Ontology Graphical UI

Arona Dorin Chowdhury - 1520045642; Mohammad Raihan Sarker Razu - 1520079042; Tamim Ahmed - 1520698642
499A Project Report; North South University

ABSTRACT

The purpose of this project is to create an interface where people can draw the graphs of their model easily and can edit anytime. In short, people can make an ontology very effectively. On many websites, we can see the graphs but we cannot edit the graph any time we want. Even once we draw a graph later we cannot add other classes or sub-classes. In that case, we have to redraw the full graph again. By protege, we can easily edit the graph according to the requirements.

1 LIST OF FIGURES

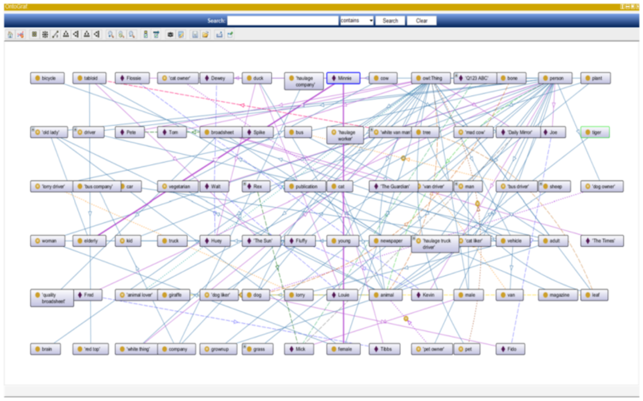


Figure 1: People ontology graph

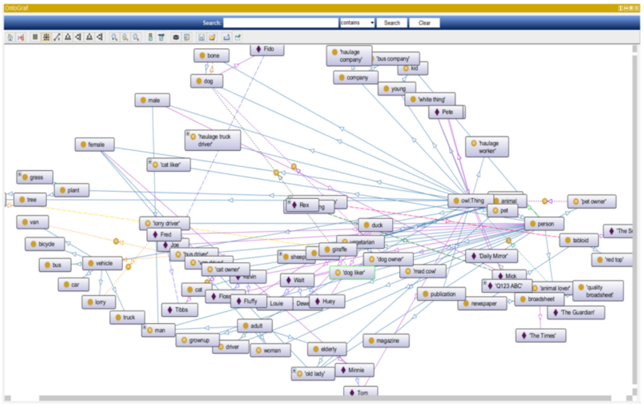
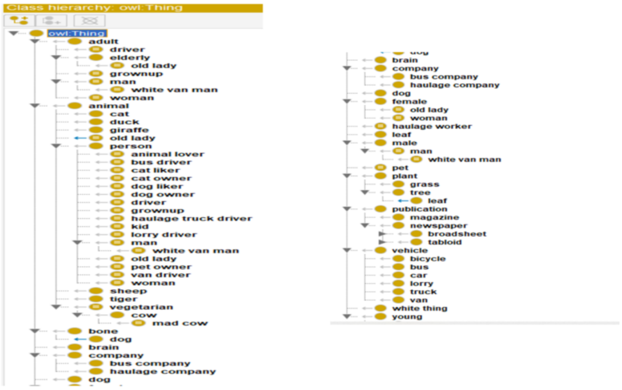
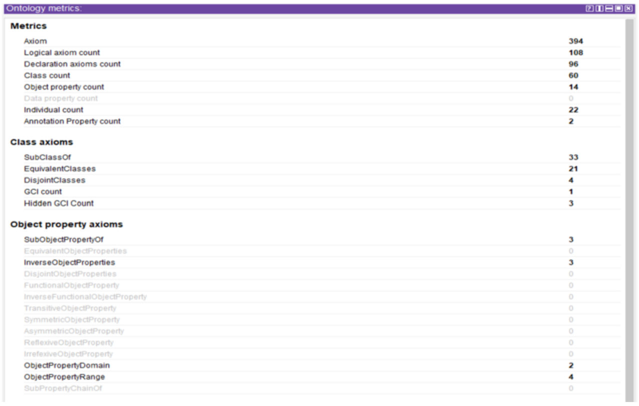


Figure 2: People ontology graph in different viewing design.



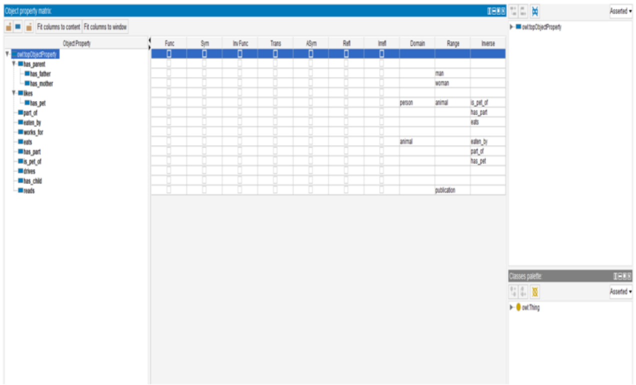
- owl:Thing
  - adult
    - driver
    - elderly
      - old lady
    - grownup
      - man
        - white van man
      - woman
  - animal
    - cat
    - duck
    - giraffe
    - old lady
    - person
      - animal lover
      - bus driver
      - cat liker
      - cat owner
      - dog liker
      - dog owner
      - driver
      - grownup
      - haulage truck driver
      - kid
      - lorry driver
      - man
        - white van man
      - old lady
      - pet owner
      - van driver
      - woman
    - sheep
    - tiger
    - vegetarian
      - cow
        - mad cow
  - bone
    - dog
  - brain
  - company
    - bus company
    - haulage company
  - dog
  - brain
  - company
    - bus company
    - haulage company
  - dog
  - female
    - old lady
    - woman
  - haulage worker
  - leaf
  - male
    - man
      - white van man
  - pet
  - plant
    - grass
    - tree
      - leaf
  - publication
    - magazine
    - newspaper
      - broadsheet
      - tabloid
  - vehicle
    - bicycle
    - bus
    - car
    - lorry
    - truck
    - van
  - white thing
  - young

Figure 3: Class hierarchy


Ontology metrics:

Metrics	
Axiom	394
Logical axiom count	108
Declaration axioms count	96
Class count	60
Object property count	14
Data property count	0
Individual count	22
Annotation Property count	2
Class axioms	
SubClassOf	33
EquivalentClasses	21
DisjointClasses	4
GCI count	1
Hidden GCI Count	3
Object property axioms	
SubObjectPropertyOf	3
EquivalentObjectProperties	0
InverseObjectProperties	3
DisjointObjectProperties	0
FunctionalObjectProperty	0
InverseFunctionalObjectProperty	0
TransitiveObjectProperty	0
SymmetricObjectProperty	0
AsymmetricObjectProperty	0
ReflexiveObjectProperty	0
IrrefexiveObjectProperty	0
ObjectPropertyDomain	2
ObjectPropertyRange	4
SubPropertyChainOf	0

Figure 4: Ontology matrix.



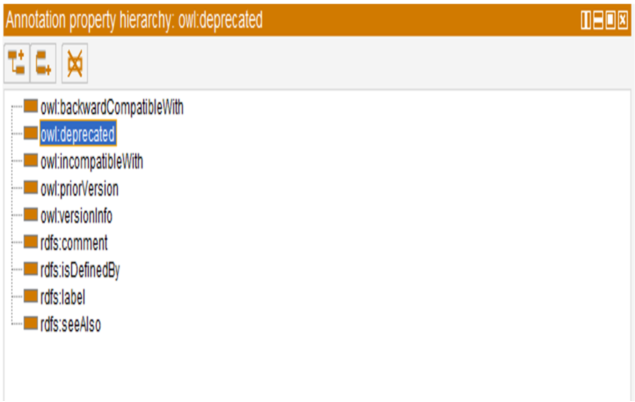
Object Property	Func	Sym	Inv Func	Trans	ASym	Refl	Irrefl	Domain	Range	Inverse
owl:topObjectProperty	□	□	□	□	□	□	□			
has_parent	□	□	□	□	□	□	□			
has_father	□	□	□	□	□	□	□		man	
has_mother	□	□	□	□	□	□	□		woman	
likes	□	□	□	□	□	□	□			
has_pet	□	□	□	□	□	□	□	person	animal	is_pet_of
part_of	□	□	□	□	□	□	□			has_part
eaten_by	□	□	□	□	□	□	□			eats
works_for	□	□	□	□	□	□	□			
eats	□	□	□	□	□	□	□	animal		eaten_by
has_part	□	□	□	□	□	□	□			part_of
is_pet_of	□	□	□	□	□	□	□			has_pet
drives	□	□	□	□	□	□	□			
has_child	□	□	□	□	□	□	□			
reads	□	□	□	□	□	□	□		publication	

Figure 5: Object property matrix.

2 INTRODUCTION

The present is the age of information technology. Now the most valuable resource in the world is information or data. The world's largest tech companies rely on this information to make money. The

, ,
© 2020
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn


Annotation property hierarchy: owl:deprecated

- owl:backwardCompatibleWith
- owl:deprecated
- owl:incompatibleWith
- owl:priorVersion
- owl:versionInfo
- rdfs:comment
- rdfs:isDefinedBy
- rdfs:label
- rdfs:seeAlso

Figure 6: Annotation property hierarchy.

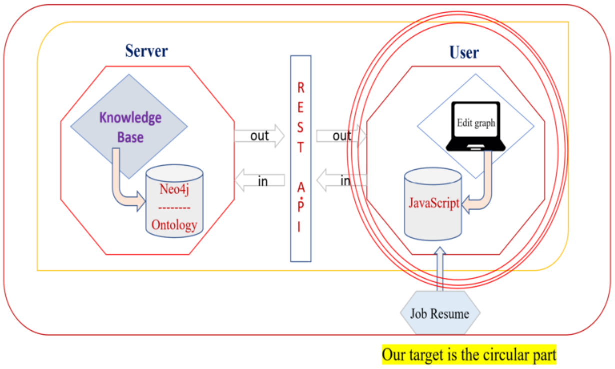


Figure 7: Project basic idea

most interesting thing is that the information is linked to one way or another. For example, when we say cancer, we understand that it is related to medical, any algorithm means related to computer science, processor update means related to electrical engineering, etc. These data are stored in the database in a static format. Which we do not directly understand which is associated with which. Ontology is used for a visual representation of data in dynamic form. (See figure 1 on page 1).

We can easily modify the ontology graph without any data change. (See figure 2 on page 1).

We know that some software is already being used to create data ontology [2] (such as protege) and it has some minor problems. So we initially want to create an ontology UI, so that the user can easily add, delete, rename data as class, sub-classes, or sub sub-classes. We plan to create it using web technology so that the user can use it very easily, so we don't need any PC configuration; anyone can take this facility only with the browser through the internet.

3 BACKGROUND AND RELATED WORK

Protege is directly involved with our work. So for the convenience of our work we have done some research on Protege.

Protege- is a free, open source ontology editor and a knowledge management system. Protege provides a graphic user interface to define ontologies [1]. It also includes deductive classifiers to validate that models are consistent and to infer new information based on the analysis of an ontology. Like Eclipse, Protege is a framework for which various other projects suggest plugins. This application is written in Java and heavily uses Swing to create the user interface. Protege recently has over 300,000 registered users. According to a 2009 book it is "the leading ontological engineering tool [3]. Protege is being developed at Stanford University and is made available under the BSD 2-clause license. Earlier versions of the tool were developed in collaboration with the University of Manchester.

Note- In our project we are using latest version of Protege. Version is 5.5.0 which was last update 2019 14th march.

Using Protege- First, make sure that "DIG Reasoner" is selected in the Reasoning menu (the default selection when you install Protege). The Reasoning menu allows you to select which is the current reasoned that should be used when the users classifies the ontology, checks the consistency, computes the inferred types, etc [4].

How works Protege- Protege is a good exploratory and experimentation environment. Quick iterations are possible between model, data, and application changes. Database tables are designed and optimized to work with a particular application in mind. Instead access the data though the Protege API.

Some important figures for understanding Protege- See figure 1 on page 1 shows the classes and sub-classes. This is the class hierarchy.

See figure 4 on page 1 shows the ontology matrix. It includes the properties and axioms.

See figure 5 on page 1 shows the object property matrix.

See figure 6 on page 2 shows the annotation property hierarchy.

4 METHODOLOGY/DESIGN

Here is the diagrams for the complete system. (See figure 7 on page 2).

Protege is a free open source ontology editor. It is used for drawing many graphs and easily can be edited at any time. Anyone can download this software from its official website because it is free. After installing this software we saw many libraries in this. They work differently while making the graph. We can create classes, sub-classes, and sub sub-classes. We can choose many types of viewing options as well. We create classes and add sub-classes in other sections (mentioned in the list of figures section). But at the very beginning when we tried to use this software, we faced many problems. And because this software is used in biomedicine, e-commerce, and organizational modeling, we rarely find any tutorial videos on the internet. Also, we read some papers according to our faculty's advice, but we could not find anything helpful. So it took a few weeks more than our expectations. But by the time we could figure it out by ourselves how this software works. Now we can create the graphs of any model we want. We can save a file and edit the graph anytime we want. Also, we can add or delete classes and sub-classes anytime according to our choices. By ontology-graph active ontology tab, we can show the model in graphs. The rest part will be completed in the 499B.

5 RESULTS

As mentioned earlier, we can draw the graphs of the models. That means if we have information about the divisions, cities of a country, we can draw this information graph by the protege software. protege creates dynamic graphs. Now we are facing a new problem to connect this software in a website. There are no such libraries which can connect protege to a website. As we cannot have access to protege privacy so also it is not possible to create a library framework by raw codes. We have given the pictures below. As our project is not completed so the final result will be added in 499B.

6 ACKNOWLEDGMENT

By the kindness of the Almighty, we have completed our senior design project entitled “Ontology Graphical UI” Our deep gratitude goes first to my faculty advisor Dr. Mohammad Ashrafuzzaman Khan, Assistant Professor of North South University for giving this project idea and having faith in us with this project. Our faculty expertly guided us in our senior design project A throughout the whole CSE499A. His guidance helped us in all types of research, writings, and completing the first part of this project. Our sincere thanks also go to North South University, Dhaka, Bangladesh for giving us such a platform where we can have an industrial level experience as a part of our academics. This part of our project is completed by Mohammad Raihan Sarker Razu (1520079042), Tamim Ahmed (1520698642), and Arona Dorin Chowdhry (1520045642). Each has contributed to this project equally by completing their part. Last but not the least, we would like to thank our family as their inspiration and guidance kept us focused and motivated.

REFERENCES

[1] John H Gennari, Mark A Musen, Ray W Fergerson, William E Grosso, Monica Crubézy, Henrik Eriksson, Natalya F Noy, and Samson W Tu. 2003. The evolution of Protégé: an environment for knowledge-based systems development. *International Journal of Human-computer studies* 58, 1 (2003), 89–123.

[2] Holger Knublauch. 2004. Ontology-driven software development in the context of the semantic web: An example scenario with Protege/OWL. In *1st International workshop on the model-driven semantic web (MDSW2004).* Monterey, California, USA.[WWW document] http://www. knublauch. com . . . , 381–401.

[3] Mark A Musen. 2015. The protégé project: a look back and a look forward. *AI matters* 1, 4 (2015), 4–12.

[4] Pornpit Wongthongtham, Elizabeth Chang, Tharam S Dillon, and Ian Sommerville. 2005. Software engineering ontologies and their implementation. In *Proceedings of the IASTED International Conference on Software Engineering (SE).* ACTA Publishing, 208–213.