Visual Support for Ontology Learning: an Experience Report

Marta Sabou
Vrije Universiteit Amsterdam
Marta.Sabou@cs.vu.nl

Abstract

Ontology learning methods aim to automate ontology construction. They are complex methods involving several elements such as documents, terms and concepts. During the development of an ontology learning method, as well as during its deployment, several situations occur where understanding the relations between these elements is crucial. Our hypothesis is that visual techniques can be used to aid this understanding. To support this claim, we present a set of such complex situations and describe the visual solutions that we developed to support them.

Keywords—**Semantic Web, Ontology Learning, Visualisation, Cluster Map**

1 Introduction

The Semantic Web aims to overcome the limitations of the current Web by augmenting Web resources with machine understandable semantics that will allow the automation of several tasks. Ontologies are key components of the Semantic Web technology being formal representations of shared domain knowledge. However, ontology building is a time consuming and difficult task thus hampering the development of the Semantic Web. The *Ontology Learning* field aims to provide methods that (semi-) automate ontology building. Learning methods involve multiple stages: they extract *terms* from a *document* corpus, then combine and transform them into *ontology elements* (concepts).

Ontology building and learning are essentially knowledge transfer processes from domain knowledge as perceived by humans to formal models that can be reasoned upon by computers. It is therefore essential that the knowledge structures involved in these processes as well as their relations are easily understood by human users. During our work on developing ontology learning methods in the context of semantic Web services [15, 16] we encountered several situations that required understanding the relation between different entities involved in the learning process.

Our hypothesis is that visual techniques, which harness human perceptual capabilities to detect patterns and outliers in visual information, can be used to aid this understanding. As such, they have the potential to facilitate the human expert's understanding of the complex structures and relations present in ontology building and learning.

Visual techniques are already used to support ontology building in ontology editors (Protege [11], WebOnto [4], OntoEdit [17]). The users of these tools need to get an insight in the complexity of the ontology they build. Therefore, these tools employ schema visualization techniques that primarily focus on the structure of the ontology, i.e., its concepts and their relations [6]. We are aware of two ontology learning approaches that employ visual techniques. First, the Text-To-Onto [9] tool employs a TouchGraph[1] based schema visualization to present the extracted ontologies. Second, in conjunction with the work presented in [13], a visualization technique is developed to present related terms (lexons) extracted from a corpus [12]. While useful, these visualizations only depict relations between entities of the same type (i.e., concepts, terms). However, due to the complexity of ontology learning, it is often crucial to know how entities of different types interrelate (e.g., *from which documents was a concept extracted?*).

To achieve this we used the Cluster Map [5], a visual technique developed by the Dutch company Aduna[2], which visualizes the instances of a set of classes, organized by their classifications. In this paper we present three different uses of the Cluster Map that support frequently occurring situations both during the development and the deployment of an ontology learning method.

In what follows we present details about the ontology learning process in general and describe the concrete ontology learning context from which our situations are drawn (Section 2). After describing the Cluster Map technique (Section 3) we detail three ontology learning situations and describe our visual solution for supporting each of them (Section 4). Finally, we conclude in Section 5.

[1] http://www.touchgraph.com/
[2] http://aduna.biz

2 Ontology Learning Basics

Ontology learning aims to alleviate the ontology acquisition problem that hampers the development of the Semantic Web. The majority of learning methods developed so far exploit textual sources. Thus, they start out with a ***corpus*** of textual ***documents***, where documents can often originate from multiple ***sources***. Generally, ontology learning methods perform three major steps.

1. Term extraction. In the first step, ***terms*** that are potentially interesting for ontology building are identified in the documents. For example, many methods identify noun phrases as terms potentially depicting domain concepts (e.g., <*dog*>, <*dogs*>, <*cat*>, <*Siamese cat*>[3]).

2. Conceptualization. In this step ***concepts*** and their ***relations*** are derived from the extracted terms. The transformation of terms into ontology elements is specific to each learning method. Some methods replace different forms of the same term with a base form and use some heuristics to place these concepts in a hierarchy [1, 16] (e.g. *Dog*, *Cat*, *SiameseCat*, where *SiameseCat* is more specific than *Cat*)[4]. Others use external knowledge, such as the WordNet lexical resource [10] or the Web [2], to derive more generic concepts (e.g., *Canine*, *Animal*).

3. Evaluation and enrichment of the ontology. The extracted ontology is inspected by a domain expert. This inspection is often performed during the development of the learning method to verify its accuracy and fine-tune it as desired. It is also performed during the deployment of the learning method. In this case a domain expert selects the relevant concepts and eventually enriches the ontology with new concepts. Understanding the extracted ontology and the functioning of the method often requires understanding how the different involved entities interrelate (e.g., sources, documents, terms, concepts).

Ontology Learning for Semantic Web Services. We developed an ontology learning method that extracts ontologies for semantically describing Web services [15, 16]. Web services are Web accessible software applications. Each document in our extraction corpora contains a short textual description of a Web service. The extracted ontologies contain two types of concepts necessary to describe Web services. First, they contain concepts that describe functionality types provided by Web services (e.g., *Search*, *Find*) grouped under the *Method* concept. Second, they contain concepts that describe the type of parameters expected by services (e.g., *Hotel*, *Airport*), grouped under the *DataStructure* concept. The situations presented in this paper are drawn from using the extraction method in two different domains: services that offer ontology storage and query facilities [15] and bioinformatics services [16].

[3]We use this notation to denote <*terms*>.

[4]We use this notation to denote *OntologyConcepts*.

We implemented our extraction method using the GATE natural language processing framework [3]. GATE already offers several visualization components such as visual program execution, visualization of parse trees and highlighting annotations in a document [14]. These visualizations provide details at sentence and document level but not at a global corpus level. Our visual components, implemented as GATE plug-ins, complement GATE in this aspect.

3 The Cluster Map Visualization

The Cluster Map technique visualizes *instances* of a set of *classes* according to their classification into these classes. As such, it provides a powerful tool to analyze, compare and query instantiated ontologies [5] and is integrated in several Semantic Web applications [7]. It can also be used to support automatic ontology population, i.e., classifying a set of instances according to the concepts of a given ontology [8].

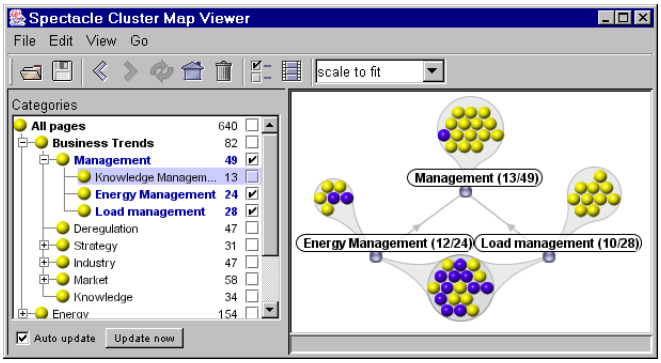

Figure 1: **Cluster Map Example.**

Fig.1 shows an example Cluster Map that visualizes a set of documents classified according to topics discussed in those documents. The map was created using the interactive GUI also presented in the figure. The left pane of the GUI presents the classes and their hierarchical relations. By selecting a class for visualization (select the corresponding checkbox) all its instances are visualized in the right pane. Each small sphere represents an instance. The classes are represented as rounded rectangles, stating their names and cardinalities. Directed edges connect classes and point from specific to generic (e.g., *Load Management* is a subclass of *Management*). Balloon-shaped edges connect instances to their most specific class(es). Instances with the same class membership are grouped in clusters (similar to Venn Diagrams). Our example contains four clusters; one of them represents the overlap of the *Load management* and *Energy Management* classes.

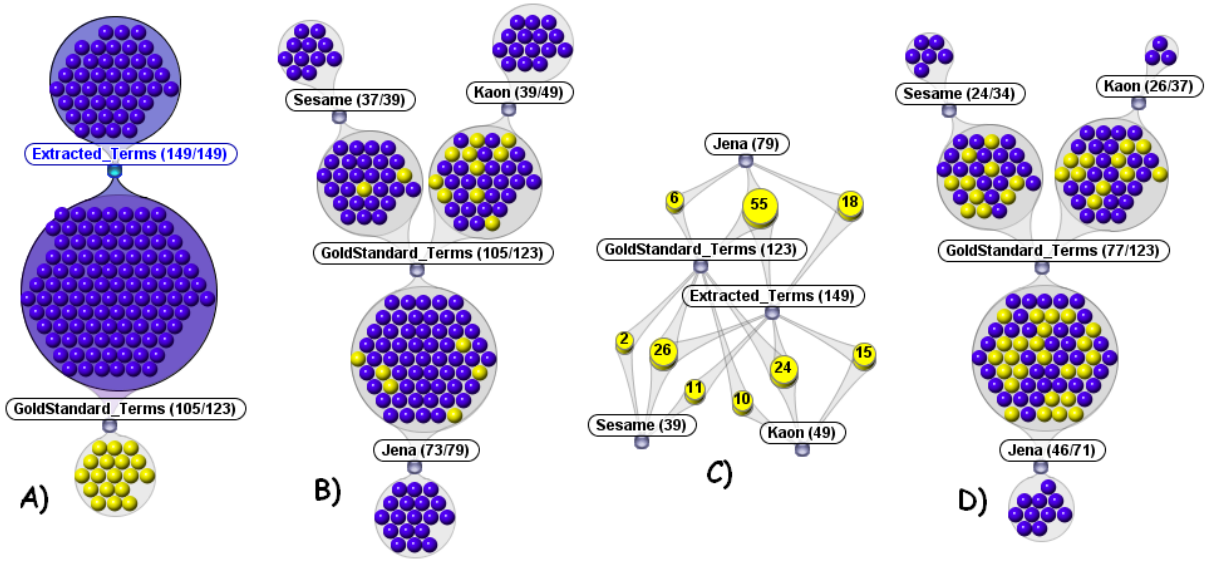

Figure 2: **Evaluating Term Extraction.** A) Recall and Precision per corpus; B) Recall per document sources; C) Accessing $correct$, $spurious$ and $missing$ clusters per sources; D) Recall per sources using another extraction method.

An attractive effect of visualizing instances only once and connecting them to all their classes is that visual closeness of classes often denotes their semantic closeness. A frequently used functionality of the GUI is *highlighting*. If a class is highlighted in the left pane then all its instances already visualized in the right pane are highlighted (their color changes from bright to dark).

4 Ontology Learning Situations

In this section we describe a set of ontology learning situations that were supported by the Cluster Map visualization. The visualizations built for each situation differ in the type of entities that play the role of classes and instances.

4.1 Evaluating Term Extraction

Situation Description. The quality of the term extraction process has a direct influence on the quality of the learned ontology. A good term extraction should identify **all** and **only** the potentially interesting terms (i.e., high Recall and Precision). To evaluate this process a Gold Standard corpus is prepared where all terms that should be extracted are manually identified. Then, these terms are compared to those automatically extracted. This comparison identifies correctly extracted terms ($correct$), terms in the Gold Standard that were not extracted automatically ($missed$) and finally terms that were incorrectly extracted ($spurious$). These values serve to compute the Precision and the Recall of the term extraction process:

$$Recall = \frac{correct}{correct + missed}$$

$$Precision = \frac{correct}{correct + spurious}$$

A visualization in this situation should support the developer of the learning method in several smaller tasks. First, the Precision and Recall of the extraction method have to be established. Further, if these are not satisfactory, the sources of the errors have to be identified and analyzed. This task involves visualizing the performance for each document source as well as granting quick access to the problematic term sets ($spurious$ and $missing$). Finally, if the extraction method is modified, the performance of two different extraction methods should be comparable.

Existing Solution. The Corpus Benchmark tool offered by GATE compares sets of terms extracted from two different versions of the same corpus (e.g., a Gold Standard vs. an automatically processed version). The output of the tool is textual and shows $correct$, $missed$ and $spurious$ terms per document, as well as the total Precision and Recall. This textual output does not facilitate access to the problematic term sets and hampers error discovery.

Proposed Visualization. We complemented the Benchmark tool with a visual output where instances represent terms. Each instance identifies the document and the position in the document where the term appears. If a term (e.g., <cat>) appears twice in a document at positions p1 and p2 two term instances are created for both occurrences. Term instances are classified along two dimensions. The first dimension represents term identification

processes: the manually identified terms are grouped in the *GoldStandard_Terms* class (*GST*), while the automatically identified terms are grouped in the *Extracted_Terms* class (*ET*). Therefore, a $correct$ term will be assigned to both *GST* and *ET* while $spurious$ and $missed$ terms will be assigned to *ET* and *GST* respectively. The second dimension represents the originating sources. Term instances are classified in classes representing the document sources from where they were extracted.

Examples. Fig. 2 illustrates several visualizations that can be obtained with this instantiation of the Cluster Map's data model. Part A visualizes both *GST* and *ET*, thus providing direct access to all important term sets, as follows. The overlap of these two classes represents the $correct$ set (i.e., extracted terms that are also identified by the Gold Standard), the cluster connected only to *ET* contains all $missed$ terms (i.e., extracted terms that are NOT identified by the Gold Standard) and the cluster connected to *GST* contains the $spurios$ concepts (i.e., Gold Standard terms that were NOT extracted). In addition, when *ET* is highlighted, the numeric fraction attached to *GST* is the ratio of all its highlighted instances (i.e., $correct$) over all its instances ($correct$ + $missed$) and therefore is equivalent to Recall. Precision can be shown similarly, but the major advantage of this visualization compared to a textual output is providing quick access to all term categories.

Recall and Precision can also be shown per document source, thus helping to localize the source that introduces the most errors. Part B of Fig. 2 shows the Recall of the extraction for three document sources. This visualization was obtained from the visualization in part A (1) by highlighting *ET* but not visualizing it and (2) by adding the three document source classes to the visualization. Clusters shared by a source and *GST* represent all terms that should have been extracted, and the bright objects represent the ones that were not extracted automatically. Since more bright objects indicate a lower Recall, it is easy to spot that the *KAON* document source introduces the most errors - in fact this document collection has a specific style which is hard to parse correctly. To access $correct$, $spurious$ and $missed$ terms for all sources a visualization such as in part C of the figure can be created. The only difference form B is that *ET* is not highlighted but selected for visualization. For each source, a cluster connected to both *GST* and *ET* represents the $correct$ set, while clusters connected only to *GST* (respectively *ET*) represent the $missed$ (respectively $spurious$) sets. A click on any of these clusters lists all their term instances and allows their inspection within the originating document.

Part D of Fig. 2 shows the Recall for the same document sources but for another extraction method. Comparing the images in part B and D it is evident that the Recall is worse in part D (more bright objects). This comparison demonstrates the benefit of the visual techniques over the textual output of the comparison tool. Rather than providing simple, global statistics the visualization gives the user the opportunity to qualitatively inspect the method's effects. It is interesting to see that in part D the KAON set no longer accounts for the majority of the errors because the method performs worse across all document sources. When combined with the developers knowledge about the quality of the document sources (e.g., that KAON's data is hard to parse) this observation clearly shows which of the two methods is better.

4.2 Understanding the Extracted Ontology

Situation Description. Ontology learning methods are not perfect. Besides often extracting irrelevant concepts, they are limited in grouping concepts under more generic concepts (discovering abstractions) as well as discovering relations between them. Therefore, any automatically extracted ontology has to be inspected by a domain expert which judges the domain relevance of the concepts and enriches the ontology with important abstractions and relations. To understand the meaning of a concept and to evaluate its correctness, it is often necessary to inspect the contexts in which it is used in the corpus, i.e., to access the documents from which it was extracted. Further, to discover new relations between concepts (or possible abstractions) an insight has to be provided on how a selected set of concepts interrelate at document level.

Proposed Visualization. To support these analysis tasks, we use the Cluster Map in a "traditional" way: each extracted concept forms a class and the documents from which it was extracted become its instances. Each document instance contains the position of the terms which resulted in deriving concepts. Documents from which no concept was learned are added under the *Top* class.

Examples. This visualization allows accessing all documents from which a concept was learned. This quick access to *evidence* information is essential for understanding the intended meaning of the concept.

To detect relations between a set of extracted concepts, we visualize these concepts and analyze the resulting image. For example, in Fig. 3 we visualized the functionality concepts extracted for the domain of bioinformatics. We observed that two groups of interconnected functionalities emerged (see part A and B of Fig. 3). Each group represents functionalities that are often offered simultaneously by Web services. At a closer look we observe that the first group (part A) contains functionalities that search or modify content, while in the second group (part B) we find functionalities concerned more with input/output operations such as *Reading* or *Writing*. The domain expert can easily access (with a simple mouse click) and inspect

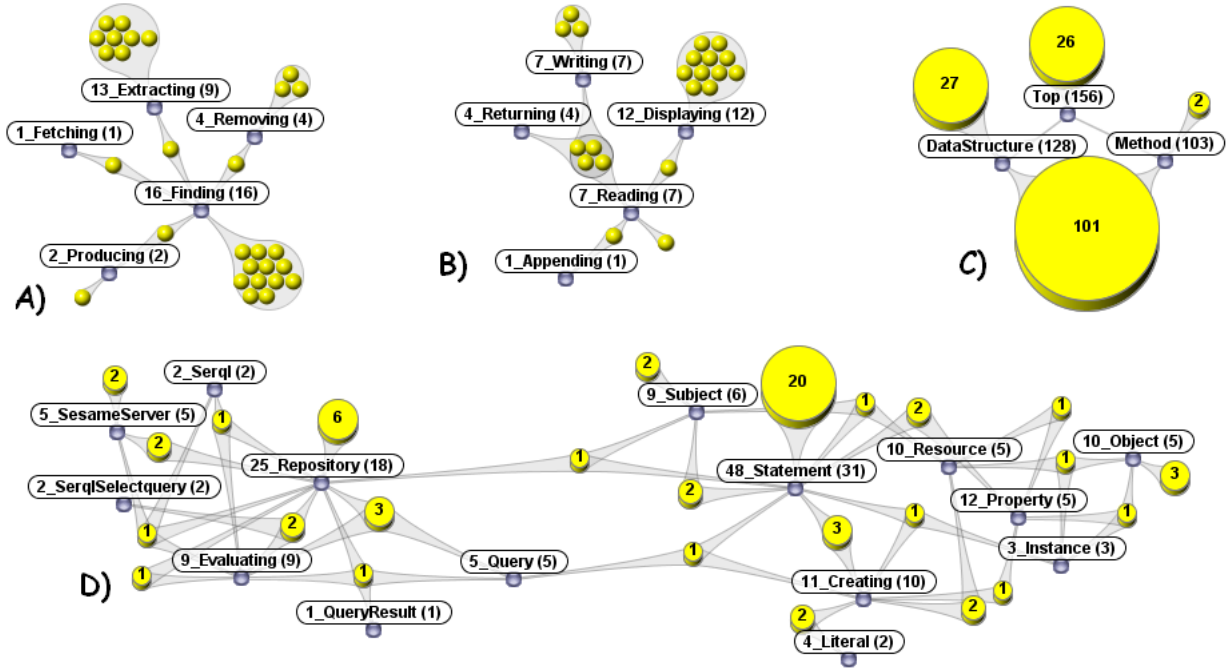

Figure 3: **Understanding the extracted ontology.** A), B) Related Web service Functionalities; C) *Method* and *DataStructure* concept type identification in the corpus; D) Detecting similar concepts based on verb's selectional restriction.

the documents that interrelate these concepts and decide if it is the case to set up new abstract categories (e.g., *ContentServices* and *InputOutputServices*).

In another example, part D of Fig. 3, we visualized both Method (*Evaluating* and *Creating*) and DataStructure type concepts. On the obtained visualization two groups of terms are formed around the verbs. The visual closeness of these concepts rightly suggests that they have some shared characteristics. The first group (by *Evaluation*) contains concepts that are involved in evaluating a query (i.e., a *SesameServer*, a *Repository* on that server as well as a *Query*). The second group contains concepts that describe elements of the RDF data model. In principle, this visualization is equivalent with the *selectional restriction* technique which is often used in ontology learning to detect semantically related concepts. This technique identifies terms that often occur in the same syntactic context as likely to be semantically related. For example, the objects of the verb *to drive* in *He drives the car* and *She drives a truck* share the characteristic of being drivable and therefore belong to the same semantic class (e.g., *Vehicle*).

Another task frequently performed by the method's developer is to measure how well the ontology covers the domain. For example, in our application domain we expect that each document should be described by at least one *Method* and one *DataStructure* concept. However, when visualizing all the documents in the corpus (*Top*), all documents from which a *Method* concept was extracted and all documents resulting in a *DataStructure* concept (see part C) we found out that only 101 out of 156 documents fulfil our expectation and that no concepts were extracted from 26 documents. It is easy to filter out documents that provided no (or only one type of) concepts and investigate the reasons. Solving them can improve the ontology learning.

4.3 Selecting the Most Important Concepts

Situation Description. By definition, an ontology is a *shared* conceptualization of a certain domain: it contains concepts on which all parties agree. Therefore, if a corpus combines documents from different sources, concepts that were derived from most sources are likely to be the more representative. The task is to detect these concepts.

Proposed Visualization. For this visualization instances represent concepts and they are classified in classes representing the document sources from which the corpus was built. Unlike the previous visualization, the concepts are now the instances under investigation and not classes.

Examples. In Fig. 4 part A we visualized the overlap of concepts extracted from three document sources. The jointly shared nine concepts promise to be commonly agreed upon terms in this domain, but concepts shared by only two sources are of interest as well. Part B shows the overlap of the same sources but for another ontology learn-

ing method. By comparing the two images one can conclude that the second method results in less extracted concepts and a lower overlap than the first one.

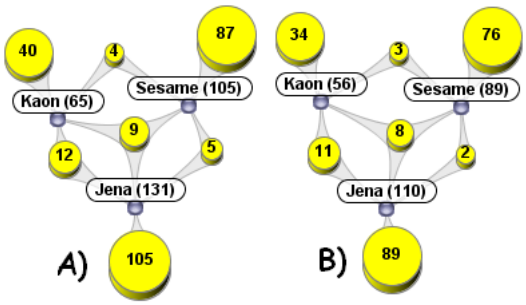

Figure 4: **Concepts shared by document sources.**

5 Conclusions

To verify our hypothesis that visual techniques can support several complex situations that occur during the development and the deployment of ontology learning methods, we identified a representative set of such situations and developed visualizations to support them. The situations were phrased at a generic level and exemplified in our concrete ontology learning context. Note that other learning methods could bring up new situations. To build the supporting visualizations, we exploited the fact that Cluster Maps visualize the relations between two types of entities (instances and classes). In the presented visualizations different types of entities (documents, terms, concepts, document sources) play the role of instances and classes to support different tasks. Naturally, the Cluster Map is NOT the only technique that can be used. Other techniques might be useful to support these (or other) situations.

Our conclusion is that visual techniques do enhance the development and deployment of ontology learning methods. In particular, the Cluster Map technique offers a powerful visual paradigm that can easily be adapted to support a wide range of generic (e.g., analysis, comparison, monitoring) or ontology learning specific (e.g., evaluation) tasks. The interactive GUI allows creating different visualizations just with a few mouse-clicks.

Future work includes investigating new situations, using other visual techniques, interlinking and integrating the presented visualizations in ontology learning tools.

References

[1] P. Buitelaar, D. Olejnik, and M. Sintek. A Protege Plug-In for Ontology Extraction from Text Based on Linguistic Analysis. *ESWS*, 2004.

[2] P. Cimiano, S. Handschuh, and S. Staab. Towards the Self-Annotating Web. *WWW*, 2004.

[3] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A framework and graphical development environment for robust NLP tools and applications. *ACL*, 2002.

[4] J. Domingue and M. Tadzebao. WebOnto: Discussing, browsing, and editing ontologies on the Web. *Knowledge Acquisition for Knowledge-Based Systems Workshop*, Canada, 1998.

[5] C. Fluit, M. Sabou, and F. van Harmelen. Ontology-based Information Visualisation. In V. Geroimenko, editor, *Visualising the Semantic Web*. 2002.

[6] C. Fluit, M. Sabou, and F. van Harmelen. Supporting user tasks through visualisation of light-weight ontologies. *Handbook on Ontologies*, 2004.

[7] C. Fluit, M. Sabou, and F. van Harmelen. Ontology-based Information Visualization: Towards Semantic Web Applications. In V. Geroimenko, editor, *Visualising the Semantic Web, Second Edition*. 2005.

[8] C. Fluit and J. Wester. Using Visualization for Information Management Tasks. *IV*. 2002

[9] A. Maedche and S. Staab. Ontology Learning. *Handbook on Ontologies*, 2004.

[10] R. Navigli and P. Velardi. Learning Domain Ontologies from Document Warehouses and Dedicated Websites. *Computational Linguistics*, 30(2), 2004.

[11] N. F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson, and M. A. Musen. Creating Semantic Web Contents with Protege-2000. *IEEE Intelligent Systems*, 16(2):60–71, 2001.

[12] A.J. Pretorius. Lexon Visualization: Visualizing Binary Fact Types in Ontology Bases. *IV*, 2004.

[13] M.-L. Reinberger, P. Spyns, and A.J. Pretorius. Automatic initiation of an ontology. *ODBASE*. 2004.

[14] P.J. Rodgers, R. Gaizauskas, K. Humphreys, and H. Cunningham. Visual Execution and Data Visualisation in Natural Language Processing. *IEEE Symposium on Visual Languages*, 1997.

[15] M. Sabou. From Software APIs to Web Service Ontologies: a Semi-Automatic Extraction Method. *ISWC*, 2004.

[16] M. Sabou, C. Wroe, C. Goble, and G. Mishne. Learning Domain Ontologies for Web Service Descriptions: an Experiment in Bioinformatics. *WWW*, 2005.

[17] Y. Sure, M. Erdmann, J. Angele, S. Staab, R. Studer, and D. Wenke. OntoEdit: Collaborative ontology development for the Semantic Web. *ISWC*, 2002.