Chapter 16
Visualizing and Navigating Ontologies with KC-Viz

Enrico Motta, Silvio Peroni, José Manuel Gómez-Pérez, Mathieu d'Aquin, and Ning Li

Abstract There is empirical evidence that current user interfaces for ontology engineering are still inadequate in their ability to reduce task complexity for users, especially non-expert ones. Here we present a novel tool for visualizing and navigating ontologies, *KC-Viz*, which exploits an innovative ontology summarization method to support a "middle-out ontology browsing" approach, where it becomes possible to navigate ontologies starting from the most information-rich nodes (i.e., *key concepts*). This approach is similar to map-based visualization and navigation in geographical information systems, where, e.g., major cities are displayed more prominently than others, depending on the current level of granularity. Building on its powerful and empirically validated ontology summarization algorithm, KC-Viz provides a rich set of navigation and visualization mechanisms, including flexible *zooming* into and *hiding* of specific parts of an ontology, visualization of the most *salient* nodes, *history* browsing, saving and loading of customized ontology views, as well as essential interface support, such as graphical zooming, font manipulation, tree layout customization, and other functionalities.

E. Motta (✉) • M. d'Aquin • N. Li
Knowledge Media Institute (KMi), The Open University, Walton Hall, Milton Keynes, MK7 6AA, UK
e-mail: e.motta@open.ac.uk; m.daquin@open.ac.uk; N.Li@open.ac.uk

S. Peroni
Department of Computer Science, University of Bologna, Bologna, Italy
e-mail: speroni@cs.unibo.it

J.M. Gómez-Pérez
Intelligent Software Components (iSOCO), S.A. Avda. del Partenón, 16-18, 28042 Madrid, Spain
e-mail: jmgomez@isoco.com

M.C. Suárez-Figueroa et al. (eds.), *Ontology Engineering in a Networked World*,
DOI 10.1007/978-3-642-24794-1_16,

16.1 Introduction

A key component of the Semantic Web is provided by the large number of ontologies available online. Given such large-scale availability of ontologies, ontology reuse is becoming more common, and tools, such as the Watson plugin for the NeOn Toolkit (d'Aquin et al. 2008), are now available, which facilitate the task of locating and directly reusing ontologies or ontology fragments. In this reuse-centric context, it is highly desirable to have mechanisms that can efficiently help users in making sense of the content of an ontology, e.g., in the context of having to make a decision about whether an ontology retrieved online is suitable for a particular set of requirements. However, the empirical studies carried out in the NeOn project (Dzbor et al. 2006) have shown that the visualization and navigation facilities available in today's ontology engineering environments do not necessarily provide effective support for making sense of and effectively exploring ontologies, and often end up hindering rather than helping users. These studies show that this is a problem, especially for non-expert users.

To address this issue, we have developed a novel tool for visualizing and navigating ontologies, called *KC-Viz*, which has been realized as a plugin for the NeOn Toolkit. KC-Viz exploits automatically created ontology summaries, based on the idea of *key concepts* (Peroni et al. 2008), to facilitate the task of making sense of large ontologies. In addition, it also provides a rich set of navigation and visualization mechanisms, including flexible *zooming* into and *hiding* of specific parts of an ontology, visualization of the most *salient* nodes, *history* browsing, saving and loading of customized *ontology views*, as well as essential interface customization support, such as graphical zooming, font manipulation, tree layout customization, and other functionalities.

In this chapter, we present a description of the main functionalities provided by KC-Viz, and we show how it attempts to address some of the limitations of current tools for ontology engineering.

16.2 Limitations of Top-Down Approaches to Navigating Ontologies

16.2.1 Ontology Sensemaking

Throughout this chapter, we will use as an illustrative example a version (v2.4) of the SmartProducts ontology[1], which is being developed in the course of the EU-funded SmartProducts project[2]. The aim of this ontology is to support the

[1] This network of ontologies can be downloaded from http://projects.kmi.open.ac.uk/smartproducts/ontologies/SP_v2_4.zip

[2] http://www.smartproducts-project.eu

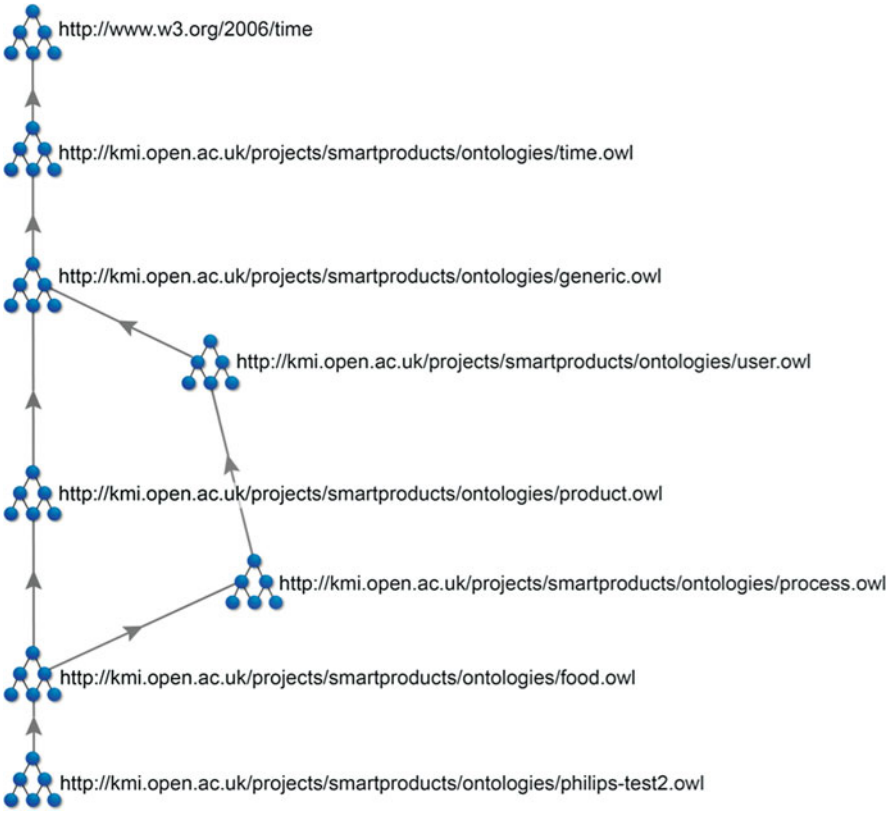

Fig. 16.1 Import relations in the SmartProducts network of ontologies

specification of smart devices, able to engage *proactively* in cooperative problem solving with other devices.

As shown in Fig. 16.1, the SmartProducts ontology is not a monolithic one but comprises a number of sub-ontologies, which define different notions, such as time, users, processes, products, etc. The project addresses three different test cases in the aerospace, car, and consumer appliances industries, and in particular, Fig. 16.1 shows the network of ontologies used to characterize the latter scenario, which we refer to as "Smart Kitchen." The structure of the network is highly reusable, with the top six nodes (i.e., ontologies) being shared across the three test cases, while the bottom two ontologies are specific to the Smart Kitchen application.

Like most other ontology engineering environments available today, the NeOn Toolkit provides an "*Ontology Navigator*" window, which supports ontology navigation using the classic top-down file system model, where clicking on a folder reveals its contents. In the case of an ontology engineering tool, the folder metaphor is used "to open up" a class, to reveal its sub-classes.

As discussed in (Katifori et al. 2007), this style of interface has several advantages, including its familiarity to users and the ability to support a systematic exploration of an ontology. For these reasons, it is more or less ubiquitous in ontology engineering toolkits and also tends to perform well in evaluations (Katifori et al. 2007). Nevertheless, it also exhibits some important limitations, including its inability to show role relations and "to support tasks related to the general ontology structure" (Katifori et al. 2007).

In this chapter, we will indeed focus on this category of tasks, which we will refer to informally as *ontology sensemaking* tasks. More specifically, we will use the term "sensemaking" to refer to the construction of a mental model of an ontology, which encompasses the ontology as a whole and is sufficient for a user to make a decision (for example) on whether an ontology is suitable for a particular application or whether it covers certain areas of interest to the required extent, with respect to user-specific criteria. In sum, the emphasis here will be less on supporting tasks which require understanding a particular detail of the ontology than on supporting tasks which require developing a "global" model of an ontology, at a certain level of abstraction. In addition, although KC-Viz also support the visualization of non-taxonomic (i.e., domain) relations, here we will focus the discussion almost exclusively on the navigation and visualization of taxonomies, on the basis that developing an understanding of the overall taxonomic structure of an ontology is an essential part of the sensemaking process.

16.2.2 Example: Using the Ontology Navigator for Sensemaking

Figure 16.2 shows a snapshot of the Ontology Navigator in the NeOn Toolkit, after we have clicked on the most specific ontology shown in Fig. 16.1 (see Sect. 16.2.1), which is called Philips-Test2. As shown in the figure, clicking on the folder Classes reveals the four topmost classes in the SmartProducts network of ontologies[3], making explicit the top-level structure of the ontology. However, while this initial visualization is useful to allow the user to understand the organization of the ontology at the highest level of abstraction, it is not yet comprehensive enough to allow the user to develop an overall model of the ontology in sufficient detail. In particular, without further exploration, it is not yet possible to achieve the following objectives, which are essential to the sensemaking process:

[3] This is because the relevant preference in the NeOn Toolkit is set to display all inherited classes, thus allowing us to browse the complete structure of the SmartProducts network of ontologies. Alternatively, we can choose to see only definitions local to the Philips-Test2 ontology, by deselecting the option "Show Imported Axioms."

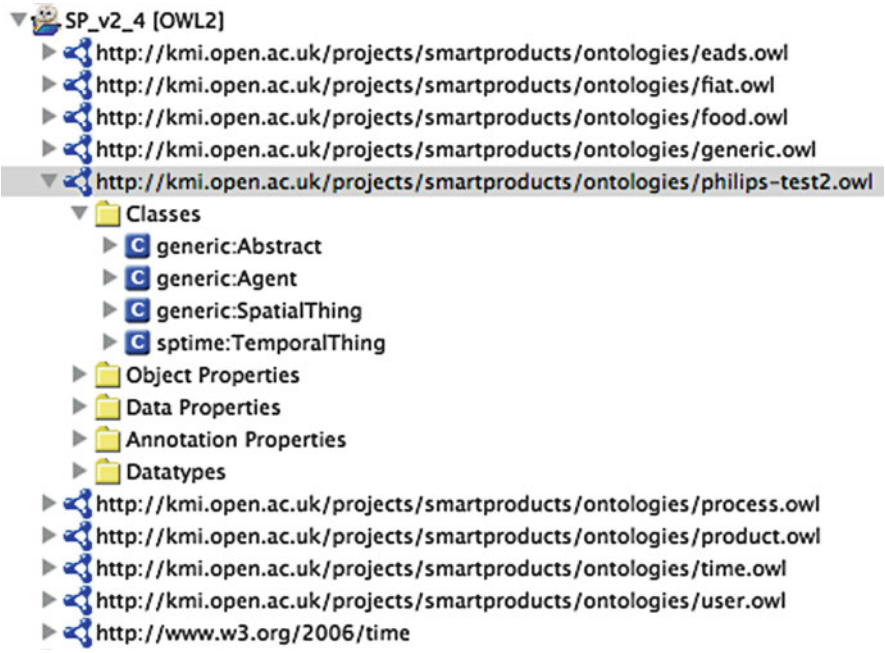

Fig. 16.2 Navigation through a file system metaphor

- Understanding the overall *size* and *shape* of the ontology. By "size" here we mean, given a node in the ontology[4], the total number of its direct and indirect sub-classes, while by "shape" we refer to an indication of the organization of the sub-classes. For instance, an ontology (or part of it) can have a *horizontal* (i.e., many sub-classes and few levels of depth), or a *vertical* (i.e., many inheritance levels and only a few sub-classes at each level) shape (Tartir et al. 2005). Understanding the shape of an ontology (or part of it) also means to understand whether it is *balanced*, indicating that all parts of the (sub-)ontology in question have been developed to a similar extent, or *unbalanced*, possibly indicating that some parts of the (sub-)ontology are less developed than others.
- Identifying the main components of the ontology and the typical *exemplars* of these components. For instance, from Fig. 16.2 (see Sect. 16.2.1), we understand that the SmartProducts ontology talks about spatial entities, a highly generic (and therefore not-so-informative) concept, but the display fails to tell us which kind of spatial entities the ontology primarily focuses on. Given that, for what we know, the sub-tree under class SpatialThing may contain dozens of sub-classes, it would be useful to have tools that could highlight to us the main spatial entities

[4] If the node is owl:Thing, then we are talking about the size of the whole ontology, otherwise the size of a particular subtree.

covered by the ontology (i.e., the *exemplars*), without the need for extensive exploration. In this case, this would require informing us that almost 50% of the sub-tree under SpatialThing concerns food-related notions. Informative exemplars can also help the user to predict the siblings of the class (i.e., the exemplar) in question, thus playing a summarization role not just with respect to its sub-tree, but also with respect to its siblings.

At this point, the reader may argue that what is needed is simply to explore the structure in more depth, by clicking on the top four classes, to open up the next level of detail. Figure 16.3 shows what happens when we do so and we click on all four level 1 classes. Thirty-eight classes are now displayed, making the picture rather complicated for the user. In addition, we are still none the wiser about which node we should further explore, which parts of the ontologies are developed more in detail, etc. And continuing to open up these nodes will simply bring more information on the screen, making it even more difficult for the user to develop a quick conceptual model of the ontology. Of course, the reader can also point out that part of the problem is the relative lack of structure underneath class Abstract, which contains 22 direct sub-classes. And indeed, a better organization of the sub-tree underneath class Abstract is obviously needed. However, it is also fair to say that the purpose of visualization and navigation tools is not simply to support navigation in relatively small, nicely organized ontologies. More importantly, they also need to help the user in making sense of and effectively explore large and possibly messy ontologies.

The brief and informal analysis shown here is consistent with the findings uncovered in more extensive empirical studies, such as (Dzbor et al. 2006), which highlight the problems users encounter when using rigid top-down navigation tools. These problems include:

- *Poor efficiency and effectiveness*. To open up the display shown in Fig. 16.3 has required six mouse clicks, and we still have a relatively poor understanding of the content of the ontology.
- *Lack of control when zooming on a particular node*. When clicking on a node, the user always opens up all the direct sub-classes. There is no way to control the number of sub-classes shown, or to open up more than one level with one mouse click.
- *No abstraction or saliency mechanisms*. The system has no way to automatically hide nodes which are deemed not important (i.e., *salient*) according to some criterion, and conversely, it is not able to bring to the attention of the user highly important nodes, again with respect to some user criterion.

It is also important to emphasize that such problems are less associated with the file system browsing metaphor than with the generic top-down navigation approach. For instance, ontology engineering toolkits such as TopBraid Composer[5]

[5] http://www.topquadrant.com/products/TB_Composer.html

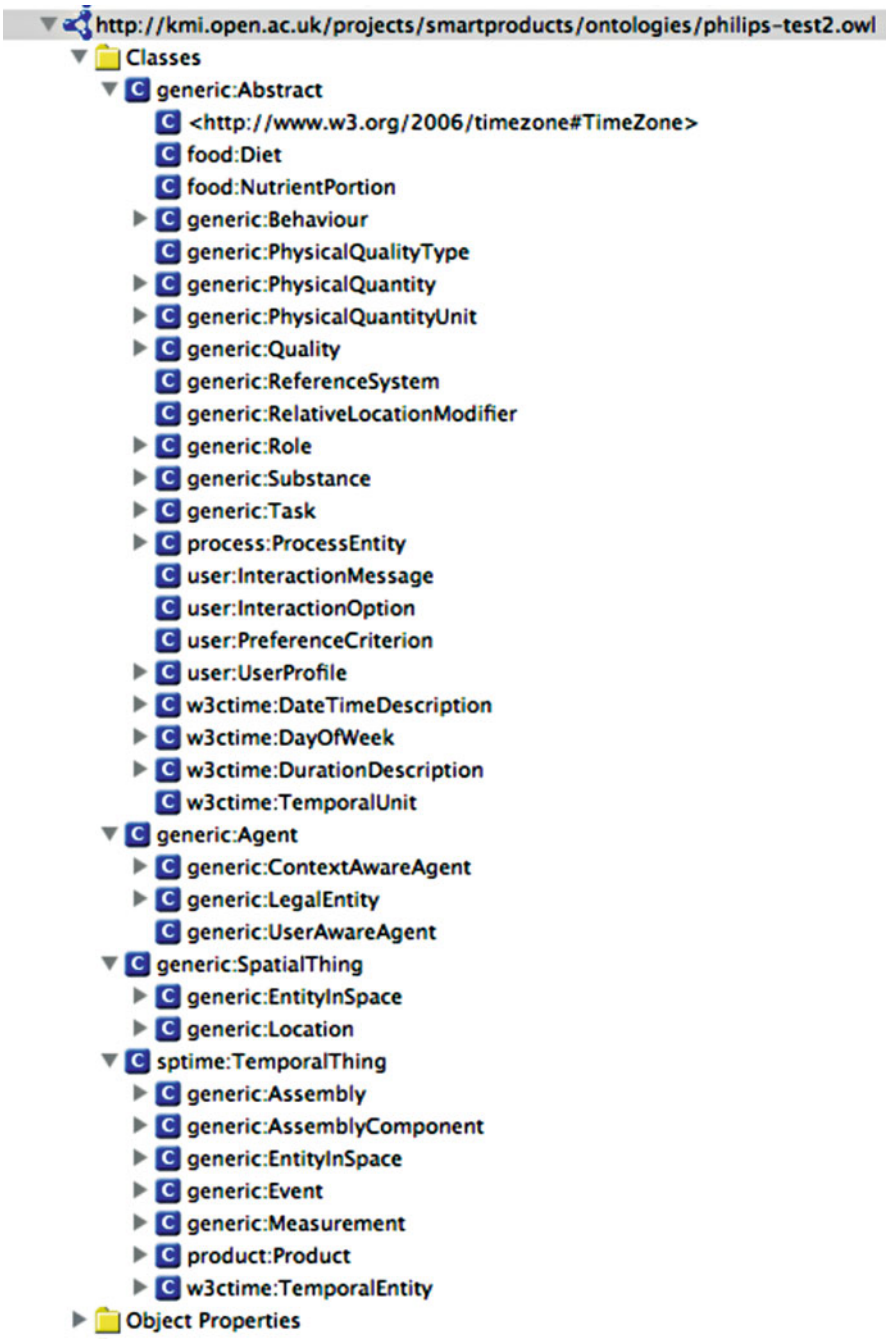

Fig. 16.3 Exploring level 2 classes through the Ontology Navigator

provide graphic tools which also implement such top-down navigation and not surprisingly suffer from the same problems. Indeed, it can be argued that a graphical interface for top-down navigation typically performs worse than a file system model, primarily because the latter usually provides a much more compact representation – i.e., showing the 38 classes in a graphical tree representation will require a much larger display area, thus making it even more complex for a user to make sense of it (Plaisant et al. 2002).

KC-Viz is an ontology visualization and navigation system, which has been designed to address the issues highlighted here, by providing a rich set of navigation and visualization mechanisms, which include flexible *zooming* into and *hiding* of specific parts of an ontology, the ability to identify the most important concepts in an ontology, according to empirically validated criteria, as well as a plethora of other mechanisms to facilitate sensemaking and exploration of ontologies. As already mentioned, a key aspect of KC-Viz is its reliance on a key concepts extraction algorithm, which allows KC-Viz to produce the kind of ontology summaries that human experts are able to produce. Hence, in what follows we will first describe the key concept extraction algorithm used by KC-Viz, before providing an overview of its functionalities.

16.3 Key Concept Extraction

Informally, key concepts can be seen as the best descriptors of an ontology, i.e., information-rich concepts, which are most effective in summarizing what an ontology is about. In (Peroni et al. 2008), we considered a number of criteria to identify the key concepts in an ontology. In particular, we use the notion of natural category (Rosch 1978), to identify concepts that are information-rich in a psycholinguistic sense. This notion is approximated by means of two operational measures: name simplicity, which favors concepts that are labeled with simple names; and basic level, which measures how "central" a concept is in the taxonomy of an ontology. Two other criteria are drawn from the topology of an ontology: the notion of *density* highlights concepts which are information-rich in a formal knowledge representation sense, i.e., they have been richly characterized with properties and taxonomic relationships, while the notion of *coverage* is used to ensure that no important part of the ontology is neglected, by maximizing the coverage of the ontology with respect to its taxonomic relationships. Finally, the notion of *popularity*, drawn from lexical statistics, is introduced as a criterion to identify concepts that are likely to be most familiar to users. The *density* and *popularity* criteria are both decomposed in two sub-criteria: *global* and *local density*, and *global* and *local popularity*, respectively. While the global measures are normalized with respect to all the concepts in the ontology, the local ones consider the relative density or popularity of a concept with respect to its surrounding concepts. The aim here is to ensure that "locally significant" concepts get a high score, even though they may not rank too highly with respect to global measures. Each of these seven criteria produces a score for each

concept in the ontology, and the final score assigned to a concept is a weighted sum of the scores resulting from individual criteria. As described in (Peroni et al. 2008), which provides a detailed account of our algorithm, KCE, and a formal definition of the criteria it employs (i.e., density, coverage, popularity, etc.), our approach has been shown to produce ontology summaries that correlate significantly with those produced by human experts.

16.4 Overview of KC-Viz

16.4.1 Initial Visualization of an Ontology with KC-Viz

Normally, a KC-Viz session[6] begins by generating an initial summary of an ontology, to get an initial "gestalt" impression of the ontology. This can be achieved in a number of different ways, most obviously by (1) selecting the ontology in question in the "Ontology Navigator" tab of the NeOn Toolkit, (2) opening up a menu of options by right clicking on the selected ontology, and then (3) choosing Visualize Ontology ➔ Visualize Key Concepts, through a sequence of menus. Figure 16.4 shows the result obtained after performing this operation on the ontology Philips-Test2[7], the most specific node in the SmartProducts network of ontologies[8]. As shown in the figure, we have now obtained an initial visualization of the network of ontologies, which includes concepts at different levels in the class hierarchy. This specific visualization includes 16 concepts because we have set the size of our ontology summary to 15, and the algorithm has automatically added the most generic concept, owl:Thing, to ensure that the visualization displays a connected graph. If we wish to display more or less succinct graphs, we can do so by changing the size of the ontology summary. The solid gray arrows in Fig. 16.4 indicate direct rdfs:subClassOf links, while the dotted green arrows indicate indirect rdfs:subClassOf links. As shown in the figure, by hovering the mouse over an indirect rdfs:subClassOf links, we can see the chain of rdfs:subClassOf relations, summarized by the indirect link.

Another important piece of information provided by KC-Viz is the size of the tree under a particular class, which is indicated by a pair of integers, indicating the

[6] All the examples in this paper have been generated using version 2.5 of the NeOn Toolkit and KC-Viz v1.3.0.

[7] It is important to point out that while Fig. 16.4 and later figures show exactly the concepts returned by KC-Viz, for the sake of readability we have, when appropriate, manually rearranged the layout, to try and minimize the compression caused by the physical size of this document. This is needed primarily because KC-Viz displays assume a landscape orientation, while this article is formatted according to a portrait orientation.

[8] Crucially, the option "Ontology summary considers also imported ontology" must be enabled in the KC-Viz preferences, otherwise only a summary of the concepts local to the Philips-Test2 ontology will be generated.

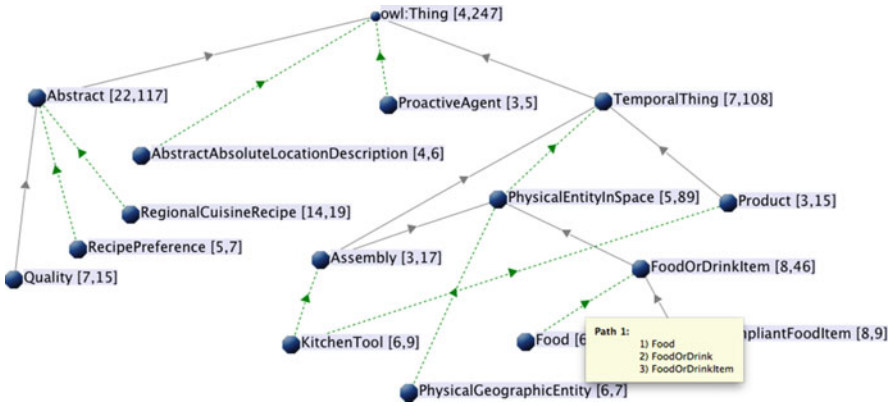

Fig. 16.4 Initial visualization of the SmartProducts ontologies

number of direct and indirect sub-classes. For instance, Fig. 16.4 tells us that class Abstract has 22 direct sub-classes and 117 indirect ones.

Although more exploration is obviously needed to get a thorough understanding of the contents of the SmartProducts ontology network, it can be argued that as a first step, the visualization shown in Fig. 16.4 already provides a rather effective starting point for the ontology sensemaking process. In particular, looking at the visualization, we can already reach a number of conclusions about the ontology, only one of which (the first one) could be concluded after opening up class owl: Thing in the ontology navigator – see Fig. 16.2, Sect. 16.2.1. For instance, we now understand that:

- The network of ontologies contains four top-level classes (i.e., classes directly linked to owl:Thing) – however, only two of them are displayed (Abstract and TemporalThing) in the initial summary.
- The ontology contains a lot of information about food – e.g., the tree under class FoodOrDrinkItem contains 46 classes.
- Key distinctions include time (TemporalThing) and space (PhysicalEntityInSpace). However, there are also temporal things, which are not physical entities in space. We can deduce this because the visualization tells us that TemporalThing has 108 sub-classes, while PhysicalEntityInSpace has 89.
- Class Abstract has a lot of sub-classes (117), but it may be relatively poorly structured, having 22 direct sub-classes.

More importantly, this initial visualization provides a much better structure for further exploration, than the rigid top-down navigation, which we illustrated in Sect. 16.2. In particular, on the basis of this initial snapshot, we can identify key "gaps" that we need to fill, in order to get a complete picture of the ontology. For instance, we may want to explore:

- The sub-tree under class Abstract to get a better understanding of this part of the ontology. As discussed in Sect. 16.2, because of the relatively poor structure of this part of the ontology, better control of the navigation process than that provided by the Ontology Navigator will be needed, in order to be able to explore this part of the ontology effectively.
- The sub-tree under FoodOrDrinkItem, as this is clearly a rich part of the ontology.
- The sub-tree under PhysicalEntityInSpace, which appears to encompass both location-related notions and device-related ones (Assembly).
- Which sub-classes of TemporalThing are not also sub-classes of PhysicalEntityInSpace.
- What kinds of agents are modeled by this ontology.
- Why products are not physical entities in space.
- Others.

In sum, the claim here is that the key concept extraction algorithm used by KC-Viz, together with the degree of control that we get over it (size of summaries and whether or not to consider imported axioms), allows the effective generation of initial ontology snapshots, which helps the user in forming an initial idea of what an ontology is about. In what follows, we show how the flexible support for exploration provided by KC-Viz capitalizes on this initial summary to facilitate effective ontology navigation and sensemaking.

16.4.2 Exploring Ontologies with KC-Viz

Let us consider our first task: to get a better understanding of the sub-tree under class Abstract. We have already seen that a rigid top-down approach does not work very well here, in particular because class Abstract contains many direct sub-classes. So, let us try exploring with KC-Viz.

If we click right on a class displayed in KC-Viz, in this case, Abstract, we obtain a menu which includes options for inspecting, expanding, and hiding a class. If we select "Expand," the menu shown in Fig. 16.5 pops up, which provides a rich set of options for exploring the sub-tree under class Abstract. In particular, the following four options for customizing the expansion algorithm are presented to the user:

- Whether to explore following taxonomic relations, other relations (through domain and range), or any combination of these.
- Whether or not to make use of the ontology summarization algorithm, which in this case will be applied only to the sub-tree of class Abstract.
- Whether or not to limit the range of the expansion – e.g., by expanding only to 1 or 2 levels.
- Whether to display the resulting visualization in a new window ("Hide"), or whether to add the resulting nodes to the current windows. In the latter case, some degree of control is given to the user with respect to the redrawing algorithm, by allowing her to decide whether or not to limit the freedom of the

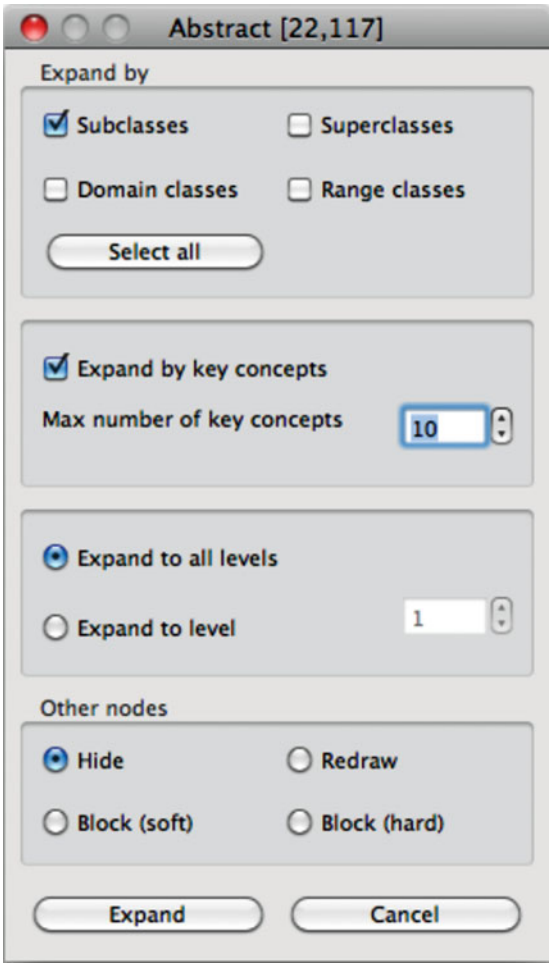

Fig. 16.5 Expanding sub-trees in KC-Viz

graph layout algorithm to rearrange existing nodes. This is particularly useful in those situations where expansion is meant to add only a few nodes, and the user does not want the layout to be unnecessarily modified – e.g., because she has already manually rearranged the nodes according to her own preferences.

As shown in Fig. 16.5, we have chosen to expand by key concepts, we have kept the limit of the expansion to 10 concepts, and we have also chosen to hide the other concepts, to be able to explore the sub-tree of class Abstract in a new window, without the "noise" from unrelated concepts.

Figure 16.6 shows the result of the expansion, which confirms the relatively poor degree of structure of the sub-tree under class Abstract, where only classes UserProfile and PhysicalQuantity appear to have been further characterized in

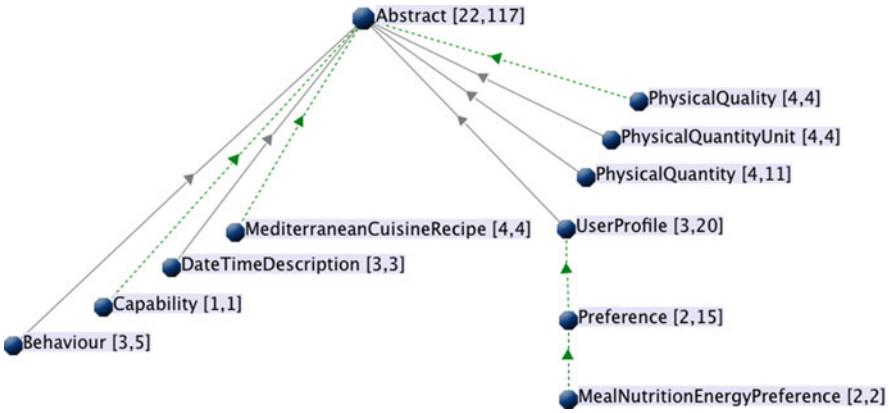

Fig. 16.6 Key concepts under class Abstract

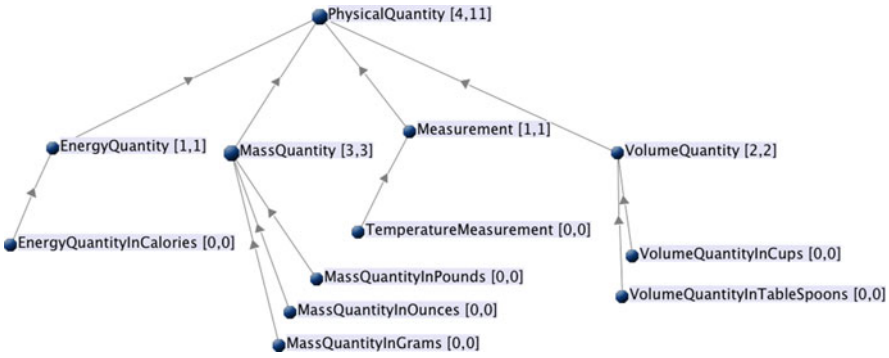

Fig. 16.7 Expanding class PhysicalQuantity

some detail and may warrant further exploration. In particular, we can look in more detail at the latter, by expanding its sub-tree at all levels, without restricting it to key concepts, as shown in Fig. 16.7.

Analogously, we can also explore the other key constituents of the SmartProducts network of ontologies, by careful expansion of the sub-trees we wish to explore. For instance, again by choosing expansion by key concepts, we can find out more about the structure of the sub-tree under FoodOrDrinkItem, as shown in Fig. 16.8.

16.4.3 Other Functionalities Provided by KC-Viz

While the flexible expansion mechanism is the key facility provided by KC-Viz to support flexible exploration of ontology trees, a number of other functionalities are

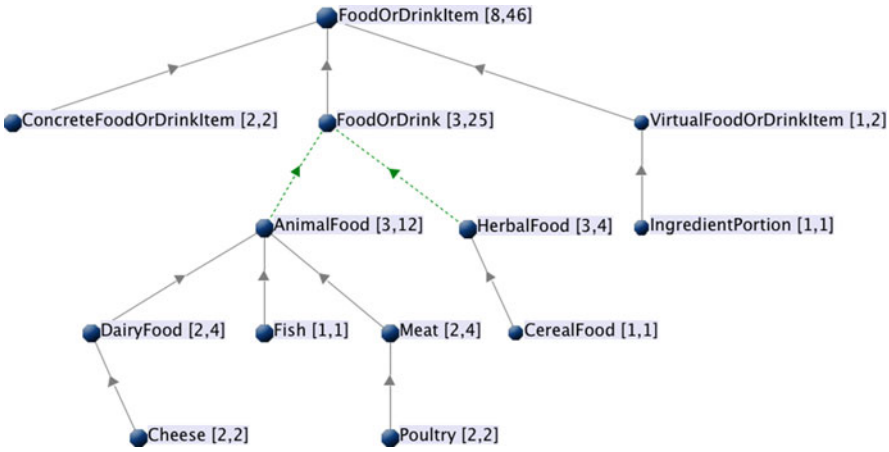

Fig. 16.8 Expanding class FoodOrDrinkItem

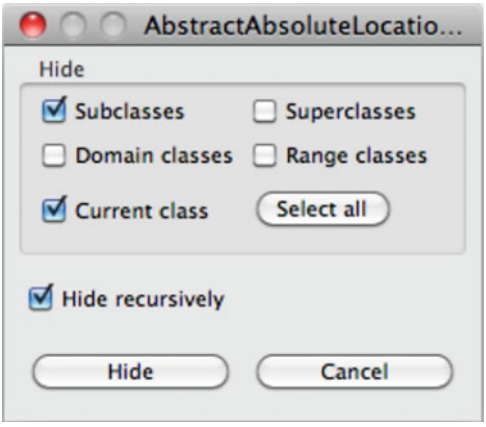

Fig. 16.9 Options for removing classes from a display

also provided, to ensure a comprehensive visualization and navigation support. These include:

- A flexible mechanism for hiding nodes, as shown in Fig. 16.9.
- Integration with the Entity Properties and Ontology Navigator tabs in the NeOn Toolkit, to support detailed inspection of classes.
- A dashboard, shown in Fig. 16.10, which allows the user to move back and forth through the history of KC-Viz operations, to modify the formatting of the layout, and to save the current display to a file, among other things.
- A preferences menu, shown in Fig. 16.11, which allows the user to set defaults for the most common operations and also enables her to switch to a more efficient (but sub-optimal) algorithm when dealing with very large ontologies.

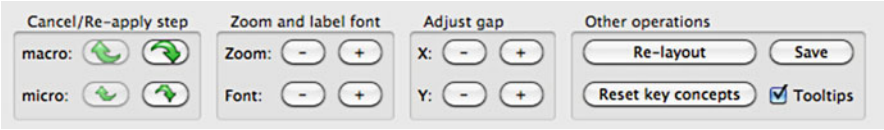

Fig. 16.10 The KC-Viz dashboard

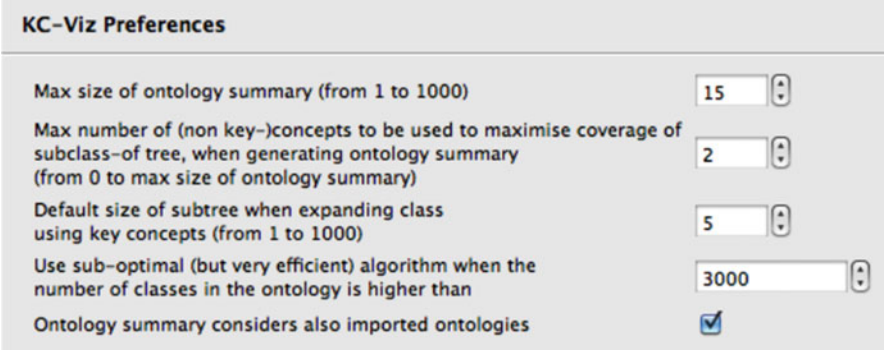

Fig. 16.11 KC-Viz preferences

16.4.4 Summing Up: How KC-Viz Addresses Key Challenges for Ontology Editors

Echoing the findings reported in the paper by (Dzbor et al. 2006), in Sect. 16.2 we highlighted a number of issues which hamper the effectiveness of current tools for visualizing and navigating ontologies. Here we revisit these issues, discussing how KC-Viz attempts to address them:

- *Poor efficiency and effectiveness*. The exploration sequence shown in Figs. 16.4, 16.6, and 16.7 (see Sect. 16.4) only required three operations and arguably provided us with a rather good understanding (at a given level of abstraction) of a significant part of the ontology. In our view, this compares favorably with the sequence described in Sect. 16.2, where several expansion operations did not dramatically improve our understanding of the ontology. It is also relatively straightforward to see that by repeating the exploration process we applied to class Abstract to other three or four key classes shown in Fig. 16.4 (see Sect. 16.4.1), we should be able to converge quickly to a rather comprehensive overview of the SmartProducts network of ontologies.
- *Lack of control when zooming on a particular node*. KC-Viz addresses this limitation by providing a very flexible set of options for node expansion, as shown in Fig. 16.5 (see Sect. 16.4.2).

- *No abstraction or saliency mechanisms*. The main abstraction mechanism provided by KC-Viz is the key concept extraction algorithm, KCE, which automatically identifies the "most important" concepts in the ontology, thus making it possible to present snapshots of the ontology to the user, while hiding away the "less important" concepts. Crucially, KCE has been empirically validated, thus providing a sound basis to the approach used in KC-Viz. In addition, by displaying information about the size of the sub-graphs under each node and by also varying the size of the graphical node representing a class in KC-Viz, the tool also provides a simple but effective mechanism to highlight the most "salient" classes in an ontology.

It is also interesting to assess the functionalities provided by KC-Viz with respect to the seven visualization task types proposed in (Shneiderman 1996). As discussed below, KC-Viz supports all of them:

- *Overview*. This is one of the key functionalities provided by KC-Viz. In contrast with other approaches – e.g., CropCircles (Wang and Parsia 2006), which sacrifice the display of explicit labels for the sake of maximizing the number of nodes on display, KC-Viz follows an alternative (and to our knowledge, unique) approach: it exploits the ontology summarization algorithm to provide initial overviews of an ontology and then allows the user to explore any part of the model in details. In our view, the advantage here is that, at any given stage of the process, only relatively few nodes are displayed and all of them are readable, thus making it easy for the user to make sense of the model on display. In addition, the simple display of information about the size of a class' sub-graph (shown as two integers describing the number of direct and indirect sub-classes) also provides summary information for the parts of the ontology which are not displayed, thus avoiding the need for abstract visualizations of clusters of nodes.
- *Zoom*. This functionality is supported by the Expand menu item, which provides a flexible set of options for exploring a sub-graph in detail. As a result, the user remains in control of both the size of the exploration space and the criteria used to generate it.
- *Filter*. This functionality is provided as a side effect of the ability of KC-Viz to focus on a particular part of the ontology and can also be invoked explicitly by the user by means of the Hide menu option.
- *Details-on-demand*. A tight integration with the Entity Properties view of the NeOn Toolkit makes it possible to click on any node displayed in KC-Viz and inspect it.
- *Relate*. KC-Viz supports the visualization of both taxonomic and domain/range relationships between classes. An example is given in Fig. 16.12, where all the "level 1" relationships between FoodOrDrinkItem and other classes in the ontologies are displayed. In particular, the dashed red arrows are used to indicate domain/range relations, with the labels being displayed when the mouse hovers over the arrow in question. In this case, we are showing that the ontology contains a relation hasNutrient, whose domain is FoodOrDrinkItem and whose range is NutrientPortion.

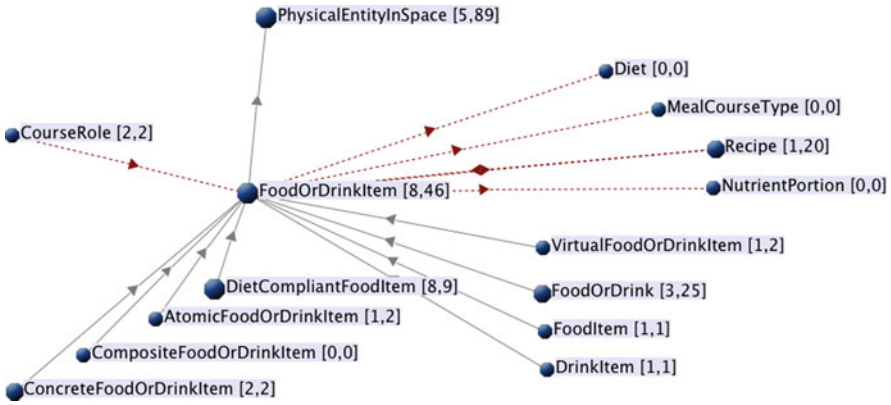

Fig. 16.12 Displaying both taxonomic and domain/range relationships

- *History*. KC-Viz supports undo/redo actions at both macro and micro level, to allow users to go back to, replay, or undo earlier operations.
- *Extract*. The key mechanism for extracting parts of an ontology is through the Expand menu, allowing flexible extraction of nodes at various levels in the hierarchy, in accordance with the key concept extraction algorithm, and following taxonomic and/or domain or range relationships.

16.5 Related Work

Surveys on ontology visualization methods, like (Katifori et al. 2007), categorize the methods for visualizing ontologies in six non-exclusive main types, called *indented list, node-link and tree*, *space-filling*, *zoomable*, *context + focus* and *distortion*, and *3D information landscapes*.

The *indented list* category covers tree-centric views of the ontology, similar to the one provided by the Ontology Navigator in the NeOn Toolkit, which was shown in Figs. 16.2 and 16.3 (see Sect. 16.2). As already pointed out, because of its familiarity to users, this style of interface is pretty much ubiquitous in ontology engineering toolkits; however, it does not support sensemaking tasks very well, especially in the case of large or unstructured ontologies.

Methods like *IsaViz*[9], *OntoViz*[10], and *SpaceTree* (Plaisant et al. 2002) are typical members of the second category *(node-link and tree)*, as they represent an ontology through a graphical display of interconnected nodes. Hence, these systems are similar to KC-Viz, with the crucial difference that, while KC-Viz uses ontology

[9] http://www.w3.org/2001/11/IsaViz
[10] http://protegewiki.stanford.edu/index.php/OntoViz

summarization to abstract out large trees, these methods tend to use preview icons – e.g., a triangle in the case of SpaceTree, to abstract out large sub-graphs.

TreeMap (Shneiderman 1992) is representative of *space-filling* approaches, which focus on optimizing the use of screen space to maximize the amount of information displayed to the user. However, in order to achieve this goal, these approaches tend to move away from the visualizations familiar to users, such as indented lists and graphs, and therefore they tend to require much more effort from users. In addition, it can be argued that no matter how much optimization a system tries to achieve, eventually it will fill all the available screen space, once a large enough set of data is given as input. Hence, while space filling is a useful secondary goal, in our view, the key goal for a visualization system remains the ability to provide views at different levels of abstraction, and in this respect, it can be argued that KC-Viz is unique in its reliance on an empirically validated ontology summarization algorithm, as opposed to general-purpose data abstraction techniques.

CropCircles (Wang and Parsia 2006) is an example of a "*zoomable visualization*," i.e., an approach which presents "the nodes in the lower levels of the hierarchy nested inside their parents and with smaller size than that of their parents" (Katifori et al. 2007). Much like KC-Viz, these approaches can be effective in providing good overviews, abstracting from large numbers of nodes. However, as already mentioned, in contrast with KC-Viz, which uses ontology summarization to provide abstraction, they sacrifice the display of explicit labels for the sake of maximizing the number of nodes on display.

The group of techniques categorized as "*context + focus and distortion*" is based on "the notion of distorting the view of the presented graph in order to combine context and focus. The node on focus is usually the central one and the rest of the nodes are presented around it, reduced in size until they reach a point that they are no longer visible" (Katifori et al. 2007). These techniques offer a good trade-off – a part of the ontology is shown in detailed (often tree-like) view, while the rest is depicted around. A typical approach here is *HyperTree* (Souza et al. 2003), which skews the visualized model to emphasize the node currently explored by the user. However, the problem with these techniques is that they essentially attempt to show everything in the model, which often makes the "context" part of little consequence and illegible. In contrast with these approaches, KC-Viz leverages the advantages derived from using ontology summaries, allowing the user to focus on ontology entities bearing the highest information value and "contextualizing" them against the entities with systematically lower information values. Crucially, it also provides flexible and effective mechanisms to change the focus to other entities, as and when required.

Finally, Katifori et al. also discuss a class of systems called "*Information Landscapes*," which provide a 3D, landscape-oriented alternative to zoomable visualizations. Hence, the remarks we made above about the latter category of systems apply to information landscapes as well.

16.6 Conclusions and Future Work

In this chapter, we have presented KC-Viz, an innovative approach to visualizing and navigating ontologies, which exploits a powerful ontology summarization algorithm, KCE, to introduce effective abstraction mechanisms in the ontology exploration and sensemaking processes. Crucially, KC-Viz maximizes the value of the foundational functionality afforded by KCE, by providing a flexible set of options to zoom in or hide specific parts of an ontology, history browsing mechanisms, flexible graphical layout formatting, and integration with other components of the NeOn Toolkit.

Our next task will be to evaluate KC-Viz formally, by comparing the performance in sensemaking tasks of users equipped with KC-Viz versus other users, to try and determine whether there is objective evidence that KC-Viz improves both the efficiency and the effectiveness of a sensemaking task.

We also plan to improve the range of functionalities provided by KC-Viz, in particular by opening up the key concept extraction algorithm to the users, to allow them to decide which criteria to prioritize in the generation of ontology summaries. Also, better explanation facilities are needed, as in some cases it is not easy to understand why a particular concept is deemed "important" by KC-Viz, while another one is not.

In conclusion, it can be argued that, with a few exceptions, the ontology engineering community has historically overlooked the importance of HCI issues and has failed to provide user interfaces that can truly support users effectively, as highlighted by Dzbor et al. (2006). With KC-Viz, we are trying to make an important step in the direction of providing better user support for ontology exploration and sensemaking, and we hope that our forthcoming empirical evaluation studies will confirm our intuition that the approach implemented in KC-Viz does indeed provide better sensemaking support for users of ontology engineering environments.

Acknowledgments This work was partially supported by funding from the European Commission, in the context of the NeOn and SmartProducts projects. The paper has benefited greatly from many insightful comments from Pierluigi Miraglia, who kindly suggested both ways to improve the presentation of this work as well as interesting new directions for future research.

References

d'Aquin M, Sabou M, Motta E (2008) Reusing knowledge from the semantic web with the Watson Plugin. Demo at the 2008 international semantic web conference, Karlsruhe, Germany

Dzbor M, Motta E, Buil Aranda C, Gómez-Pérez JM, Goerlitz O, Lewen H (2006) Developing ontologies in OWL: an observational study. Workshop on OWL: experiences and directions, Athens, GA, USA, Nov 2006

Katifori A, Halatsis C, Lepouras G, Vassilakis C, Giannopoulou E (2007) Ontology visualization methods—a survey. ACM Comput Surv 39(4):Art.10

Peroni S, Motta E, d'Aquin M (2008) Identifying key concepts in an ontology through the integration of cognitive principles with statistical and topological measures. In: Third Asian Semantic Web Conference, Bangkok, Thailand

Plaisant C, Grosjean J, Bederson BB (2002) Spacetree: supporting exploration in large node link tree, design evolution and empirical evaluation. In: Proceedings of the international symposium on information visualization, 2002

Rosch E (1978) Principles of categorization. In: Cognition and categorization. Lawrence Erlbaum, Hillsdale

Shneiderman B (1992) Tree visualization with tree-maps: a 2d space-filling approach. ACM Trans Graph 11(1):92–99, 15

Shneiderman B (1996) The eyes have it: a task by data type taxonomy for information visualizations. In: Proceedings of the 1996 IEEE symposium on Visual Languages (VL 1996), Boulder, CO, USA. IEEE Computer Society, Washington, DC, USA

Souza K, Dos Santos A, Evangelista SRM (2003) Visualization of ontologies through hypertrees. In: Proceedings of the Latin American conference on Human-computer interaction, 2003, p 251–255

Tartir S, Arpinar IB, Moore M, Sheth AP, Aleman-Meza B (2005) OntoQA: Metric-based ontology quality analysis. In Proceedings of the IEEE Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources, co-located with the 5th IEEE International Conference on Data Mining (ICDM 2005). November 27, 2005, Huston, Texas, USA

Wang TD, Parsia B (2006) Cropcircles: topology sensitive visualization of owl class hierarchies. In: Proceedings of the 5th International Semantic Web Conference 2006, Athens, GA, USA