

# Extending Community Ontology Using Automatically Generated Suggestions

Vít Nováček and Maciej Dabrowski and Sebastian Ryszard Kruk and Siegfried Handschuh

Digital Enterprise Research Institute  
National University of Ireland, Galway

Email: {vit.novacek, maciej.dabrowski, sebastian.kruk, siegfried.handschuh}@deri.org

## Abstract

In this paper we propose an ontology (formal knowledge base) creation methodology based on integrating external ontologies into the one developed by a community of the domain experts. We present the MarcOntX agent, a service, which allows to automate the process of generating suggestions of changes to the ontology. The suggestions are inferred from the external sources, such as large corpora of documents or concepts introduced by the users. We show the feasibility of our approach within two use case scenarios.

## Introduction

There are many scenarios within the ontology development process. Two basic approaches can be distinguished – (1) experts utilise their domain insight in the knowledge acquisition from scratch (collaborative ontology development) and (2) automatic NLP and machine learning methods are used for the processing of relevant resources and extracting knowledge from them (ontology learning). The collaborative creation is a dynamic process in which multiple developers participate, releasing subsequent versions of an ontology. There is a need of providing domain experts with a set of tools that will allow them to collaboratively develop ontologies. MarcOnt Portal, part of MarcOnt Initiative (Dabrowski 2006), is a tool designed for collaborative ontology development.

Sometimes, there is a need for the incorporation of a part of an external ontology into the MarcOnt ontology in order to improve it. Moreover, there are very data-intensive disciplines, for example bio-medicine, where it is not feasible to cover the whole domain only by collaborative means. In such cases ontology learning can help by producing possibly less precise and complex, but much broader ontologies enhancing the primary ontologies developed by experts. The main contribution of this paper is the design of the MarcOntX agent, a tool that supports easy (automatised) integration of the primary (collaborative) and external (e.g. learned) ontologies.

We further develop our motivations in next section. In the Section *Design of the MarcOntX Agent*, we discuss the

design of the proposed solution to MarcOntX – a semi-automatic ontology integration tool. We describe the application and preliminary partial evaluation of the MarcOntX agent in Section *Use Cases and Preliminary Evaluation*. Finally we conclude the paper in Section *Conclusions and Further Work*.

## Motivation

In this section we give an overview of the two main sources of our motivations and summary of related work. The general motivations are in line with the concrete use case examples in Section *Use Cases and Preliminary Evaluation*.

## MarcOnt Initiative Perspective

MarcOnt Initiative<sup>1</sup> is related to the librarian domain. MarcOnt is not only an ontology that conforms to the legacy bibliographic formats, MARC21, Dublin Core and BiBTeX. This initiative also provides domain experts with a set of tools for collaborative ontology development. These tools allow translation between other bibliographic formats supported by technologies for efficient reasoning and querying the knowledge base. The most important tools are MMS – MarcOnt Mediation Services and RDFT – RDF Translator.

**MarcOnt ontology** The MarcOnt ontology is a central point of MarcOnt Initiative. This ontology not only plays the role of an aggregation format, but also is a mediation format in the MarcOnt Mediation Services translation process. The MarcOnt ontology combines concepts from several bibliographic description formats and mediates between digital libraries using different metadata. Thus the ontology requires input from many people, especially domain experts.

Another tool delivered as a part of MarcOnt Initiative is MarcOnt Portal. This tool is responsible for the collaborative ontology development (see Fig. 1). Community members suggest changes (e.g. additional classes, properties, concepts) to the ontology. Other members annotate the suggestions and discuss their validity. On the base of these discussions, the members of MarcOnt community can vote for or against the changes.

**JeromeDL digital library** All resources in JeromeDL are semantically annotated using MarcOnt ontology. The de-

<sup>1</sup>See <http://www.marcont.org/>.

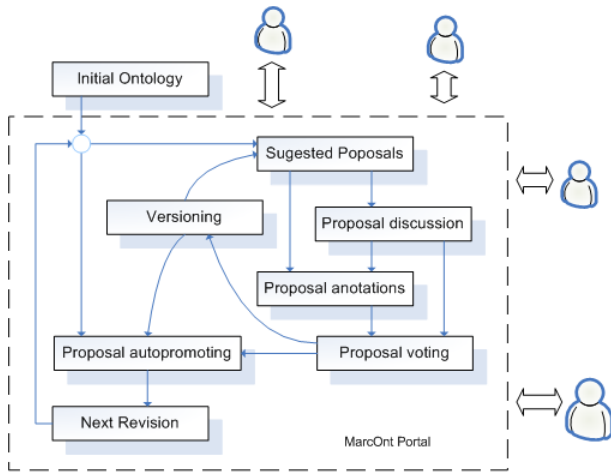


Figure 1: Cycle of development stages of the MarcOnt ontology

scription is created by the JeromeDL user within uploading of new items to a database. JeromeDL users can add new items to the library, described with MarcOnt ontology concepts, or with new concepts they find necessary. Until now, to incorporate these new concepts to the MarcOnt ontology, a user had to switch to MarcOnt Portal and add a suggestion (e.g. new property) of changes in the MarcOnt ontology. However, the process of gathering necessary data from many JeromeDL libraries should not involve user effort. This is the basic motivation for MarcOntX creation - automatic discovery and suggestion generation of new concept proposals in the description of JeromeDL resources.

### Dynamic Ontology Lifecycle Perspective

Figure 2 depicts a part of a dynamic ontology lifecycle scheme that has been developed within our other work (Nováček *et al.* 2006).

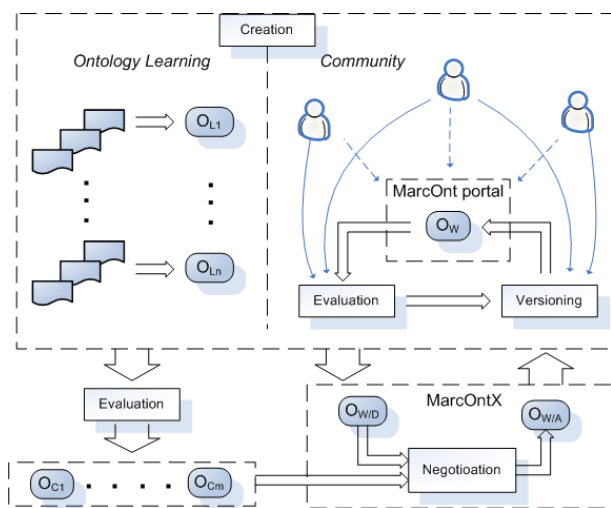


Figure 2: Ontology lifecycle part

It is related to ontology creation. The component has two parts. The automatic *ontology learning* phase is meant to dramatically enhance the coverage of community-driven *collaborative ontology development*, especially in such data intensive domains as medicine. Manually designed ontologies are supposed to be relatively complex and precise, however, it is very difficult and expensive to obtain extensive ontologies in these vast domains. On the other hand, the learned ontologies are perhaps simple and partially incorrect, but they are very broad in coverage of the domain. Therefore, a tool that can compare a learned and collaboratively created ontology and transform the differences into a set of reasonable extension suggestions for community experts would be desirable. Moreover, as the experts are generally not supposed to be ontology engineering professionals, the suggestions should be easily comprehensible. The MarcOntX agent's design is perfectly suited for this integration role within ontology lifecycle scenario.

### Related Work

Current research in the ontology management process related to the digital library domain involves many different approaches. These approaches concern issues from ontology versioning (Noy & Musen 2003) to the general ontology management frameworks (Iverson 2004). However, only a few propose extension of the collaboration ontology development process with other methodologies in technical point of view (Haase, Völker, & Sure 2005).

There exist various general methodologies of ontology development (Noy & Klein 2004; Fernandez-Lopez, Gomez-Perez, & Juristo 1997). They concern the whole cycle of ontology development, inspired either by the software lifecycle (Fernandez-Lopez, Gomez-Perez, & Juristo 1997) or by more specific and application-oriented demands (Noy & Klein 2004). All of these lack orientation to collaborative and distributed development, which has become very popular recently. (Schaffert, Gruber, & Westenthaler 2006) offers a wiki-based mechanism of distributed ontology development. Nonetheless, there is no support for automatic tools that is essential in data-intensive settings of the current web. Current methodologies or tools seldom propose or implement a concrete and efficient way of how to facilitate the integration of the two important parts of ontology creation – the ontology learning and the collaborative development. Therefore we present a possible universal solution to this task within the MarcOntX agent in this article.

### Design of the MarcOntX Agent

In the overview of MarcOntX agent (see Fig. 3) the  $O_E$  is an external ontology that contains possible extensions applicable to the master community ontology (represented with  $O_W$ ). The MarcOntX agent compares the  $O_E$  ontology with the dump of  $O_W$  ( $O_{W/D}$ ) and generates the suggestions that are proposed to the community. The DL-specific suggestion generation is described in the next section. The process which is implemented in the MarcOntX agent is introduced in Section *General Point of View*.

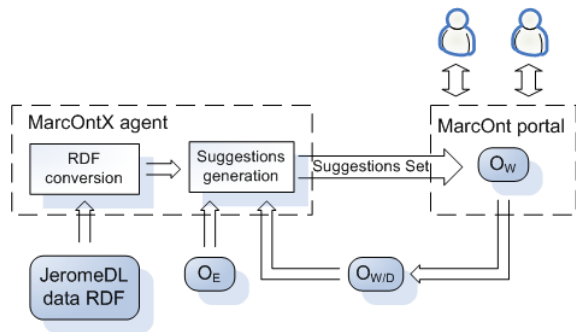


Figure 3: Architecture of the MarcOntX agent

### Suggestions Generation for the DL Domain

Digital libraries domain is not like any other, as it spans over many disparate topics from its very nature. DL systems address the needs of various users, therefore delivering one ontology for this domain is a very difficult process. Metadata solutions that have been adopted so far are either too generic as DublinCore, or perhaps too precise and heavy-weighted as MARC21. MarcOnt Initiative aims to deliver an ontology that would fulfill the requirements of various types of users. The ontology is collaboratively maintained within MarcOnt Portal. However, as building a community of various domain experts and library users is still on the way; there is a need of more independent means of delivering suggestions to the experts.

The suggestions of changes in the ontology are drawn to the MarcOnt Portal with the MarcOntX agent (see Fig. 3). The agent gathers RDF data from JeromeDL through JeromeDL’s RDF Query engine (Kruk, Decker, & Zieborak 2005). In the next phase, loosely coupled RDF triples are converted to the format compliant with the MarcOnt suggestions format (part of OWL-DL ontology). This conversion process is realized with the RDF Translator tool (RDFT). RDFT converts input RDF data according to the provided rules. MarcOntX then produces a set of suggestions as an output. Currently, suggestions described in Section *Use Cases and Preliminary Evaluation* are supported, however, we work on their extension by natural language recommendations generated according to the universal algorithms described in the next section.

### General Point of View

Possible extension of a master ontology  $O_W$  by elements contained in an external ontology  $O_E$  naturally corresponds to the differences<sup>2</sup> between them. Particularly, the possible extensions equal to the additions  $O_E$  brings into  $O_W$ . However, the number of the additions can be quite large, so an ordering that takes a relevance measure of possible suggestions into account is needed. Thus we can for example eliminate suggestions with low relevance level when presenting the final set to the users (without overwhelming them with

<sup>2</sup>Discovered by means of *SemVersion* library (Völkel & Groza 2006), which is actually a part of the MarcOnt Portal.

a large amount of possibly irrelevant suggestions). The suggestions are not meant to be exhaustive. They should rather identify a reasonable area of focus for the community, typically when extending the precise working ontology by possibly less precise, but very broad results of ontology learning. And eventually, as the users are generally supposed to be laymen with respect to ontology engineering, the suggestions should be as simple and intuitive as possible (preferably in natural language). Algorithm 1 describes this basic idea in the form of a meta-code.

---

#### Algorithm 1 Meta-algorithm of suggestion generation

---

**Require:**  $O_E, O_W$  — ontologies in RDF-based format

**Require:**  $PREF \neq \emptyset$  — set of user preferences

---

```

1:  $T_E \leftarrow makeRDFTriples(O_E)$ 
2:  $T_W \leftarrow makeRDFTriples(O_W)$ 
3:  $DIFF \leftarrow semVersion.getSemanticDiff(T_E, T_W)$ 
4:  $TRIPLES \leftarrow DIFF.getAdded()$ 
5:  $SORTED \leftarrow sortTriples(TRIPLES, PREF)$ 
6:  $SUGG \leftarrow []$ 
7: for  $T \in SORTED$  do
8:    $SUGG \leftarrow SUGG \oplus getNL(T)$ 
9: end for
10: return  $SUGG$ 

```

---

The *makeRDFTriples()* function creates RDF triples from non-RDF ontologies (thus being identity for RDF input). It is based on a simple method described in (Tablan *et al.* 2006). The method is not lossless in general (e. g., the precise semantics of complex OWL structures like union or other sophisticated constructors is not preserved). However, as was said before, we aim at the autonomous production of fairly simple suggestions, therefore the losses are not necessarily harmful. The suggestions are produced in the form of a very simple natural language statements. These are obtained directly from the sorted triples, using a minor modification of the generation process (the *getNL()* function) described in (Tablan *et al.* 2006)<sup>3</sup>. Concrete examples of natural language suggestions are shown in the next section.

The system has to be able to process thousands of possible extensions from learned ontologies. The most important element is the sorting of the triples in the extending set. As a possible solution to this task, we have proposed and implemented a method (the *sortTriples()* function).

Its core is sorting<sup>4</sup> according to a relevance computation based on string subsumption and Levenshtein distance (Levenshtein 1966). These two measures are used within relevance computation by comparing the predicate, subject and object lexical labels of a triple to two sets ( $S_p, S_n$ ) of words,

<sup>3</sup>Currently being incorporated into the MarcOntX agent.

<sup>4</sup>Note that the complexity of sorting itself mostly contributes to the overall complexity of the relevance-based sorting of suggestions. As can be found out from Algorithm 2, the complexity is in  $O(mnl^2 + m \log m)$  ( $m$  – number of triples,  $n$  – number of words in the preference sets,  $l$  – maximal length of a word in triple labels, basically a constant), which gives  $O(m(n + \log m))$ . As the size of the sets of user preferences can be practically treated as constant, we obtain the  $O(m \log m)$  complexity class with respect to the number of triples, which is feasible.

provided by users. The  $S_p$  and  $S_n$  sets contain preferred and unwanted words respectively, concerning the lexical level of optimal extensions.

A relevance score function is crucial in the sorting algorithm. It is given by the formula:

$$getScore(T, S_p, S_n) = rel(T, S_p) - rel(T, S_n),$$

where  $rel(T, S)$  is a function measuring the relevance of the triple  $T$  with respect to the words in the set  $S$ . The higher the value, the more relevant the triple is. We develop the relevance function in detail in Algorithm 2. Concrete examples of the sorting performance and its preliminary evaluation are provided in the following section.

The function naturally measures the “closeness” of the  $P, S, O$  labels to the set of terms in  $S_w$ . The value of 1 is achieved when the label is a direct substring of or equals to any word in  $S_w$  or vice versa. When the label’s Levenshtein distance from a word in  $S_w$  is lesser than or equal to defined threshold  $t$ , the relevance decreases from 1 by a value proportional to fraction of the distance and  $t$ . If even this is not the case (the label’s distance is greater than  $t$  for each word in  $S_w$ ), the similar principle is applied for possible word-parts of the label and the relevance is further proportionally decreased (minimal possible value being 0).

## Use Cases and Preliminary Evaluation

The first use case is related to the digital library domain; it presents the automatic suggestion generation from the data stored in JeromeDL digital library. The usecase highlights the improvement of the collaborative ontology development (in this case – the MarcOnt ontology) with automatic suggestions created with MarcOntX. The second use case is related to the translation medicine domain. It highlights that the MarcOntX agent can be applied in any other domain, in this example – the bio-medicine domain.

## Digital Library Domain

Digital libraries domain is not like any other; it spans over all possible topics; digital library systems address needs of various users; to deliver one ontology for this domain is a very hard process. Therefore, metadata solutions delivered for the libraries, so far, are either as generic as DublinCore, or as precise and heavy-weighted as MARC21. MarcOnt Initiative aims to deliver an ontology that would answer requirements of various types of users; the ontology is collaboratively maintained with MarcOnt Portal. However, building a community of various domain experts and library users is still on the way; we need some more independent means of delivering suggestions to this community.

MarcOntX allows to suggest changes to the ontology based on the requirements gathered from the JeromeDL (Kruk, Decker, & Zieborak 2005) users. Uploading process in JeromeDL adopts research on folksonomies (Mika 2005); users are free to use any properties to annotate publications; each property is referenced by an URI or a short name. In the latter case, if the property is not recognized as a property from the MarcOnt ontology, a new property is created in the MarcOntX namespace<sup>5</sup>. This ap-

<sup>5</sup>MarcOntX namespace: <http://www.marcont.org/xontology>

---

## Algorithm 2 The relevance function

---

**Require:**  $T = (P, S, O)$  — triple, consisting of  $P, S, O$  — *predicate, subject* and *object* labels respectively; possibly multiword

**Require:**  $S_w$  — set of words

**Require:**  $w_r, w_c, \rho \in (0, 1)$  — real constants; by setting  $w_c$  or  $w_r$  non-equal to 1, we can favour the structure (predicate) or content (subject and object) parts of triples respectively;  $\rho$  influences the absolute value of relevance measure

**Require:**  $t$  — integer constant; maximal allowed distance

**Require:**  $levDist(s_1, s_2)$  — Lev. distance implementation

```

1:  $R_P, R_S, R_O \leftarrow 0$ 
2: for  $elem \in \{P, S, O\}$  do
3:   if  $elem$  is a substring of or equals to any word in  $S_w$  or vice versa then
4:      $R_{elem} \leftarrow 1$ 
5:   else
6:      $d \leftarrow \infty$ 
7:     for  $v \in S_w$  do
8:       if  $levDist(elem, v) < d$  then
9:          $d \leftarrow levDist(elem, v)$ 
10:      end if
11:    end for
12:    if  $d \leq t$  then
13:       $R_{elem} \leftarrow (1 - \frac{d}{t+1})$ 
14:    else if  $elem$  is a multiword term then
15:       $L \leftarrow$  set of single terms in the  $elem$  label expression
16:       $EXP \leftarrow 0$ 
17:      for  $u \in L$  do
18:        if  $u$  is a substring of or equals to any word in  $S_w$  or vice versa then
19:           $EXP \leftarrow EXP + 1$ 
20:        else
21:           $d \leftarrow \infty$ 
22:          for  $v \in S_w$  do
23:            if  $levDist(u, v) < d$  then
24:               $d \leftarrow levDist(u, v)$ 
25:            end if
26:          end for
27:          if  $d \leq t$  then
28:             $EXP \leftarrow EXP + (1 - \frac{d}{t+1})$ 
29:          end if
30:        end if
31:      end for
32:      if  $EXP = 0$  then
33:         $R_{elem} \leftarrow 0$ 
34:      else
35:         $R_{elem} \leftarrow \rho \frac{1}{EXP}$ 
36:      end if
37:    end if
38:  end if
39: end for
40: return  $\frac{w_r R_P + w_c 2max(R_S, R_O)}{3}$ 

```

---

proach allows users to semantically annotate publications, and contribute to the development of the ontology at the same time.

The MarcOntX agent (see Fig. 4), which runs on MarcOnt Portal, executes an RDF query (see Fig. 5) on known JeromeDL instances; an RdfQuery servlet delivers a secure read-only access to the world-open part of information in the JeromeDL. The query retrieves all MarcOntX properties (see Fig. 6) together with objects they point to, and their types, if possible. A reasoning with RDFTranslator<sup>6</sup> (see



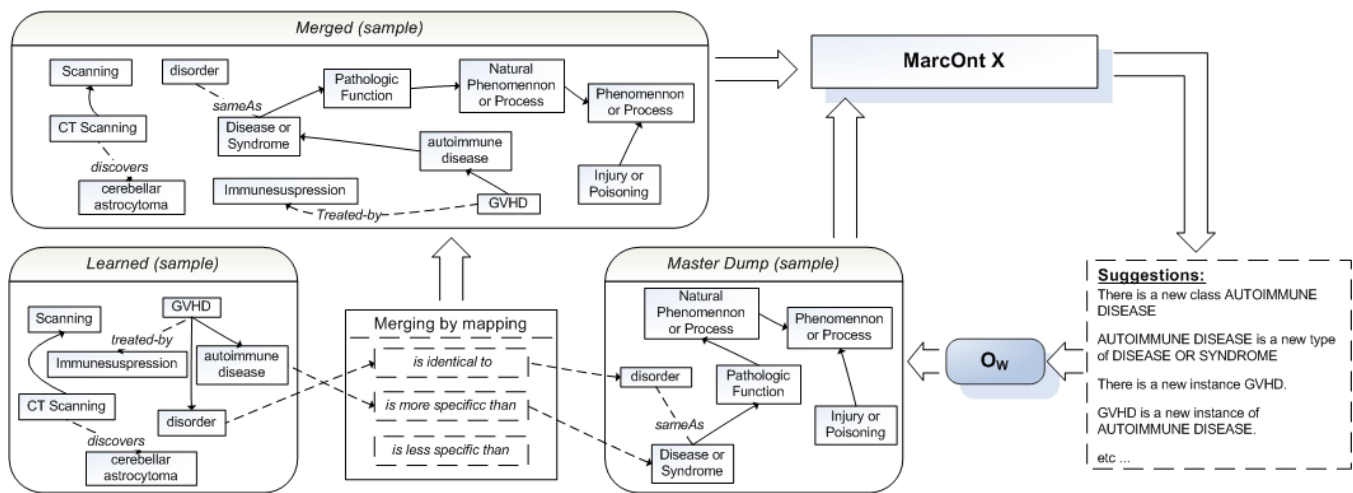


Figure 9: Medicine use case

## Conclusions and Further Work

In this paper, we presented the idea of extending community ontology using automatically generated suggestions. We also introduced the MarcOntX agent that implements the idea. As a proof of concept, we provided a use case that highlights the benefits of applying the MarcOntX agent to the digital library domain. Another application scenario, related to the bio-medicine domain, supports the universal perspective of our approach.

We have delivered an application prototype in the digital library domain. We have implemented and tested the most important part from the general point of view – the relevance-based sorting of suggestions. However, we need to implement the rest of presented algorithms in order to make the MarcOntX concept universally applicable in the ontology lifecycle scenario. Subsequent evaluation in real industry domains is needed then as well.

## Acknowledgements

This work has been supported by the EU IST 6th framework's NoE 'Knowledge Web' (FP6-507482) and by Enterprise Ireland Grant No. \*ILP/05/203\*, partially also by the Academy of Sciences of Czech Republic, the grant AV 1ET100300419 and by the Ministry of Education of Czech Republic within the Center of basic research LC536.

## References

- Dabrowski, M. 2006. Marcont initiative technical report. Technical report, DERI, Digital Enterprise Research Institute.
- Fernandez-Lopez, M.; Gomez-Perez, A.; and Juristo, N. 1997. Methontology: from ontological art towards ontological engineering. In *Proceedings of the AAAI97 Spring Symposium Series on Ontological Engineering*, 33–40.
- Haase, P.; Völker, J.; and Sure, Y. 2005. Management of dynamic knowledge. *Journal of Knowledge Management* 9:97–107.
- Iverson, L. 2004. Collaboration in digital libraries: a conceptual framework. In *Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*.

Kruk, S. R.; Decker, S.; and Zieborak, L. 2005. JeromeDL - Adding Semantic Web Technologies to Digital Libraries. In *Proceedings of DEXA'2005 Conference*.

Levenshtein, V. I. 1966. Binary codes capable of correcting deletions, insertions and reversals. *Cybernetics Control Theory* 10:707–710.

Mika, P. 2005. Ontologies are us: A unified model of social networks and semantics. In *International Semantic Web Conference*, LNCS, 522–536. Springer.

Nováček, V.; Handschuh, S.; Laera, L.; Maynard, D.; Völkel, M.; Groza, T.; Tamma, V.; and Kruk, S. R. 2006. Report and prototype of dynamics in the ontology lifecycle (D2.3.8). Deliverable 238, Knowledge Web. In press.

Noy, N. F., and Klein, M. 2004. Ontology evolution: Not the same as schema evolution. *Knowledge and Information Systems* 428–440.

Noy, N. F., and Musen, M. A. 2003. Ontology versioning as an element of an ontology-management framework.

Schaffert, S.; Gruber, A.; and Westenthaler, R. 2006. A semantic wiki for collaborative knowledge formation. In *Proceedings of Semantics 2005*. Trauner Verlag.

Tablan, V.; Polajnar, T.; Cunningham, H.; and Bontcheva, K. 2006. User-friendly ontology authoring using a controlled language. In *Proceedings of LREC 2006 - 5th International Conference on Language Resources and Evaluation*. ELRA/ELDA Paris.

Völkel, M., and Groza, T. 2006. SemVersion: RDF-based ontology versioning system. In *Proceedings of the IADIS International Conference WWW/Internet 2006 (ICWI 2006)*.