Journal of Theoretical and Applied Information Technology
15 July 2012. Vol. 41 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

COMPREHENSIVE ONTOLOGY COGNITIVE ASSISTED VISUALIZATION TOOLS—A SURVEY

V. SWAMINATHAN, R. SIVAKUMAR
Department of Computer Science, A.V.V.M. Sri Pushpam College,
Bharathidasan University, Trichirappalli, India

E-mail: vswaminathanthanjavur@yahoo.com, rskumar.avvmspc@gmail.com

ABSTRACT

Comprehensive Ontology is used in different contexts with different authors. Ontology comprehension is a collection of techniques that facilitate the understanding of ontologies. These tools are being applied for further development in various disciplines for better understanding knowledge. These tools commonly use six methods, namely, Indented List, Node-link and tree, Zoomable Visualization, Space-filling, Focus + context or distortion and 3D Information landscapes. The purpose of the work is to present a study on application of these six methods in the development of different kinds of Comprehensive visualization Ontology tools and categorize their characteristics so that it assists in method selection and promotes further future research in the area of ontology visualization. This paper overviews different technology for ontology visualization. It is an attempt to summarize existing literature related to ontology visualization and provide comprehensive cataloguing of existing method characteristics as well as record their strong points and weaknesses in relation with user tasks.

Keywords: *Comprehensive Ontology, Ontology Visualization Technique, Cognitive Support, Human Agents*

1. INTRODUCTION

Visualization is used as a cognitive aid for managing ontology and knowledge representations. The continuing need for more effective information retrieval has lead to the creation of the notions of the semantic web and personalized information management areas of study that very often employ ontology to represent the semantic context of a domain. The Ontology is machine understandable and thus needs some means of graphical visualization for humans to comprehend. Consequently, the need for effective ontology visualization for design, management and browsing has arisen. Visualization tools make it easy to understand large and complex ontology important for easy representation of selected parts. There are several ontology visualizations available through the existing ontology management tools. Ontology is a term initially borrowed from Philosophy, where ontology is a systematic account of existence that is trying to answer the question 'what properties can explain the existence'. In computer science according to Gruber [1], an ontology is an explicit specification of a conceptualization. The term "conceptualization" is defined as an abstract, simplified view of the world, which needs to be represented for some purpose. It contains the objects, concepts, and other entities that are presumed to exist in some area of interest, and the relations that hold among them. That is Ontology describes basic concepts in a domain and defines relations among them.

Many ontology visualization techniques have been already developed such as Protege class browser, OntoVis, IsaViz, Space Tree, OntoSphere, Jambalaya, Crop Circles, GopherVR, Prot´eg´e TGVizTab and 3D Hyper- bolic Tree. They are reported in literature [3, 4]. However, the research society is in need of knowing the recent developments and advances in ontology tools. Hence this paper focuses on studying the features of recently developed tools such as Altova Semantic Works, FlexViz, Knoodl-OntoVis, Ontopia, Collaborative Protégé and Wandora, which may help the researchers to think in different directions.

The remaining part of the paper is organized as follows: Section 2 presents the overview of ontology comprehensive and visualization techniques. Next, Section 3 presents ontology characteristics and the complete survey of different ontology tools. A discussion is also presented in Section 4. Finally Section 5 concludes this paper.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

2. ONTOLOGY COMPREHENSIONS

The term ontology comprehension is used in different contexts with different meanings by different authors (for example [2, 3, 4]). For this reason it is important to clarify the meaning of ontology comprehension within the context of this work. Ontology comprehension is the interaction between human agents and the knowledge expressed in an ontology. Ontology comprehension is a collection of techniques that facilitate the understanding of ontologies. An ontology understanding technique is an abstract idea that can have many implementations. For example, two computer programs can implement the same technique in different ways [5].Ontology understanding techniques can operate independently, or can interact to enhance understanding. A depiction of ontology comprehension frame work is given in Fig. 1.

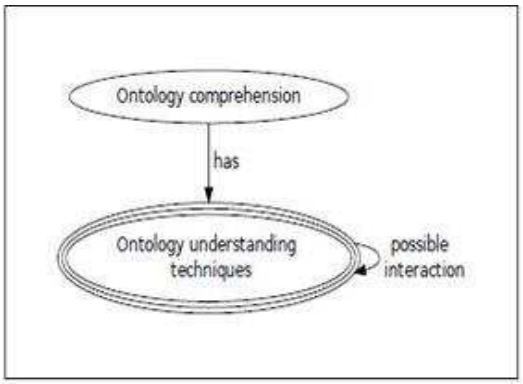

Figure 1: Ontology Comprehension framework.

2 .1 VISUALIZATION TECHNIQUES

The visualization techniques are presented in tree or graph representation for the presentation of all existing ontology management tools in existing ontology visualizations. For a method to be eligible for the visualization of an ontology, it has to support the presentation of ontology ingredients i.e. classes (or entity types), relations, instances and properties (or slots).

The methods can be grouped according to different characteristics of the presentation, interaction technique, functionality supported or visualization dimensions. For the needs of this survey the methods were grouped in the following categories, representing their visualization type:

1. Indented List
2. Node–link and Tree
3. Zoomable
4. Space–filling
5. Focus + Context or Distortion
6. 3D Information Landscapes

The methods grouped in these six general categories were further categorized according to the number of space dimensions they employ, i.e. 2D and 3D. The 2D methods use the screen space as a plane and do not use any notion of depth. The 3D methods exploit the third dimension either to create visualizations that are closer to real world metaphors or to improve usage of space and/or usability. More specifically, these methods allow the user to manipulate—rotate and move—3D objects and/or to navigate inside the 3D space.

This second-level grouping was chosen due to the specific needs that characterize the 3D visualizations which are also reflected upon the interaction techniques employed and functionality which can be catered for target user group characteristics and even system requirements. 3D visualization, in general , requires increased system resources in order for navigation and viewing to be smooth and without delays and, as a result, is probably not suitable for web use .

3. ONTOLOGY PRESENTATION CHARACTERISTICS

Classes: The visualization method should display all the ontology classes, at once or at the request of the user, providing at least their name, in an intelligible manner.

Instances: The instances are the actual data associated with the ontology and in most cases what the user is actually interested in. However, representing them as nodes connected to a class is not always effective because of their great number and other alternatives should be used, like presenting the instances of a selected class as a list within a separate window.

Taxonomy (Is a relation): The presentation of the taxonomy on which the ontology is based is essential for understanding the inheritance relations between classes. The system should at least provide a holistic view of this taxonomy, in a hierarchical representation. Partial views, allowing the user to focus on a portion of the taxonomy, are also a desirable feature.

Multiple inheritances: The cases where a class has more than one parents are not easy to represent in combination with an effective representation of the taxonomy. It is desirable for the visualization to indicate nodes with multiple parents and provide efficient means to view all node direct ancestors. It

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

should be noted here that many of the presented ontology visualizations support multiple inheritance by replicating child nodes under all their parents. Hierarchical visualizations that currently do not support this feature could be adapted to support it.

Role relations: Role relations are essential, but like the multiple inheritance links, not easy to represent. Apart from the link that should be visible, a label with the link name (effectively, the role type) should also be displayed (possibly with the option to hide it, to avoid display cluttering). Multiple inheritance and role relations are two types of links that transform the ontology from a hierarchy to a graph, a structure inherently more difficult to represent than a tree.

Properties: The properties associated with an entity are also very important and a complete visualization should include their representation, either on the main ontology visualization or within separate space. Apart from these ontology presentation characteristics, two more are added. These are keyword search and software availability.

3.1 ALTOVA SEMANTIC WORKS

Altova Semantic Works®2012 is the ground-breaking visual RDF and OWL editor for the Semantic Web. Graphically designed RDF instance documents, RDFS vocabularies, and OWL ontologies, then output them in either RDF/XML or N-Triples formats. Semantic Works make the job easy with tabs for instances, properties, classes, etc., context-sensitive entry helpers, automatic format checking, and more. RDFS vocabularies define the allowable properties (predicates) for RDF instances within a particular domain. RDFS also allow defining classes to further classify the relationships between resources. Semantic-Works displays the instances, properties, and classes in an RDFS vocabulary on separate tabs, allowing one to view and edit these different items with ease. The Instance tab lists all resources in the document, and the Properties tab lists all the properties. When a property is selected in the main pane, the domain of that property is displayed in another window. The Class tab lists all the classes available in the vocabulary with a separate window that lists the instances and properties of the selected class as they appear throughout the Redstone can view and edit the details of any item listed by clicking its expand button. Semantic Works display resources graphically according to their associations with other resources. The Semantic Works display is highly configurable. One can adjust the width of the items in the graph, display it with a vertical or horizontal orientation, adjust the distances between parent and child nodes, and even change the font styles and colors used. To help immediately visualize class relationships, RDFS classes are enclosed in yellow boxes in the graphical display. Holding the mouse over any item or icon display reveals its meaning or corresponding URI. The same entry helpers and context-sensitive choices described in the RDF editing section above are available for RDFS editing, and syntax checking based on the RDFS specification ensure that one's document is valid. To creating a visual RDFS design, Semantic Works R 2012 is auto-generating the corresponding RDF/XML or N-Triples code behind the scenes, and one can view and edit it at any time by clicking the Text tab.

3.2 FLEXVIZ

Graphs are the basic concepts in discrete mathematics and data structure. The applications of graphs are very extensive and vary from common events to complex mathematical or computer science problems. The building blocks of a graph are vertices (nodes) and edges. Ontologies can be represented as a graph by using the basic properties of graphs, such as directed graphs. FlexViz is a graph based visualization tool written is Adobe Flex. It supports single ontology browsing. Nodes of graph are mapped as concepts and relationships (edges) between nodes (e.g. "is a", "depends on") are represented as arcs. It is designed to provide a light-score, interactive, and visually accessible ontologies on Web. [9] Figure 2 shows a demo of FlexViz, web-based visualization primarily render static images or text representations. However, FlexViz greatly enhances user tasks and promotes new technologies by making it very interactive and light-weight. [9] FlexViz has many interesting features such as filtering based on nodes (concepts) and edges (relationships), searching, results view in various graphical layout, customization of node and edge (arc) labels , customizing node and arc tool tips, zooming , forcibly stay of node in the screen ,back and forward button to navigate history, nodes expansion to show or hide children, colors of nodes and customization of edges, visualization widget with a fixed ontology and export of graph as a image data or xml file .

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

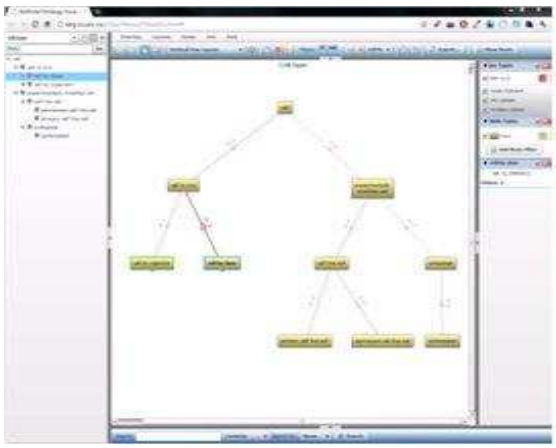

Figure 2: Flexviz Demo

3.3 KNOODL–ONTVIS

One can start up an OntVis page by going to any resource page in Knoll and clicking the "graph" tab. On the main ontology page the "graph tab" takes the user to a blank visualization. From any resource page, the graph will load with that resource in place. The class and property trees are crucial for making diagrams in OntVis. If the tree is not visible upon load, the user has to select it from the "View" menu. Once the tree menu is loaded, one can drag classes and properties from it onto the Graph section. The only way to view graphs containing instances is to click the graph tab on the knoll resource page for that instance. The toolbar helps to change the size of the graph, and also lets switch between the pointer and the eraser . To view relations and annotations for resources, the user has to click the three-bar menu symbol that appears when the mouse hovers over the resource. The menu that appears varies depending on whether the resource in question is a class, property, or instance, There are three options in classes first one, Show related classes which places subclasses, disjoint classes, and other related classes on the graph and in some cases, the relations that appear will make use of resources already on the graph and the second one, Show annotations, expands the resource's q2wsdef;klop'container to show its OWL annotations and the final one, Show properties, expands the resource's container to show related properties. The properties appear inside the resource box, may be dragged onto the graph. Note that properties, when dragged onto the graph, will appear as edges rather than as nodes. If one can want to view a property as a resource, one must drag it out from the tree view and also related properties are properties whose domain is declared to be the class in question, as well as properties with property restrictions declared on the class With properties, first one Show related classes , that places sub properties, inverse properties, and other related properties on the graph, in some cases, the relations that appear will make use of resources already on the graph. It also places the classes declared to be the domain and range of the property in question on the graph. With Instances, the first option, Show related instances, places related instances on the graph. The next one, Shows class assertions , expands the resource box to show which classes the instances is a member of and here the Class assertions may be dragged onto the graph, It also shows property assertions and shows properties asserted on this instance and their values where property assertions may be dragged onto the graph. Once the graph looking the way one wants, the user can choose "Export" under the "File" menu. One can choose PNG or PDF as the format.

3.4 ONTOPIA

Ontopoly is built as a client/server application. As a client, to use web browser, while the server is a web server bundled with the distribution. The server-side application is built using the Navigator Framework and Web Editor Framework, which are parts of the Ontopia Knowledge Suite. Ontopoly is accessible within Ontopia, an application that provides easy access to Ontopoly, Dominator, and Litigator. Monopoly's primary purpose is to enable the manual creation and maintenance of topic maps that may be based on a variety of ontologies. In order to be able to provide the most intuitive possible user interface for such a generic application, Ontopoly is ontology-driven. What this means is that the forms-based interface for creating and maintaining a topic map is generated automatically from the underlying ontology and the rules that are defined for it. Ontopoly is divided into two main parts: the ontology editor and the instance editor. The application also has an administration interface, i.e., a page for adding metadata to the topic map (description, creator, version, etc.), validating it, etc.; and an interface for exporting to various interchange syntaxes. The Topic Map Index Page allows the user to open an existing topic map, create a new one, or import one from outside Ontopoly. Once selected one of these actions, then it will take to the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

application pages. The overall architecture and the navigation paths between the various parts of the application are shown in Fig. 3. Monopoly's two primary functions provide a comprehensive Topic Maps editing environment such as configuring a topic map and Populating a topic map. Topic Map Index Page: the user before creates either the ontology or the instance should open a topic map. When one can first access Ontopoly from Ontopia, then take it to the Topic Map Index Page, on the left, is a column for Ontopoly Topic Maps, in the middle is the list of Other Topic Maps, and on the right is the area for creating new topic maps. Another column, Missing Topic Maps, will also appear on this page if an Ontopoly topic map has been deleted from outside syntaxes; it will also import RDF. A topic map created in Ontopoly (an Ontopoly topic map) will differ from one created outside of it (a non-Ontopoly topic map). An Ontopoly topic map carries along with it topics that let it interact with the Ontopoly application and topics that define the schema (the system topics). One can always export a topic map from Ontopoly to remove the system topics, but they are needed for the topic map to be understood within the application Similarly, pre-existing non-Ontopoly topic

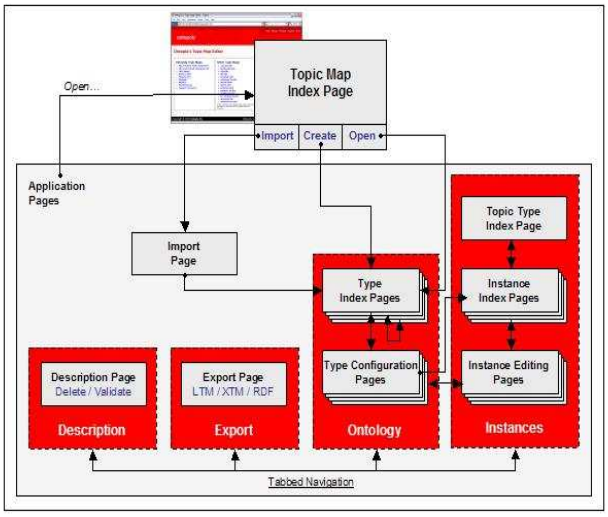

Figure 3: Overview of Monopoly's architecture

Maps will need these system topics to be added. There are five kinds of types in Topic Maps: topic types, occurrence types, association types, role types, and name types. Each of these has its own kind of Type Configuration Page, which is accessed via the links at the top on the Type Index Pages. Each Type Configuration Page is unique, but some fields are common to all of them. They are Name, Subject identifier, Read-only and Hidden. The additional properties specific to topic types are Abstract, Super class and Subclass.

3.5 COLLABORATIVE PROTÉGÉ

Collaborative Protégé is an extension of the existing Protégé system that supports collaborative ontology editing. In addition to the common ontology editing operations, it enables annotation of both ontology components and ontology changes. It supports the searching and filtering of user annotations, also known as notes, based on different criteria. One can implement two types of voting mechanisms that can be used for voting of change proposals. Multiple users may edit the same ontology at the same time. In multi-user mode, all changes made by one user are seen immediately by other users. There are two working modes available for Collaborative Protégé. Both modes such as multi-user mode and standalone mode support multiple users working on ontology. The main feature of Collaborative Protégé is the ability to create notes attached to different things. This is the same idea as if someone would read an article and would add marginal notes on the paper. In the same way, the notes mechanism of Collaborative Protégé allows a user to create his own remarks about a certain part of the ontology. This feature can also be used to discuss the ontology with other users either in standalone or multi-user mode. The notes are also called annotation. The main functionalities of Collaborative Protégé are annotation of classes and properties and instances with different types of notes (e.g., Comment, Advice, Example, etc.) The classical Protégé display shows the details of the different ontology components, such as classes, properties (slots) and instances. The collaboration panel adds functionality to support the collaborative development of ontologies. The collaboration panel is made up of two types of tabs such as Entity notes & Changes tabs and Ontology notes, all notes, Search and Chat tabs. In the main Protégé display user will also see the call-out icon if a property (slot) or an instance have annotations attached to them. A user may add notes attached to a specific ontology component in the Entity Notes. By ontology component, to mean classes, properties (or slots) and instances. The image shows an example of note attached to the Domain Concept class. The number of notes attached directly to Domain Concept is shown next to the Notes icon;

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3.6<NEW> WANDORA

Wandora is a general purpose information extraction, aggregation, management, and publishing application based on Topic Maps and Java. Wandora has graphical user interface, layered presentation of knowledge, several data storage options, huge collection of data extraction, import and export options, embedded server, and open plug-in architecture. Wandora is a FOSS application with GNU GPL license. Wandora is well suited for constructing information mashups and is capable of extracting and converting a wide range of open data feeds to Topic Map formats. Beyond Topic Maps conversion this feature allows Wandora user to aggregate multidimensional information mashups where information from Flicker interleaves with information from Geo Names and YouTube, for example. Pandora's embedded HTTP server enables easy mashup publication. Wandora can export Topic Maps in Grapheme format. The export starts with File > Export > Export Grapheme. Exported graph can be visualized with hyper graph application for example.

4. DISCUSSION

Altova supports the following bugs with its 2012 release: Arabic characters not accepted as class names by Semantic Works, classes in taxonomy are declared invalid, enable import of RDF schema in OWL Full, semantic check fails to note incorrect object property , URI ref prefixes toggles on/off spontaneously when editing file in Text View, syntax and semantic check in Semantic Works incorrectly report error on missing type in instance (OWL DL), semantic works crashes when hitting 'del' in the instances detail view while the drop down menu to change a data type is open, export function does not check for validity, failed verification of equivalence class extension, and rdf:XMLLiteral is not offered as data type for a value in RDF/OWL view. On the other hand, Flexi has many interesting features that can be summarized as follows: Filtering based on nodes (concepts) and edges (relationships), searching, results may be viewed in various graphical layout, customization of node and edge (arc) labels is possible, customizing node and arc tool tips, zooming, forcibly staying node in the screen (can cause nodes to overlap), back and forward button to navigate history, nodes can be expanded or collapsed to show or hide children, colors of nodes, edges are customizable, visualization can be displayed as a widget on a web page with a fixed ontology, graph can be exported as a image data or a xml file, source code is released in SourceForge9 which offers customization and extension based on requirement. With Ontvis, one can feel easy-to-use and easy-to-visualize capabilities to show the semantic contents, resource page, graph tab, manipulating the resources in the graph, edge decorations and exporting. As far as Ontopia is concerned it has the following advantages: detailed information of a topic and applicable to any Topic Map and can be generated automatically fast and cheap. It has also the following disadvantages: for every node a new graph is generated constantly changing visualization, user cannot create a mental model of the information structure, no big picture, usability is poor and only one visualization concept.

The Collaborative Protégé has the following advantages: Adding shared notes to ontology, discussion threads, view and discuss changes made to an ontology and live chat during editing. Wandora offers extractors with the following facts: Search engine extractors, HTML structures extractors, simple files extractors, news and syndication extractors, media extractors, micro format extractors, social & bookmark extractors, wake extractors, bibliographical extractors and simple RDF extractors. Similarly it provides various following generators: random graph generator, fully connected graph generator, tree graph generator, linear list graph generator, finite group graph generator, platonic solid graph generator, hypercube graph generator, tiling graph generator (square, triangular, and hexagonal tiling), edge generator and L-system generator.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 1: Summary of the ontology visualization characteristics in comprehensive tools.

Tool Name	Month & Year of implementation	Classes and Instances	Taxonomy	Multiple Inheritance	Role relations	Visualization type
Altova Semantic-Works	Oct. 2001	Class tab lists all the classes available in the vocabulary with a separate window that lists the instances	Graph View	Child nodes are placed under both parents	No. Supported through the properties window only.	Intended List
FlexViz	Jan. 2008.	No, Filtering based on nodes and edges	Graph View	No	Represented as arcs	**Zoomable**
Knoodl OntVis	Mar. 2011	Classes are represented as nodes & graph Tab	Tree view	No	No	Zoomable
ontopia	May 2010	Class and instance are created by populating by Topic map	Tree view	Subtypes will be inherited of its fields	Using parent/child combination	Node-link and Tree
Collaborative Protégé	Feb. 2009	Classes and Instances are Presented as Nodes in an indented, expandable and retractable tree.	Tree view	No	No	Node-link and Tree
(New) Wandora	Sep. 2011	class-instance relations as edges.	Graph View	No	No	Zoomable

5. CONCLUSION

In this paper, comprehensive ontology tools using ontology presentation characteristics were surveyed and analyzed. The visualization of comprehensive ontology tool is a particular sub problem of this area with many implications due to the various features that ontology visualization should present. The current work is an attempt to summarize the research that has been done so far in this area, providing an overview of the comprehensive ontology tools. As seen from the survey and the information provided in the table 1, it can be concluded that there is no one specific method that seems to be the most appropriate for all applications and, consequently, a viable solution is providing the user with several visualizations, so as to be able to choose the one that is the most appropriate for one's current needs.

REFERENCES

[1] Gruber. T.R., (1993), "A Translation approach to portable ontology specifications, knowledge acquisition special issue: current Issues in knowledge modeling", Vol. 5. Issue 2, 199–220.

[2] Chandra. V.K, Stickle. M.E, Thom ere. J.F, Salinger. R.J, (2000) et al. "Using prior knowledge: Problems and solutions". In National conference on artificial intelligence, pp. 436–442.

[3] Gibson. A, Wolstencroft. K and Stevens. R, (2007), "Promotion of ontological comprehension: Exposing terms and metadata with web 2.0", In Workshop on social and collaborative construction of structured knowledge at 16th International World Wide Web Conference, Retrieved on 2010/01/01 from http://citeseerx.ist.psu.edu/viewdoc/download?doi=11.142.8462{\&}rep=rep1{\&}type=pdf.[4] Keet. C, (2007), "Enhancing comprehension of ontologies and conceptual models through abstractions", Artificial intelligence and human- oriented computing, pp. 813–821.

[5] Johann Rath Bergh, (2010), "Ontology Comprehension", Division of Computer Science Stellenbosch University, Private Bag X1, 7602Matieland, South Africa.

[6] http://www.altova.com/semanticworks.html.

[7]http://www.altova.com/semanticworks/rdf-editor.html.

[8] Vivek Srivastava, (2011), "Methods to visualize ontology", Koblenz.

[9] Sean M. Falconer, Chris Callendar, and Margaretanne Storey, (2009) "FLEXVIZ: Visualizing Biomedical Ontologies on the Web", In International Conference on Biomedical Ontology, Software Demonstration, Buffalo, NY.

[10]http://knoodl.com/ui/groups/knoodl/wiki/ Help/entry/diagrams.

[11] Ontopoly: The Topic Map Editor-User's Guide.

[12] A Practical Guide To Building OWL Ontologies Using Prot´eg´e 4 and CO-ODE Tools (2009).

[13] Noy. N. F, Chugh. A, Liu. W, and Musen. M. A, (2006), " A framework for ontology evolution in collaborative environments", In 5th Intl. Semantic Web Conference, ISWC, volume LNCS 4273, Athens, GA, Springer.

[14] Tudorache. T,. Noy. N. F, and Musen. M. A, (2008), "Supporting collaborative ontology development in Protégé", In 7th Intl. Semantic Web Conference, ISWC 2008, Karlsruhe, Germany.

[15] Tania Tudorache, Jennifer Vendetti, Natalya F. Noy, (2008), "Web Prot´eg´e: A Lightweight OWL Ontology Editor for the Web".

[16] http://protegewiki.stanford.edu/wiki/ WebProtege

[17] http://protegewiki.stanford.edu/wiki/ Collaborative Protégé.

[18] http://www.wandora.org/wandora/wiki /index.php?httitle= Image: Exportxmlexample.gif.