Ontology Visualization: An Overview

Nassira Achich[1(✉)], Bassem Bouaziz[1], Alsayed Algergawy[2], and Faiez Gargouri[1]

[1] Higher Institute of computer science and Multimedia, University of Sfax, Sfax, Tunisia
achichnassira@gmail.com, {bassem.bouaziz,faiez.gargouri}@isims.usf.tn
[2] Friedrich-Schiller-Universität Jena, Jena, Germany
alsayed.algergawy@uni-jena.de

Abstract. As the main way for knowledge representation for the purpose of completely machine understanding, ontologies are widely used in different application domains. This full machine understanding makes them harder to be easily understood by a human. This necessitates the need to develop ontology visualization tools, which results in the existence of a large number of approaches and visualization tools. Along with this development direction, the number of published research papers related to ontology visualization is largely increasing. To this end, in this paper, we introduce a systemic review on different directions related to ontology visualization. In particular, we start by describing different application domains that make use of ontology visualization. Then, we propose a generic visualization pipeline that incorporates main steps in ontology visualization that could be later used as main criteria during comparing and discussing different visualization tools. By this review, we aim to introduce a general visualization pipeline that is useful when comparing ontology visualization tools and when developing a new visualization technique. Finally, the paper moves into the description of future trends and research issues that still need to be addressed.

1 Introduction

Ontologies are the basic components of the semantic Web, where underlying data are well structured for the purpose of full machine understanding. As defined by Gruber, "an ontology is a formal, explicit specification of a shared conceptualization". It consists of a set of concepts (classes), a set of attributes (data type properties), relationships (object properties), and constraints to abstractly represent a specific event [20,36]. An important aspect is how to facilitate the process of design, manage, and browsing such kind of complex structure. This results in a growing needs for ontology visualization tools that simplify the user involvement in these ontology-based management processes [24].

As a main way of knowledge representation, they have been becoming more and more largely used in different application domains. Even its importance in different application domains, ontology visualization is not a simple task. Since

A. Abraham et al. (Eds.): ISDA 2017, AISC 736, pp. 880–891, 2018.
https://doi.org/10.1007/978-3-319-76348-4_84

an ontology is more than a hierarchy of concepts. It further models role relations among concepts, and each concept has a set of properties attached to it. Therefore, a large number of ontology visualization approaches have been proposed and a set of tools have been developed [11,24,31,33,36]. Ontology visualizations is an important step for working with ontologies and it should provide a multi-level view in order to handle even entities, classes, properties, and blocks. To be useful, ontology graphical representation tools should encourage domain experts in the ontology creation and manipulating processes. Consequently, a smart visualization tool will enable the direct input from domain experts and reduce the dependency on knowledge engineers at every step of ontology development.

For example, by reducing the semantic gap between different overlapping representations of the same domain ontologies, visualizations techniques should provide support for ontology matching techniques. The process and the result of ontology matching need to be visualized as well, to get contextual feed back from user and produce better alignment results [2,11]. Ontologies visualization tools like *CODEX* [22], *REX* [10] and *OntoVIEW* [34] focus on ontology evolution, aiming at tracking and marking the changes happened between versions of ontologies. In this field, visualization tools should provide distinguish representation symbols, color or pattern to differentiate the old ontologies content (concepts, links, entities, etc.) from the newest.

Motivated by these challenges, in this paper, we introduce a systematic review of existing ontology visualization approaches and tools in order to draw a road map of using ontology visualization implementations in ontology-based management systems. In particular, we start by motivating this review by presenting different application domains that make use of ontology visualization. We further introduce a generic ontology visualization pipeline to be used as a basis for discussing and comparing existing tools. After that, we elaborate a set of visualization tools. The paper also includes a discussion on the challenges and benefits that the field of ontology visualization brings forward. It is hoped that the survey would be helpful both to developers and to users. The rest of the paper is organized as follows: a set of application domains that benefit from ontology visualization is presented in the next section. We then introduce a generic pipeline that guides the ontology visualization in Sect. 3. In Sect. 4, we review ontology visualization tools and techniques. In Sect. 5, we discuss current state of the art solutions and enumerate most common challenges to be solved in our future work.

2 Application Domains

As mentioned before, ontologies are being widely used in different domains, and visualizing the ontology becomes an important step in the whole pipeline within these domains. To motivate the importance of ontology visualization, we summarize its use in some of these application domains.

– **Ontology creation and development** - Developing ontologies is an important aspect of the Semantic Web [19]. For this reason, there exist a number

of tools for ontology development [19], such as Protege [26], SWOOP [32], OntoLingua [15] and OBO-edit [9]. Ontology development tools allow the user to create and/or to modify the ontology by adding new concepts, relations and instances. Those tools may also contain many features like graph visulizor and search and constraint checking capabilities [19].

- **Ontology browsers** - Several tools were developed to browse ontologies. Some of them are dedicated to analyse specific data sets such as agriGO [12], some others are for ontology exploration in general, such as Amigo [7], OLSViz [39] and FLEXViz [13]. Text-based ontology viewer, like Amigo [7], use a folder/subfolder-interface to explore hierarchies [39]. OLSViz [39] and FLEXViz [13], were created to improve the exploration of bio-ontologies. However, OLSViz [39] makes more efficient and intuitive use of the available screen space than FLEXViz [13].
- **Ontology matching** - is the process that identifies and discover corresponding concepts across different ontologies [35]. It plays a crucial role in different shared-data applications, such as data and ontology integration [30]. Due to its importance, many matching algorithms have been proposed and a myriad of matching tools have been developed. Most of these tools provide a way to visualize the whole matching process for different goals [28,37]. One is to enhance and support visual analytic in a semi-automatic matching process [28]. For example, *ENViz* is an approach to integrate data that performs joins enrichment analysis of two type matched datasets. In that tool, the role of ontology visualization is obvious to support user analysis during matching different datasets [28]. Another dimension is to involve the user in the matching process to validate the automatically generated alignment [11]. To sum up, there are different ways of user intervention in the ontology matching process: to select base matchers, to adjust similarity weights, or to validate matching result. All these kinds require an effective way to visualize ontologies to support user intervention.
- **Ontology evolution** - is the process that timely adapts of the ontology due to the arisen changes and the consistent propagation of these changes to dependent artifacts [38]. For example, and based on statistics on BioPortal[1], the gene ontology in one month has about 17 different versions, which indicates a high rate of expanding the ontology. This necessitates the need for tools that support users during the ontology evolution process [27]. An ontology evolution system should have a set of functionalities: among them, showing ontology version, compare different versions, identify conflicts, showing conflicts, etc. Ontology visualization techniques could be used to support the desired functionality [5,27]. For example, *REX* is a tool for discovering evolution in ontology regions by providing an interactive and user-friendly visualization to determine (un)stable regions in large life science ontologies [10]. *CODEX* is a tool that allows identifying semantic changes between two versions of an ontology which users can interactively analyze in multiple ways [22].

[1] http://bioportal.bioontology.org/ontologies/GO.

– **OBDA** - Ontology-based data access (OBDA) is an elegant approach to improve data accessibility in which one ontology (or a set of ontologies) can be used as a mediator between data users and data sources [6,18,25,36]. In the context of *OBDA* systems, visualization can be used in different scenarios. The development of the conceptual layer ontology is a clear scenario where ontology visualization is needed, where the ontology engineer can manage the ontology development and evolving process. Another dimension is the development of an ontology-based visual query system as an extension for the stream temporal one [36].

3 Generic Visualization Pipeline

In order to conduct a good survey and to construct a fair basis for comparing existing ontology visualization tools, a high-level architecture for a generic pipeline for ontology visualization is proposed. Figure ?? depicts the pipeline with three basic steps: ontology parsing, processing, and visual representation.

3.1 Ontology Representation

As mentioned before an ontology is a formal representation of a set of concepts within a domain and their relationships. In designing an ontology language, a tradeoff between the power of expressiveness and the efficiency of reasoning should be considered. As a result, there exist variant ontology languages based on these two criteria [1,21]. For example, the Ontology Web Language (*OWL*) is currently the standard language for the semantic Web and it is compatible with early ontology languages, such as *SHOE* (Simple HTML Ontology Extension), *DAIM + OIL* (DARPA Agent Markup Language + Ontology Interchange Language). However, it is not feasible to satisfy the tradeoff between the expressiveness and the efficiency of reasoning, *OWL* comes with three various formats:

OWL Lite adds the possibility to express definitions and axioms, together with a limited use of properties to define classes. *OWL DL* supports those users who want the maximum expressiveness while retaining good computational properties. *OWL Full* is meant for users who want maximum expressiveness with no computational guarantees.

After reading and parsing an input ontology, the next step is how to internally represent it. Two common representation scheme could be used: *tree-based* or *graph-based* scheme. The first one is used to illustrate super/subclass relationships. Several tools such as *SQuaRE* [3] and *CODEX* [22] adopted this technique. The second one is the node-link diagrams (i.e., graphs), which represents ontologies as a set of interconnected nodes through edges that illustrate ontological entities and the relationships that exist among them. A number of visualization techniques have been used over the tools, such as tree-maps, tree-layouts, fisheye views [16], birdeye views, hyperbolic and 3D hyperbolic layouts. *REX* [10], the ontology evolution tool, as an example, used the fisheye view, since it makes the selected concept in the center, surrounded with its subconcepts. It's useful

when it's the case of displaying a large structures. An ontology visualization tool should be generic and has the ability to handle ontology in different languages and formats. Therefore, a first step in the visualization is *ontology representation*, how the tool reads the input ontology and how it internally represents the ontology to capture its content. To simplify reading input ontologies, several APIs and frameworks have been proposed. Two commons are Apache Jena API[2] and the OWL API[3].

3.2 Processing

As part of the whole ontology visualization pipeline, processing has to support not only the presentation and of ontology components which are: classes (or entity types), relations, instances, and properties (or slots), but also the coarse and fine ontology manipulation. Those operations are fundamentals when dealing with domain expert ontology creation, editing and validating.

Basically, processing step might include zoom-in and zoom-out for locating specific nodes and provide a comprehensive view of the hierarchy level the user is zoomed in or out. Rotation and moving nodes are also the basic operation that should be integrated. However, an ontology visualization tool as we assume, should also contain some deep operations, like ontology alignment and merging. Consequently, matching two ontologies, for example, can be achieved by automatically discovering correspondences between nodes and by graphical mapping interfaces that might assist the process of refining these correspondences. Additionally, merging two ontologies in a new one, translating a query addressed to a source ontology into a query addressed to a target ontology should be mapped into the user interface and then handled graphically. As fine operation, merging two concepts in ontologies means that the user has to determine the scope of overlapping and decide if two concepts or more that are similar.

Verbalization is a part of processing step that should be integrated into the visualization tool. It gives a textual description or details about the selected ontology element from the graphical representation. In order to facilitate the access to ontology content, querying and search related operations should allow highlighting a specific element that user is looking for. An interpreter that parse textual or visual query is part of the processing step. As for the evolution related operations, history browsing, saving and loading of customized ontologies views is required. Consequently, a toolbox managing ontology version should be considered. Hence, our goal is to give an easy graphical way to deal with ontology processing.

3.3 Visual Representation

An ontology is composed of several elements. Visual representation of the ontology is to represent those elements visually and conceptually, in a way that the

[2] https://jena.apache.org/.
[3] http://owlapi.sourceforge.net/.

user can understand the whole hierarchy of the ontology [24]. The visualization method should display all the ontology classes, providing at least their name, in an intelligible manner. Also, it should display the data associated with the ontology which called the instances. The presentation of the "Isa relations" on which the ontology is based is also essential for understanding the inheritance relations between classes. For this reason, the system should at least provide a holistic view of this taxonomy, in a hierarchical representation. Finally, 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.

4 Visualization Tools

This section is devoted to present current ontology visualization tools directed by generic steps in the proposed pipeline.

- **OWLGred**[4] - is implemented as a web based application for ontology visualization [29]. After parsing input data using OWL API, the processing step includes data transformation aiming to retrieve information necessary for visualization. The analysis of axioms (*TBox* and *ABox* elements) aims at constructing UML-class diagram for the representation since most OWL features are mapped to UML concepts (OWL classes to UML classes, datatype properties to class attributes). As for verbalization process, authors of the tool used a Grammatical framework to facilitate the implementation of the CNLs. The user can then select an element from the ontology to generate a CNL verbalization of the corresponding axioms in ACE (Attempto Controlled English). Visual representation of this tool is provided through UML-notations. Generated graphs use an orthogonal layout where the inheritance-defining relations are presented in a hierarchical layout [29] and all other relations "flow" in the direction perpendicular to it. It is based on HTML5 canvas[5] element using KineticJS library where visualization is drawn according to the graph structure, element coordinates, and styles contained in its JSON structure.
- **ProtgVOWL**[6] - are plugins for the ontology editor Protege [31], aiming to give a visual language dedicated not only for human expert, but also for users not familiar with ontologies. After parsing by OWL API, the processing step allows to transforming internal representation of parsed ontologies into the data model required by the Prefuse visualization toolkit[7] [23]. For visual representation, protégéVOWL is based on VOWL specifications which provide a visual language that can be understood by users less familiar with ontologies. Graphical primitives forming the alphabet of the visual language contains circles to depict classes, lines representing to represent property relations, while

[4] http://owlgred.lumii.lv/.
[5] https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API.
[6] http://vowl.visualdataweb.org/webvowl.html.
[7] http://prefuse.org/.

property labels and datatypes are shown in rectangles. VOWL defines also a color scheme for a better distinction between the different elements. Interface of VOWL contains three main layouts which are: VOWL Viewer for displaying the ontology, VOWL Sidebar for giving details about a selected element from the ontology and VOWL Controls for adapting the force directed graph layout.

- **MEMO GRAPH** [17] - is an ontology visualization tool that follows the design-for-all philosophy. A preprocessing step is based on the parsing of OWL/RDF using OWL API. For the processing step, authors integrate a force-directed algorithm to display visualization and provide zoom in-out on nodes of ontologies. Besides, the tool supports the search functionality by keywords based search according to modalities typing and dictation. It supports also node selection to look for information expressed in easy to understand wording rather than the ontology jargon. Enriched by delivering auditory information, visual representation in *MEMO GRAPH* covers all the key elements of the ontology which are classes, instances, data and object properties. Relations between related nodes are represented using labeled links. Nodes of the ontology generated graph are identified by using pictures and labels. The interface of MEMO GRAPH is divided into three parts: The "MEMO GRAPH Viewer" displaying the ontology visualization as a graph, the "MEMO GRAPH details" listing details about a selected node, and the "MEMO GRAPH search" providing a key word search option.
- **OntoSphere**[8] - is a 3D ontology visualization tool developed in Java as a stand-alone application [4]. For the preprocessing step Jena API is chosen, consequently the tool can allow to easily load and manage ontologies and taxonomies written either in RDF, RDFs, DAML, OWL or N-triple. The processing step has many manipulation features provided for users like the rotation, zoom, object selection, etc. It relies on browsing ontology as well as updating it by adding new concepts and new relations. The visual representation is based mainly on a 3D representation on which authors worked on increasing the number of dimensions (colors, shapes, transparency, etc.). It exploits different scene managers as *RootFocus*, *TreeFocus* and *ConceptFocus* that present and organize the information on the screen. The RootFocus Scene presents a big earth-like sphere bearing on its surface a collection of concepts represented as small spheres. The TreeFocus Scene shows the subtree displaying the hierarchical structure as well as semantic relations between classes. The ConceptFocus Scene depicts all the available information about a single concept, at the highest possible level of detail.
- **REX**[9] - Region Evolution Explorer is based on a region discovery algorithm used mainly to determine differently changing regions for periodically updated ontologies and to interactively explore the changing intensity of those regions [10]. The tool handle supports the import of ontologies in different

[8] http://ontosphere3d.sourceforge.net/userGuide.html.
[9] https://dbs.uni-leipzig.de/de/research/projects/evolution_of_ontologies_and_mappings.

formats such as OWL and OBO. Based on the resulted parsed ontologies, the processing step aims at first computing differences between two versions to determine changes. It then propagates change costs within the is-a hierarchy of the ontology and transfers these costs from the first to the last considered version. Based on computed change intensities differently evolving ontology regions are discovered. Visual representation of ontology is a graph where nodes represent the concepts and edges represent the relations. The color feature was used in this tool to describe the concepts change intensity. Quantitative changes are also visualized in a graphical way as to curve of frequencies changes.

- **CODEX** - is an ontology evolution tool, used to obtain knowledge about the evolving ontology [22]. It provides support for determining complex changes between two versions of ontologies. The tool deals with OBO and OWL ontologies and flat files formats in the preprocessing step. The processing offered by the tool, supports changes such merging, splitting, adding and moving subgraph. Others functions are implemented to facilitate exploration of changes. The user interface contains multiple views. A high-level view provides statistics about the number of relations and concepts in two ontology versions as well as the number of changes between the versions. After selecting a change, the changes can be explored in a tree-like manner. Hence, customizable overview statistics such frequent updates nodes, Tag clouds to visualize changes and modified content Tree-based change explorer and Impact analysis.
- **Other tools** - *OWLeasyViz* [8] is based on an approach that combines a textual and graphical representation of OWL ontologies, where the textual representation presents class, data properties and object properties in a three-column table and the graphical representation which contains the graphical representation of the ontology, where we find child nodes which are visualized inside their parents, with smaller size. Also, nodes are represented by many shapes according to their hierarchy. Leaf nodes are visualized as rounded rectangles, while parent nodes as elliptical shapes. Also in this tool, they used zoomable technique, and also Searching and filtering mechanisms. They exploit the visualization strategies used in Grokker, which is a generic system for displaying of knowledge maps.

SQuaRE [3] is a query and R2RML mappings environment. It provides a visual editor for creating and managing R2RML mappings as well as for creating and executing SPARQL queries. It contains two sides, the client side, and the server side. The client side consists of modules working on a client's machine in a user's web browser and provides a presentation layer. The server side contains the data source, DBMS manager, ontology handler for OWL ontology processing, and SPARQL query executor. SQuaRE applies a set of tools to handle OWL ontologies, relational data and SPARQL queries. For the

visualization of ontologies, it uses OWL-API and javascript libraries like *AngularJS*[10], *jQuery*[11], *Cytoscape.js* with *CoSE Bilkent* layout, *jsPlumb*[12] and *jsTree*[13].

ViZiQuer is a tool for data analysis query definition and translation into SPARQL [40]. The ViziQuer notation is based on UML class diagrams. A query in the ViziQuer notation is a graph of class boxes connected with association links. Visualizer tool for the Agreement Maker system[14] [14] focuses on ontology matching visualization. They add to Agreement Maker tool, which is an ontology matching tool, the capability to visualize the results of matching large ontologies with a user interface that supports navigation and search of the ontologies.

5 Discussion and Future Directions

Keeping in mind the pipeline that we described for the ontology visualization tools, the overview of most existing works shows that the preprocessing step is fundamentals in the whole system and consists on primitives designed for parsing ontologies even in OWL or RDF based on JENA or OWL APIs. This preprocess in sometimes extended by some mapping primitives to generate intermediate data representations aiming at facilitating the processing step.

Actually, the main operations enabled in most of the current systems are limited to some transformation functions aiming at generating graphs except a few number of tools focusing on ontologies evolution. In that case, most functions are related to matching between two versions of ontology and aiming at highlighting the change in general by intensity color and distinguished links. Some quantitative measures to visualize the frequency of changes are implemented in these tools. Search functions are also provided based on keyword textual query.

Within the last step of the pipeline, the visual representation is handled in most of the existing tools by a hierarchical representation based on graphs (circle, line, color, etc.) and less on such UML language notation. A particular attention is given to 3D visualization based tools which add a new dimension to get more flexibility for ontology content visualization. The main objective is to facilitate navigation in the ontology and providing multi level view options.

Although the interest given to ontology visualization expressed in recent publications and already developed tools, we think that the goal of building an ontology tool for unfamiliar user is far to be reached. The challenges arisen can be summarized in the following points:

– Dealing with very large ontologies since most of the tools suffer from overlapped links and labels due to lack of compact visualizations widgets.

[10] https://angularjs.org/.
[11] https://jquery.com/.
[12] https://jsplumbtoolkit.com/.
[13] https://www.jstree.com/.
[14] https://github.com/AgreementMakerLight/AML-Jar.

- Looking for optimization techniques aiming at accelerating the process of large ontology even for parsing and visualization.
- Most tools are designed only for ontology experts and lack for generality.
- Few represent all key elements of the ontology (i.e., classes, instances, datatype properties and object properties) and show data properties as labeled links.
- Basics processing integrated into most of the tools need to be extended by more advanced operations such as merging nodes according their similarities (matching). Those primitives when implemented can provide a real support for domain expert construction and editing ontologies.
- Implement visual language representation for ontology based on standards is a step for building common useful tools.
- Integrate natural language processing techniques to the verbalization process. This should provide the semantic description of ontology elements and make ontology content common for expert and non-expert users.
- Working on dynamic visualization approach that could be adapted to the content and size of input ontology.
- Including ontology evolution related primitives to manage changes and version.

As for continuity of our work on ontology matching, alignment, and merging, we are currently working on an approach that provides at first a guideline for the development of ontology visualization tool. This guideline considers all the above challenges and aims at designing and implementing tools that can easily deal with large ontologies and generalize its use by nonexpert users.

Acknowledgments. This work was supported by the DAAD funding through the BioDialog project.

References

1. Antoniou, G., Franconi, E., van Harmelen, F.: Introduction to semantic web ontology languages. In: First International Summer School on Reasoning Web, Tutorial Lectures, pp. 1–21 (2005)
2. Aurisano, J., Nanavaty, A., Cruz, I.F.: Visual analytics for ontology matching using multi-linked views. In: Proceedings of the International Workshop on Visualizations and User Interfaces for Ontologies and Linked Data Co-located with 14th International Semantic Web Conference (ISWC 2015), p. 25 (2015)
3. Blinkiewicz, M., Bąk, J.: SQuaRE: a visual approach for ontology-based data access. In: LNCS (including subseries LNAI and LNBI), vol. 10055, pp. 47–55. Springer, Cham (2016)
4. Bosca, A., Bonino, D.: OntoSphere3D: a multidimensional visualization tool for ontologies. In: 17th International Workshop on Database and Expert Systems Applications (DEXA 2006), pp. 339–343 (2006)
5. Burch, M., Lohmann, S.: Visualizing the evolution of ontologies: a dynamic graph perspective. In: International Workshop on Visualizations and User Interfaces for Ontologies and Linked Data Co-located with ISWC 2015, p. 69 (2015)

6. Calvanese, D., Cogrel, B., Komla-Ebri, S., Kontchakov, R., Lanti, D., Rezk, M., Rodriguez-Muro, M., Xiao, G.: Ontop: answering SPARQL queries over relational databases. Semant. Web **8**(3), 471–487 (2017)
7. Carbon, S., Ireland, A., Mungall, C.J., Shu, S., Marshall, B., Lewis, S.: AmiGO: online access to ontology and annotation data. Bioinformatics **25**(2), 288–289 (2009)
8. Catenazzi, N., Sommaruga, L.: Generic environments for knowledge management and visualization. J. Ambient Intell. Hum. Comput. **4**(1), 99–108 (2013)
9. Catenazzi, N., Sommaruga, L., Mazza, R.: User-friendly ontology editing and visualization tools: the OWLeasyViz approach. In: 13th International Conference Information Visualisation, pp. 283–288 (2009)
10. Christen, V., Hartung, M., Groß, A.: Region evolution explorer - a tool for discovering evolution trends in ontology regions. J. Biomed. Semant. **6**, 26 (2015)
11. Dragisic, Z., Ivanova, V., Lambrix, P., Faria, D., Jiménez-Ruiz, E., Pesquita, C.: User validation in ontology alignment. In: 15th International Semantic Web Conference (ISWC 2016), pp. 200–217 (2016)
12. Du, Z., Zhou, X., Ling, Y., Zhang, Z., Su, Z.: agriGO: a GO analysis toolkit for the agricultural community. Nucleic Acids Res. **38**(suppl 2), W64–W70 (2010)
13. Falconer, S.M., Callendar, C., Storey, M.-A.: FLEXVIZ: visualizing biomedical ontologies on the web. In: International Conference on Biomedical Ontology, Software Demonstration, Buffalo, p. 1 (2009)
14. Faria, D., Martins, C., Nanavaty, A., Taheri, A., Pesquita, C., Santos, E., Cruz, I.F., Couto, F.M.: Agreementmakerlight results for OAEI 2014. In: Proceedings of the 9th International Workshop on Ontology Matching Collocated with the 13th International Semantic Web Conference (ISWC 2014), Riva del Garda, pp. 105–112, 20 October 2014
15. Farquhar, A., Fikes, R., Rice, J.: The Ontolingua server a tool for collaborative ontology construction. Int. J. Hum. Comput. Stud. **46**(6), 707–727 (1997)
16. Furnas, G.W.: Generalized fisheye views. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 16–23, April 1986
17. Ghorbel, F., Ellouze, N., Métais, E., Hamdi, F., Gargouri, F., Herradi, N.: MEMO GRAPH: an ontology visualization tool for everyone. In: 20th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES-2016), pp. 265–274 (2016)
18. Giese, M., Soylu, A., Vega-Gorgojo, G., Waaler, A., Haase, P., Jiménez-Ruiz, E., Lanti, D., Rezk, M., Xiao, G., Özçep, Ö.L., Rosati, R.: Optique: zooming in on big data. IEEE Comput. **48**(3), 60–67 (2015)
19. Grover, P., Chawla, S.: Ontology creation and development model. **6**(1991), 3878–3883 (2014). ijircce.com
20. Guarino, N., Oberle, D., Staab, S.: What is an ontology? In: Handbook on Ontologies, pp. 1–17 (2009)
21. Gutierrez-Pulido, J.R., Ruiz, M.A.G., Herrera, R., Cabello, E., Legrand, S., Elliman, D.: Ontology languages for the semantic web: a never completely updated review. Knowl. Based Syst. **19**(7), 489–497 (2006)
22. Hartung, M., Groß, A., Rahm, E.: CODEX: exploration of semantic changes between ontology versions. Bioinformatics **28**(6), 895–896 (2012)
23. Heer, J., Card, S.K., Landay, J.A.: prefuse: a toolkit for interactive information visualization. In: Proceedings of the 2005 Conference on Human Factors in Computing Systems (CHI 2005), pp. 421–430 (2005)
24. Katifori, A., Halatsis, C., Lepouras, G., Vassilakis, C., Giannopoulou, E.G.: Ontology visualization methods - a survey. ACM Comput. Surv. **39**(4), 10 (2007)

25. Kharlamov, E., Hovland, D., Jiménez-Ruiz, E., Lanti, D., Lie, H., Pinkel, C., Rezk, M., Skjæveland, M.G., Thorstensen, E., Xiao, G., Zheleznyakov, D., Horrocks, I.: Ontology based access to exploration data at statoil. In: 14th International Semantic Web Conference (ISWC 2015), pp. 93–112 (2015)
26. Klein, M., Fensel, D., Kiryakov, A., Ognyanov, D.: Ontology versioning and change detection on the web. In: Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web, pp. 197–212 (2002)
27. Lambrix, P., Dragisic, Z., Ivanova, V., Anslow, C.: Visualization for ontology evolution. In: Second International Workshop on Visualization and Interaction for Ontologies and Linked Data co-located with the 15th International Semantic Web Conference, VOILA@ISWC 2016, pp. 54–67 (2016)
28. Li, Y., Stroe, C., Cruz, I.F.: Interactive visualization of large ontology matching results. In: International Workshop on Visualizations and User Interfaces for Ontologies and Linked Data Co-located with ISWC 2015, p. 37 (2015)
29. Liepins, R., Grasmanis, M., Bojars, U.: Owlgred ontology visualizer. In: Proceedings of the ISWC Developers Workshop 2014, pp. 37–42 (2014)
30. Livingston, K.M., Bada, M., Baumgartner Jr., W.A., Hunter, L.E.: KaBOB: ontology-based semantic integration of biomedical databases. BMC Bioinform. **16**, 126:1–126:21 (2015)
31. Lohmann, S., Negru, S., Haag, F., Ertl, T.: Visualizing ontologies with VOWL. Semant. Web **7**(4), 399–419 (2016)
32. Malik, S.K., Rizvi, S.: Ontology design towards web intelligence: a sports complex ontology case study. In: Fourth International Conference on Computational Aspects of Social Networks (CASoN), pp. 366–371 (2012)
33. Negru, S., Lohmann, S.: A visual notation for the integrated representation of OWL ontologies. In: 9th International Conference on Web Information Systems and Technologies (WEBIST 2013), pp. 308–315 (2013)
34. Noy, N.F., Klein, M.C.A.: Ontology evolution: not the same as schema evolution. Knowl. Inf. Syst. **6**(4), 428–440 (2004)
35. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. IEEE Trans. Knowl. Data Eng. **25**(1), 158–176 (2013)
36. Soylu, A., Giese, M., Jiménez-Ruiz, E., Kharlamov, E., Zheleznyakov, D., Horrocks, I.: Ontology-based end-user visual query formulation: why, what, who, how, and which? Univ. Access Inf. Soc. **16**(2), 435–467 (2017)
37. Steinfeld, I., Navon, R., Creech, M.L., Yakhini, Z., Tsalenko, A.: ENViz: a cytoscape app for integrated statistical analysis and visualization of sample-matched data with multiple data types. Bioinformatics **31**(10), 1683–1685 (2015)
38. Stojanovic, L.: Methods and tools for ontology evolution. Ph.D. thesis, Karlsruhe Institute of Technology, Germany (2004)
39. Vercruysse, S., Venkatesan, A., Kuiper, M.: OLSVis: an animated, interactive visual browser for bio-ontologies. BMC Bioinform. **13**(1), 116 (2012)
40. Zviedris, M., Barzdins, G.: Viziquer: a tool to explore and query SPARQL endpoints. In: 8th Extended Semantic Web Conference (ESWC), pp. 441–445 (2011)