth* of Property Tree: 0 *Avg. Depth* of Property Tree: 0.0	Classes with *Multiple Inheritance*: 0 *Max. Depth* of Class Tree: 4 *Min. Depth* of Class Tree: 2 *Avg. Depth* of Class Tree: 3.3 *Max. Branching Factor* of Class Tree: 5 *Min. Branching Factor* of Class Tree: 1 *Avg. Branching Factor* of Class Tree: 2.8

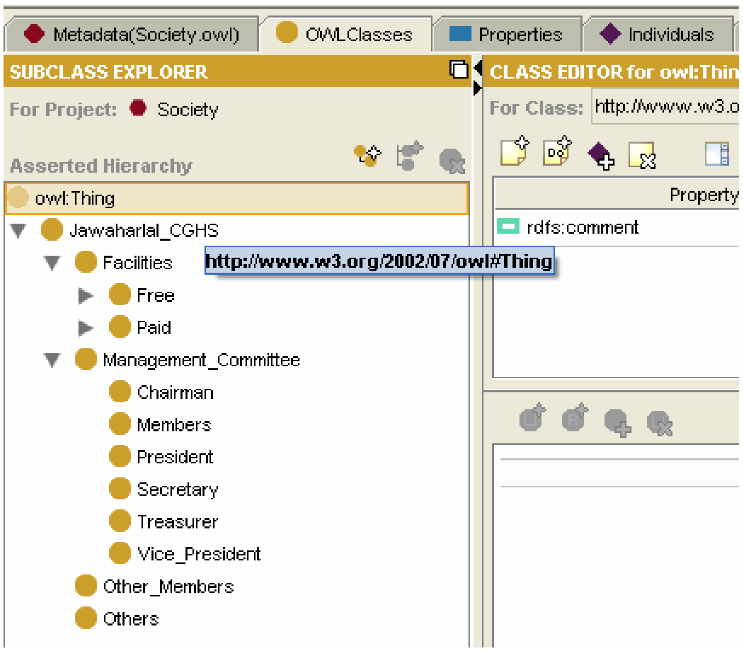
Results/observations

SuperClasses	Facilities, management committee
Total no. of classes	20
Total no. of GCIs	3
Total no. of sub classes	19
Total no. of annotations	02(version Info, comment)
Max depth of class tree	4
Min depth of class tree	2
Average depth of class tree	3.3
Max branching factor of class tree	5
Min branching factor of class tree	1
Avg. branching factor of class tree	2.8

6 Ontology creation in Protégé

Now, we create the above ontology in Protégé 3.4.4.

Figure 11 Showing class hierarchy of Jawaharlal cooperation housing society using Protégé 3.4.4 (see online version for colours)



- owl:Thing
  - Jawaharlal_CGHS
    - Facilities
      - Free
      - Paid
    - Management_Committee
      - Chairman
      - Members
      - President
      - Secretary
      - Treasurer
      - Vice_President
    - Other_Members
    - Others

We obtain the above screen by clicking on OWL classes tab on the top of the Protégé editor. This hierarchy has super classes and sub-classes. Jawaharlal_CGHS is the super class and the sub-classes of Jawaharlal_CGHS are: Facilities, Management_Committee, Other_Members and Others. The sub-classes of Management_Commitee are: Chairman, Members, President, Secretary, Treasurer and Vice_President. The subclasses of 'Facilities' class are 'Paid' and 'Free'. Now, we present a pictorial view of the sub-classes. The sub-classes of 'Facilities' class are 'Paid' and 'Free' created using OWLVIZ plugin of Protégé 3.4.4.

A query language is used for querying RDF graphs known as SPARQL which stands for 'Simple Protocol and RDF query Language', which is basically an RDF query language that defines a data access protocol and standard query language to be used with the RDF data model. SPARQL Query can also be used to retrieve data from RDFS (RDF Schema) as well as from Web Ontology Language (OWL) (Euzenat, 2006) which can be created and executed on ontology tools like Protégé. Now, SPARQL queries can be executed on the given ontology showing various sub-classes and super classes. Below is the illustration of executing an SPARQL query on the given ontology showing various super classes and sub-classes.

Figure 12 Pictorial representation showing sub-classes of 'Facilities' class using OWLVIZ plugin of Protégé 3.4.4 (see online version for colours)

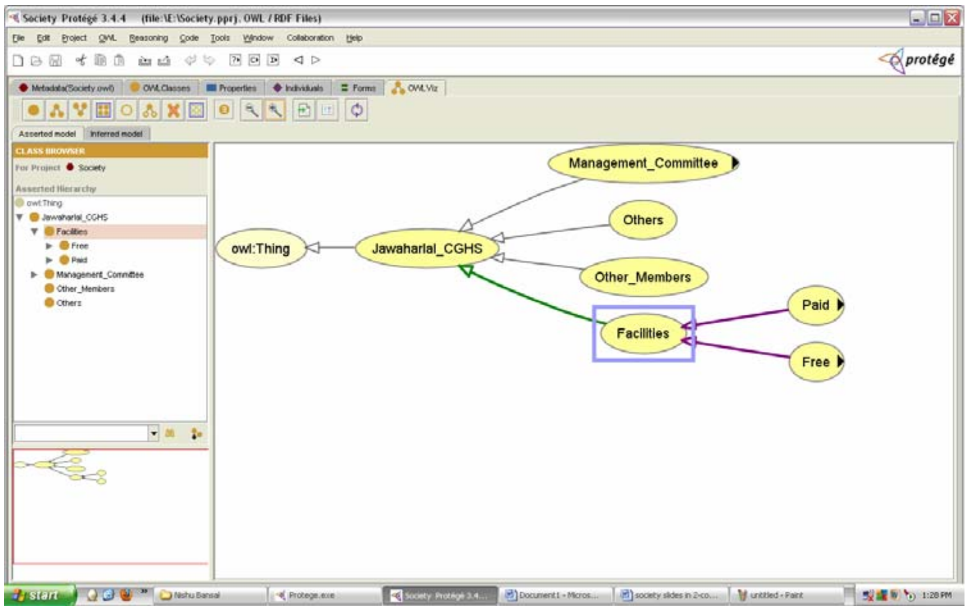
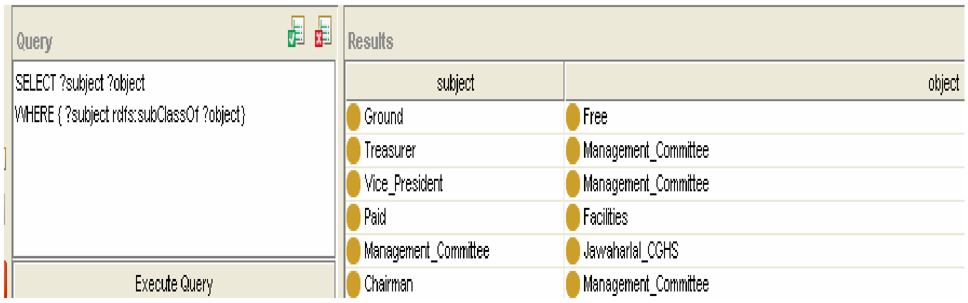


Figure 13 Sub-class and super-class result by executing the SPARQL query on the given ontology (see online version for colours)



```
SELECT ?subject ?object
WHERE { ?subject rdfs:subClassOf ?object}
```

subject	object
Ground	Free
Treasurer	Management_Committee
Vice_President	Management_Committee
Paid	Facilities
Management_Committee	Jawaharlal_CGHS
Chairman	Management_Committee

The SPARQL query when executed on the given ontology shows the relationship subclass-superclass results. Similarly various queries may be executed and the work may be populated.

The *Code Snippet* is as follows:

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

```
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xml:base="file:/E:/Society.owl">
<owl:Ontology rdf:about="file:/E:/Society.owl"/>
<owl:Class rdf:ID="Club">
<rdfs:subClassOf>
<owl:Class rdf:ID="Paid"/>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Free">
<rdfs:subClassOf>
<owl:Class rdf:ID="Facilities"/>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Secretary">
<rdfs:subClassOf>
<owl:Class rdf:ID="Management_Committee"/>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Swimming_Pool">
<rdfs:subClassOf>
<owl:Class rdf:about="file:/E:/Society.owl#Paid"/>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="file:/E:/Society.owl#Management_Committee">
<rdfs:subClassOf>
<owl:Class rdf:ID="Jawaharlal_CGHS"/>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Chairman">
<rdfs:subClassOf rdf:resource="file:/E:/Society.owl#Management_Committee"/>
</owl:Class>
<owl:Class rdf:ID="Gym">
<rdfs:subClassOf>
<owl:Class rdf:about="file:/E:/Society.owl#Paid"/>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Members">
```

```
<rdfs:subClassOf rdf:resource="file:/E:/Society.owl#Management_Committee"/>
</owl:Class>
<owl:Class rdf:ID="Other_Members">
<rdfs:subClassOf rdf:resource="file:/E:/Society.owl#Jawaharlal_CGHS"/>
</owl:Class>
<owl:Class rdf:ID="Treasurer">
<rdfs:subClassOf rdf:resource="file:/E:/Society.owl#Management_Committee"/>
</owl:Class>
<owl:Class rdf:ID="Others">
<rdfs:subClassOf rdf:resource="file:/E:/Society.owl#Jawaharlal_CGHS"/>
</owl:Class>
<owl:Class rdf:ID="Table_Tennis">
<rdfs:subClassOf rdf:resource="file:/E:/Society.owl#Free"/>
</owl:Class>
<owl:Class rdf:ID="Ground">
<rdfs:subClassOf rdf:resource="file:/E:/Society.owl#Free"/>
</owl:Class>
<owl:Class rdf:about="file:/E:/Society.owl#Paid">
<rdfs:subClassOf>
<owl:Class rdf:about="file:/E:/Society.owl#Facilities"/>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Vice_President">
<rdfs:subClassOf rdf:resource="file:/E:/Society.owl#Management_Committee"/>
</owl:Class>
<owl:Class rdf:ID="President">
<rdfs:subClassOf rdf:resource="file:/E:/Society.owl#Management_Committee"/>
</owl:Class>
<owl:Class rdf:about="file:/E:/Society.owl#Facilities">
<rdfs:subClassOf rdf:resource="file:/E:/Society.owl#Jawaharlal_CGHS"/>
</owl:Class>
<owl:Class rdf:ID="Badminton_Court">
<rdfs:subClassOf rdf:resource="file:/E:/Society.owl#Free"/>
</owl:Class>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.4.4, Build 579) http://protege.stanford.edu -->
```

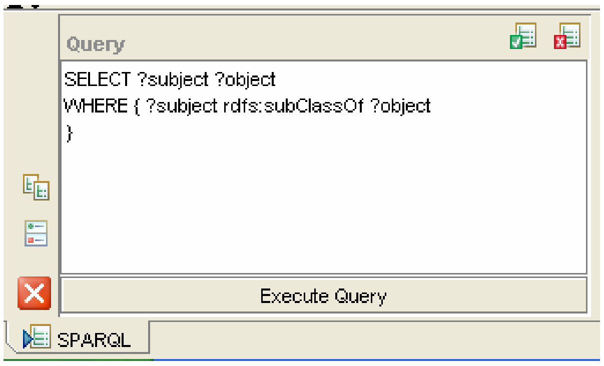
The above OWL code can be used in Protégé tool for obtaining the subject and object relationship i.e., superclass and subclass hierarchy.

7 Executing a SPARQL query on Jawaharlal CGHS ontology using Protégé

We click on 'open SPARQL query panel' from OWL in menu bar of Protégé 3.3.1. Then, let us take a SPARQL query as below and click on 'Execute' at the bottom which will display the output on the right hand side panel.

```
SPARQL Syntax:-
SELECT ?subject ?object
WHERE { ?subject rdfs:subClassOf ?object }
```

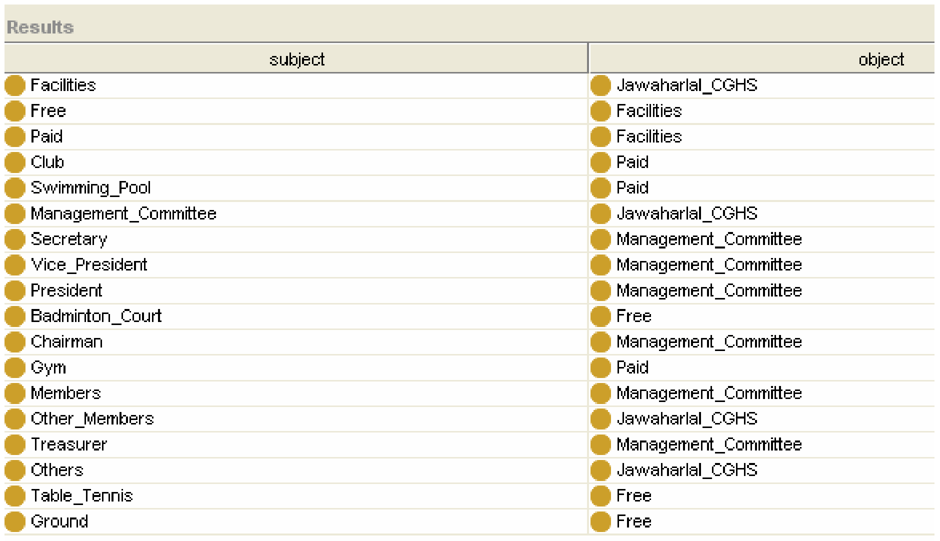
Figure 14 Query written in the query panel (see online version for colours)



```
SELECT ?subject ?object
WHERE { ?subject rdfs:subClassOf ?object
}
```

When the above SPARQL Query is executed on the OWL file of Jawaharlal CGHS ontology created in Protégé, following output result is obtained:

Figure 15 Output obtained from SPARQL execution on OWL code (see online version for colours)



Results

subject	object
● Facilities	● Jawaharlal_CGHS
● Free	● Facilities
● Paid	● Facilities
● Club	● Paid
● Swimming_Pool	● Paid
● Management_Committee	● Jawaharlal_CGHS
● Secretary	● Management_Committee
● Vice_President	● Management_Committee
● President	● Management_Committee
● Badminton_Court	● Free
● Chairman	● Management_Committee
● Gym	● Paid
● Members	● Management_Committee
● Other_Members	● Jawaharlal_CGHS
● Treasurer	● Management_Committee
● Others	● Jawaharlal_CGHS
● Table_Tennis	● Free
● Ground	● Free

The above output snapshot displays the results of the SPARQL executed using Protégé which displays 'Subject' and 'Object' showing the subclass and superclass relationships e.g., 'Vice Chancellor(subject) is the subclass of GGSIP_University(object)'

8 Conclusions and future work

An illustration of a group housing society ontology to explore the usage of Swoop and Protégé has been presented with classes and sub-classes and, in Swoop, semantics is created by using general concept inclusion axiom and displayed by class expression table. Query retrieval process with graphical CropCircle and use of partition automatically option has been illustrated. We infer that code can be copied and used in other tools so that structure of the ontology is not disturbed and similar ontologies don't need merging rather relevant or semantic classes are selected from other ontology. Then, the same ontology creation is illustrated in Protégé 3.4.4 along with a brief comparison of Swoop and Protege. Some of the solutions or steps proposed in the paper need to be elaborated upon, implemented and optimised in SWOOP and Protégé in the future. Moreover, a formal evaluation of the features needs to be done by performing usability studies and comparing it against existing ontology editor tools. More options in Swoop and Protégé need to be presented and a detailed comparison may be made.

References

Alesso, H.P. and Smith, C.F. (2006) *Thinking on the Web, Berners-Lee, Godel and Turing*, p.14, Wiley-Interscience (John Wiley & Sons).

Antoniou, G. and Harmelen, F.V. (2008) *A Semantic Web Primer*, The MIT Press, Massachusetts, London, England.

Berners-Lee, T., Hendler, J. and Lassila, O. (2001) 'The semantic web', *Scientific American*, May, Vol. 284, No. 5, pp.34–43.

Chandrasekran, B., Josephson, J.R. and Benjamins, V.R. (1999) 'What are ontologies and why do we need them', *IEEE Intelligent Systems*, January/February, Vol. 14, No. 1, p.1.

Gruber, T. (1995) 'Towards principles for the design of ontologies used for knowledge sharing', *International Journal of Human-Computer Studies*, Vol. 43, Nos. 5–6, p.1.

http://protege.stanford.edu/

http://www.mindswap.org/2004/SWOOP/

Lambrix, P. (2005) 'Towards a semantic web for bioinformatics using ontology-based annotation', *Proceedings of the 14th IEEE International Workshop on Enabling Technologies: Infrastructures for Collaborative Enterprises*, pp.3–7.

Malik, S.K. and Rizvi, S.A.M. (2012) 'A framework for SPARQL query processing, optimization and execution with illustrations', *International Journal of Computer Information Systems and Industrial Management Applications*, Vol. 4, pp.208–218, ISSN 2150-7988, available at http://www.mirlabs.org/ijcisim/volume1.html.

Noy, N.F. and McGuinness, D.L. (2000) *Ontology Development 101: A Guide to Creating Your First Ontology*, Stanford University, CA.