TermIt: A Practical Semantic Vocabulary Manager

Martin Ledvinka, Petr Křemen, Lama Saeeda and Miroslav Blaško
Department of Computer Science, Faculty of Electrical Engineering, Czech Technical University in Prague, Technická 2, Prague 6 - Dejvice, Czech Republic

Keywords: Vocabulary Management, Glossary, Semantic Search, SKOS.

Abstract: Many organizations already benefit from using semantic vocabularies which help them systematize, search and reuse their data. However, to efficiently manage such vocabularies, appropriate and adequate tools are needed. In this paper, we present TermIt, an integrated system for managing a set of interconnected vocabularies, identification of individual concepts in source documents, and using such terms for semantic data asset annotation and subsequent search. We relate TermIt to other relevant tools and present usage scenarios we have identified so far.

1 INTRODUCTION

Consider the following sentences: "The construction of the Large Hadron Collider took ten years." and "The construction of the Large Hadron Collider is hidden in a 27 km-long underground tunnel." The word *construction* is used in different meanings conveyed by the context – an event/process in the former case and its outcome in the latter. Now consider another example: according to the Energy Management Act of the Czech Republic, a *building* is a construction both above and below ground with heating. On the other hand, according to the Land Registry Act, a *building* is a construction above ground with solid foundations. It can be seen that the words denote possibly different underlying *concepts*. Contemporary search engines are based on words and would make no distinction between documents using the word *building* in either sense.

Such issues lead to the need for building (pun intended) *glossaries* of terms with precise definitions of the concepts they represent. These glossaries can then be used as a reference and disambiguate the meaning of plain language words, and are of crucial importance when delivering well-defined search over data assets (e.g., open data sets provided by the public administration) – both for query specification and result presentation. A glossary may start as a flat list of terms with definition and label, but usually at least a basic hierarchical structure specifying that some terms are more general (in various senses) than others can be derived. In this case, organizational schemes like SKOS (Miles and Bechhofer, 2009) need to be used to describe this structure. Once a glossary exists, the relationships between its terms can be refined – an ontological *model* of the domain is built. The glossary, containing a hierarchy of terms, and the model, specifying concrete relationships between the terms, comprise the *vocabulary* of the domain in discourse.

1.1 Motivation

While it is reasonable to expect that domain experts will be able to create a glossary, the task of building a domain model from this glossary is far more complex. In our experience, most domain experts need to be guided through this process, or even worse – they just validate model proposals offered by knowledge engineers (Křemen and Nečaský, 2019). This is mostly because a model is typically built using a specific formalism, which gives its elements and their relationships a predefined semantics. Such formalisms may range from purely technical like UML (Rumbaugh et al., 2004) or OWL (Motik et al., 2012) to more abstract like the Unified Foundational Ontology (UFO) (Guizzardi, Giancarlo, 2005). Domain experts are rarely proficient in these modeling formalisms and a modeling expert is needed for the task of transforming the glossary into a model.

The goal of this work is to streamline the process of vocabulary creation and management by giving domain experts *TermIt* – a tool for efficient glossary management. In addition, the glossary terms can be utilized to provide a more precise resource (data as-

Ledvinka, M., Křemen, P., Saeeda, L. and Blaško, M.
TermIt: A Practical Semantic Vocabulary Manager.
DOI: 10.5220/0009563707590766
In *Proceedings of the 22nd International Conference on Enterprise Information Systems (ICEIS 2020) - Volume 1*, pages 759-766
ISBN: 978-989-758-423-7

set) search, in which the meaning of the terms rather than their label is used to describe the resources.

The rest of the paper is structured as follows: Section 2 introduces the theoretical foundations of our terminology editor, whereas Section 3 presents the tool itself. Section 4 reviews related work and Section 5 discusses our experience with using the tool. The paper is concluded in Section 6.

2 ONTOLOGICAL FOUNDATIONS

Semantic Web ontologies, thanks to their ability to uniquely identify and formally specify concepts and their relationships (Guarino et al., 2009), share schema defined in standard languages, and allow inference of implicit knowledge thanks to expressive underlying formalisms, are a natural choice for building domain vocabularies. In such a scenario, a vocabulary is an ontology which can be split into a glossary and a model, which are ontologies themselves, albeit at a different level of complexity. However, ontology-based terminology editors and modeling tools add yet another layer of knowledge required from their users. While ontology engineers can take care of modeling, domain experts may not appreciate having to learn this new formalism.

Nevertheless, one should not give up the benefits of Semantic Web technologies so easily. That is why the terminology editor presented in this work has strong ontological foundations, whilst providing an easy to use interface even for users without Semantic Web background. TermIt stores the vocabularies as RDF (Cyganiak et al., 2014) under a well-defined ontological model which is itself based on several other more general ontologies. This hierarchy is visualized in Figure 1 and described below.

2.1 Unified Foundational Ontology

The *Unified Foundational Ontology* (UFO) (Guizzardi, Giancarlo, 2005) is an upper-level ontology – an ontology of generic terms (Mascardi et al., 2007). It allows more specific ontologies to share a common meta-model of basic concepts and relationships. This allows, for example, someone with the knowledge of UFO to quickly gain an insight into a domain by examining how its elements relate to the generic UFO model. UFO draws inspiration from philosophy and modal logic and introduces class stereotypes (essentially, classes of classes) such as *Kind*, *Phase*, *Role*, or *Relator*, and relationship stereotypes like *Mediation*, *Characterization*, and several part-whole types.

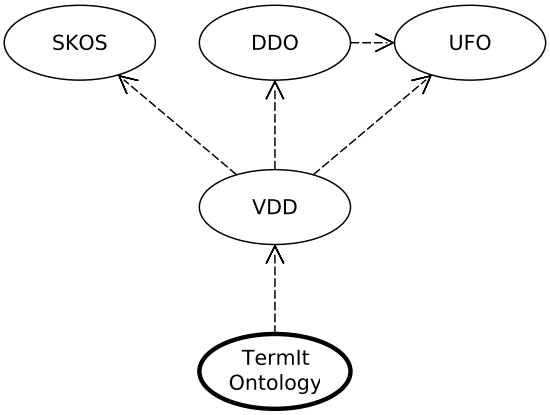

Figure 1: Hierarchy of ontologies on which TermIt is based. Arrows represent dependency. The TermIt ontology is emphasized. Acronyms are explained in the text.

As mentioned, using UFO allows to ground an ontology in a common, well-defined and computable structure of basic concepts.

2.2 Simple Knowledge Organization System

The *Simple Knowledge Organization System* (SKOS) (Miles and Bechhofer, 2009) provides a simple and generic vocabulary for organizing knowledge in a wide range of domains. SKOS recognizes two main classes – a *Concept* which represents a notion in the domain of discourse, and a *Concept scheme*, representing a collection of concepts. Concepts can then be characterized by a number of properties, like *preferred* and *alternative label*, and put into a hierarchy using the *broader* and *narrower* properties. The meaning of all the SKOS notions is intentionally vague, so that it fits various use cases.

The difference between use cases of SKOS and UFO is analogous to our ideas of glossary and model. While a glossary uses SKOS for domain decomposition due to its simplicity, a model uses UFO, which gives the ontology engineer greater power to precisely describe the understanding of the domain.

2.3 Dataset Descriptor Ontology

The *Dataset Descriptor Ontology* can be used to characterize datasets (Blaško et al., 2016). The original idea is that every dataset can be described using a *descriptor* – an easy to interpret and visualize RDF summary. Such a descriptor can provide, for example, information about the number of concepts and their usage in the dataset. In the case of TermIt, vocabularies as well as resources are considered datasets.

2.4 Vocabulary for Data Description

The Vocabulary for Data Description (VDD)[1] is a general-purpose ontology for describing vocabularies, data sources and their attributes. For example, it recognizes three kinds of vocabularies: agenda (describing an operational area), data (describing a data source), and document (stemming from a normative document) vocabularies.

The TermIt ontology relies on the VDD and adds more application-specific classes and properties necessary for the proper functionality of the system.

3 TermIt

TermIt is an easy-to-use vocabulary manager and glossary editor. It allows domain experts to define their own terms (SKOS concepts with defining metadata), organize them in (SKOS) hierarchies and use them for resource annotation and search. A simplified schema of TermIt's approach can be seen in Figure 2.

The two main functional areas of TermIt are:

- Vocabulary management
- Resource annotation and search

3.1 Vocabulary Management

TermIt allows its users to manage vocabularies and, primarily, edit their glossaries. The glossaries have SKOS-compatible structure. That is, the terms are arranged in a (potentially ambiguous – the relationships can be of different nature) hierarchy. For each term, basic data like label, definition, source of the definition, and additional commentary can be provided. Moreover, the terms can be classified using a simplified, UFO-based vocabulary. This classification vocabulary is configurable, so, for example, for the area of legislation document vocabularies, more specific types are available. For instance, a term can thus be classified not only as a relationship, but, more precisely, a legal relationship.

Vocabularies managed by TermIt can be related to each other, typically also in a hierarchical manner. The term hierarchy can then span a hierarchy of vocabularies as well. For instance, in one of the use cases in the urban planning domain, there exists a hierarchy of three vocabularies: the Building Act, the Prague Building Regulations and the Metropolitan plan of Prague. Figure 3 then shows how a hierarchy of selected terms may cross the boundaries between these vocabularies.

Structuring vocabularies and terms in a hierarchy is natural, but it is of course possible to relate them without assuming any broader-narrower relationship. Therefore, a vocabulary simply imports another vocabulary. As far as terms are concerned, one can use the SKOS *related* property to specify an associative relationship between terms without giving its details (which may be provided later by the ontology engineer).

TermIt currently supports regular vocabularies and document vocabularies. Document vocabularies are associated with a document which may consist of several files (see Section 3.2). The idea is that a document vocabulary is typically based on a normative document, whose text is the source of the terms in the vocabulary and their definitions. For instance, the vocabularies from Figure 3 are all document vocabularies.

Besides the aforementioned SKOS-based attributes and provenance data (author, date of creation etc.), a term can have additional properties. These properties can be used to provide further information about the terms, for example, the department responsible for maintenance of a specific term. Furthermore, it allows TermIt to partially cross the boundary into the modeling realm. For instance, consider the SKOS broader relationship between Non-buildable surface and Surface from Figure 3. Using these additional properties, it is possible to declare that Non-buildable surface is actually a specialization of Surface. Of course the ergonomics of this interface is far worse than that of a modeling tool, but it can provide at least a basic insight into the model to a more advanced user.

The glossary editing part of TermIt does not cover the whole SKOS vocabulary, instead, a selected set of classes and properties deemed relevant is used. This set can be seen in Figure 4. Some of the relationships (e.g., the SKOS inScheme property) need not be explicitly stated in the data. Instead, reasoning capabilities of the underlying triple store and the TermIt ontology schema are utilized to infer them.

3.2 Resource Annotation and Search

TermIt allows to register resources in order to support two main scenarios: 1) creating vocabularies based on normative documents; 2) semantic search of resources facilitated by terms assigned to them.

Four types of resources exist in TermIt:

Resource base type of data assets without any closer characterization,

[1] Available at http://onto.fel.cvut.cz/ontologies/slovnik/agendovy/popis-dat/current/index-en.html, accessed 2020-03-15.

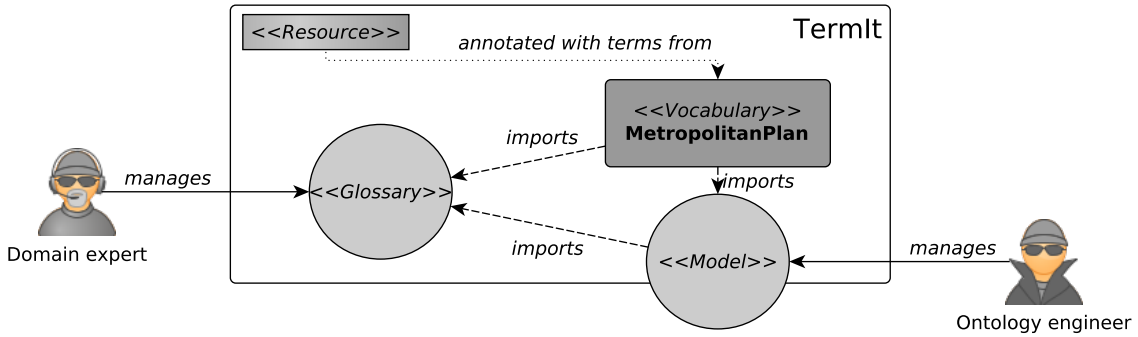

Figure 2: Simplified schema of TermIt usage.

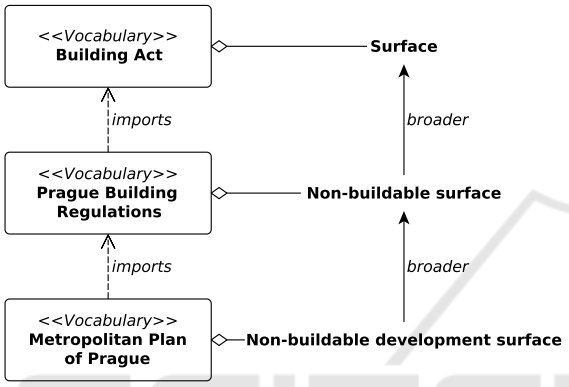

Figure 3: Vocabulary and term hierarchy example. Borderless nodes represent terms.

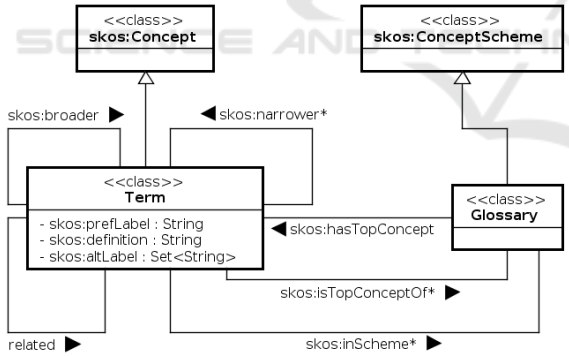

Figure 4: UML class diagram illustrating the use of SKOS classes and properties in TermIt. Prefix-less classes are local to TermIt (they actually come from the VDD). Properties marked with an asterisk are inferred.

Dataset a set of arbitrary data, which may or may not be in a Semantic Web-compatible format. For instance, many public organizations nowadays publish their datasets as part of the open data initiative, yet, a lot of these datasets are in the form of CSV files,

Document a logical container consisting of zero or more files. Document vocabularies are based on documents,

File represents a single file whose content may be uploaded into the system and, in case of textual files, analyzed.

Documents and textual files are relevant for scenario 1), whereas scenario 2) covers all the four types of resources.

3.2.1 Creating Vocabularies based on Documents

This use case represents a situation when one wants to create a vocabulary based on a document. Since normative documents often rely on auxiliary files, a document, in the understanding of TermIt, is primarily a logical container of files. In this scenario, the user creates a document vocabulary and uploads the relevant file(s) into TermIt. Afterwards, it is possible to run an automated text analysis service on these files. This service is able to suggest new terms based on their significance in the file content. The text analysis service will be discussed in further detail in Section 3.3.1.

The user can then open the annotated file in TermIt, review the suggestions (marked in text), create terms from them, or mark and create new terms.

3.2.2 Resource Annotation

This scenario assumes that a non-empty vocabulary already exists. This vocabulary can be used to annotate arbitrary resources registered in TermIt and thus provide a term-based resource search.

It is possible to *assign* terms to resources, indicating that they are relevant for the resource in question. For example, datasets published online can be registered in TermIt, assigned terms and searched for.

In addition, for textual files, the text analysis service can also be used to discover *mentions* of terms in the content. The service will suggest the mentions, so that the user may approve or discard them. If

the mention has sufficient confidence score (see Section 3.3.1), the term is also assigned to the resource.

With this, one can use TermIt as a catalogue of resources and search them more accurately thanks to the terms. Consider an example related to the motivational scenario in the beginning: a user is interested in resources mentioning "construction" as the process of building something. They open TermIt and run regular full-text search for the word "construction". TermIt finds terms and vocabularies whose relevant attributes (label, definition, description) match the search string. The user selects the appropriate term. On the term detail screen, TermIt offers the user to view resources to which the term is assigned, effectively eliminating cases where "construction" is meant as a physical artifact.

Such a functionality can go even further in that the user may discover related resources – resources sharing common terms. This is currently not implemented directly in the TermIt user interface, but rather provided as a Web service for other tools. An experimental instance of the Czech National Open Data Catalog[2] is able to find related datasets via the terms they share. Another example of annotating resources concerns vocabulary models – ontology engineers responsible for modeling a vocabulary register a resource representing the model's diagram in TermIt and annotate it with the terms it contains. TermIt users can then immediately see the topic of the model just by looking at the terms it is annotated with.

3.3 Technical Viewpoint

TermIt is an open source Semantic Web-based information system consisting of an ontological model and a regular Web application. The application itself has a backend[3] written in Java and a frontend[4] written in TypeScript using the React framework.[5] The system stores its data in a triple store with a custom set of rules for inference tasks. As far as expressiveness of this ruleset is concerned, it supports RDFS (Brickley and Guha, 2014) and selected OWL features such as inverse properties. The frontend communicates with the backend via a REST API using JSON-LD as the data format (plain JSON is supported as well). The REST API is documented using Swagger.[6]

[2]https://data.gov.cz/english/, accessed 2020-03-15.

[3]Source code available at https://github.com/kbss-cvut/termit, accessed 2020-03-15.

[4]Source code available at https://github.com/kbss-cvut/termit-ui, accessed 2020-03-15.

[5]https://reactjs.org/, accessed 2020-03-15.

[6]https://app.swaggerhub.com/apis/ledvima1/TermIt/, accessed 2020-03-15.

Since the data in TermIt are stored in a triple store, it is easy to provide them as Linked Data (Wood et al., 2013). The current instances achieve this via a *Pubby* (Cyganiak and Bizer, 2007) instance which is pointed to the underlying repository.

3.3.1 Annotace

Annotace,[7] a text annotation service, is implemented and used in the context of TermIt. Annotace enhances the process of building vocabularies that are related to textual resources in the two scenarios already discussed in Sections 3.2.1 and 3.2.2:

- In the first scenario, a new document (or, more precisely, a set of files comprising the document) is uploaded into the TermIt document manager, and a newly created vocabulary is associated with it. The vocabulary is empty at this point. The task is to help the user start building the vocabulary based on the text present in the document. Annotace starts analyzing the text based on statistical methods, namely, it uses the keyword extractor tool (KER) (Libovický, 2016) to extract the most significant mentions from the text as candidate concepts in the vocabulary. This step does not involve any semantic technology since there is no semantic information present in the knowledge base yet. The extracted information is then presented to the user by TermIt as highlighted text with actions. These actions allow the user to create a new term in the vocabulary or reject the suggested term if it is irrelevant to the associated vocabulary.
- The second scenario has a lot in common with the previous one, but it assumes that the vocabulary already has seed terms. Besides the steps introduced in the first scenario, Annotace starts analyzing the document using the concepts in the associated vocabulary to find mentions in the text that refer to specific entities in the vocabulary and provides links between them. These mentions are also presented as highlighted text in the document, but differ from the extracted terms in the statistical step by providing a link to the associated term directly. Similar to create and reject actions, the user is allowed to approve the suggested association or change it to a different term.

Both scenarios suggest human interaction with the system to approve or reject the output of Annotace. The semi-automatic approach is paramount to keeping high precision of building the ontology and saving the user time and effort needed to be spent with

[7]Source code is at https://github.com/kbss-cvut/annotace, accessed 2020-03-15.

the manual process. Annotace produces data in the HTML format and the annotations are created using RDFa (Adida et al., 2015) – an extension to HTML 5 that allows injecting Linked Data annotations into the structure of an HTML document. Whenever a token is recognized as a mention of an entity in the vocabulary, a new annotation is injected around this token with properties like a unique identifier, the *resource* attribute referring to the identifier of the vocabulary term, the type of the annotation in the ontology model, and the accuracy of the prediction represented by the *score* attribute.

3.3.2 Demo

The current deployments of TermIt are in Czech, but an English deployment has been created for demonstration purposes.[8] It contains two urbanism-related vocabularies – the Prague Building Regulations vocabulary and the vocabulary of the Metropolitan Plan of Prague. Both vocabularies were created based on the respective normative documents (only in Czech) and translated to English as well.

Since the demo repository also contains vocabulary models, additional properties are available for the terms in the user interface, e.g., more precise specification of the broader/narrower relationships.

It is possible to log into the demo deployment and explore the functionalities of TermIt under the username **demo** and with password **demo**.

4 RELATED WORK

Related work can be split based on the two use cases supported by TermIt – generic vocabulary management (as opposed to existing domain-specific vocabulary management tools like SnowOwl/Snowstorm for SNOMED CT), and resource annotation and search.

4.1 Vocabulary Management

There exist several glossary editors and vocabulary management tools. Their common problem is that they expect knowledge of the Semantic Web principles from their users. This represents a significant barrier for most domain experts. Compared to TermIt, non-commercial tools also provide limited support for linking the terms to their source documents.

VocBench (Stellato et al., 2015) is a tool for collaborative SKOS thesauri management. It supports many advanced features like vocabulary versioning and user access control. However, its interface is arguably hard to use even for people with Semantic Web background. Also, it does not support resource management.

iQvoc (Bandholtz et al., 2010) is an open source Web-based vocabulary management tool for SKOS-XL (SKOS eXtension for Labels) which supports publication of glossaries, multilingual labels, and a simple term management workflow focused on the end user. It internally stores data in a relational database, but is able to provide vocabularies as Linked Data as well thanks to integration with a triple store. It does not address resource management.

Skosmos (Suominen et al., 2015) is a tool for publishing and exploring SKOS vocabularies. It does not allow to directly edit the vocabularies, it rather imports a SKOS-compatible vocabulary and provides access to it. It concentrates on searching for the most appropriate terms and providing Linked Data access to the concepts. Multilingual concept labels are supported, but resource annotation is not.

Protégé (Musen, 2015) is a desktop ontology editor allowing to model all aspects of a vocabulary. It also supports using a reasoner for knowledge base consistency validation. One of Protégé's biggest benefits is its large community and modular design (over one hundred plugins). Recently, *WebProtégé* (Tudorache et al., 2013) was introduced as a collaborative ontology engineering platform, featuring, e.g., concept tagging. Once again, while the breadth of features allows users of Protégé to maintain both glossaries and models, the user interface for editing the model is relatively cumbersome and knowledge of the underlying Semantic Web principles is necessary.

PoolParty Semantic Suite[9] is a commercial set of tools mainly focused on workflow-backed design and management of vocabularies based on text corpora. It provides additional services such as resource annotation, Linked Data management, semantic search, text mining, data integration, analytics and visualization.

TopBraid Composer[10] and other applications from the TopBraid family are commercial tools which allow, among other things, to create and manage semantic vocabularies and use them to annotate resources. The annotation is automatic and uses terms from vocabularies managed by the platform. Vocabulary glossaries are based on SKOS, but TopBraid also allows to maintain the corresponding vocabulary models.

Finland approaches public administration vocabulary management by developing proprietary tools

[8] Available at https://kbss.felk.cvut.cz/termit-demo, accessed 2020-03-15.

[9] https://www.poolparty.biz/, accessed 2020-01-23.

[10] https://www.topquadrant.com/products/topbraid-composer/, accessed 2020-01-23.

which are similar to TermIt.[11] They involve thesaurus management as well as UML modeling. The user interface seems more enhanced than that of TermIt, but the tool does not seem to have support for semi-automated vocabulary construction based on existing documents or resource annotation. Also the ontology modeling part is based on pure UML class diagrams, without any grounding in top-level ontologies, thus, for instance, lacking the capability to express the examples of meaning ambiguities mentioned in the introduction of this paper.

4.2 Resource Annotation and Search

Enterprise search is a vast domain with many commercial and non-commercial vendors. However, most of the tools provide keyword-based search which may give imprecise results due to the lack of explicit semantics of the keywords associated with the resources. Various vendors, for example, Google, Microsoft, IBM, Oracle, provide such search as a feature of their products. On the other hand, there exist specialized search solutions. Among the best known is *Apache Lucene*[12] – a text search engine library. Lucene is the foundation of many other popular solutions, including *Apache Solr*[13] and *Elasticsearch*.[14]

Indexing and search based on semantic terms associated with resources provide, for example, the aforementioned PoolParty Semantic Suite and TopBraid. Semantic technologies are also partially supported by the Rosette Text Analytics platform,[15] although the amount of relevant information for this commercial tool is limited.

5 EXPERIENCE

The original goal of TermIt was to build an easy-to-use glossary editor based on Semantic Web technologies. Text analysis was supposed to be used to help build new vocabularies from normative documents. Over time, it turned out that resource annotation using existing terms is as important a goal as creation of new vocabularies.

Thus, two main scenarios for using TermIt to build and manage vocabularies and use them in connection with resources have been identified:

Vocabulary Creation. In this scenario, the users start with an empty vocabulary and a normative document relevant to the vocabulary. They use Annotace to discover possible new terms in the document, review the suggestions, and create the new terms. Additional new terms are created manually.

Resource Annotation. Resource annotation assumes that a vocabulary with terms already exists. These terms are assigned, either manually, or using Annotace, to resources registered in the system. TermIt can then be used for semantic search – finding resources based on terms instead of words, getting related resources, etc.

TermIt has been used in two domains – urban planning and healthcare – and is being tested for application in another important domain – public administration. Vocabularies of laws and office agendas would be created and maintained through TermIt at the Ministry of the interior of the Czech Republic.

So far, several relevant features have been identified as lacking in TermIt:

Term creation workflow representing a term's lifecycle. In many domains, new terms are not created based on a single preexisting normative resource. They emerge from a collaborative effort of multiple users, who need to discuss the meaning of the term and eventually arrive at an agreement acknowledged by an authorized supervisor.

Synonyms and alternative labels allow users to gather alternative textual representations of a term.

Multilingual labels are required in cases where terms exist in an international context. For instance, a term like *Network administration* from a public administration vocabulary may have its origin in an ISO standard, but a local translation is needed for usability with resources not written in English.

6 CONCLUSIONS

We have presented TermIt – a practical open source vocabulary manager intended for domain experts with possibly limited knowledge of the Semantic Web. The system provides the ability to manage SKOS-based glossaries, annotate resources with terms, link the terms to their mentions in documents, and perform semantic search based on the annotations. Moreover, the ontological model backing TermIt is rooted in a top-level formalism – UFO – improving its interoperability.

[11] https://yhteentoimiva.suomi.fi/en/, accessed 2020-03-15.
[12] https://lucene.apache.org/, accessed 2020-03-15.
[13] https://lucene.apache.org/solr/, accessed 2020-03-15.
[14] https://www.elastic.co/, accessed 2020-03-15.
[15] https://www.basistech.com/text-analytics/rosette/, accessed 2020-03-15.

TermIt is still under development and there are several important features planned in the near future. One is the ability to link the definition of a term directly to a part of a normative file. The other is support for accessing remote vocabularies, so that a distributed network of TermIt instances can be created. This is especially important for the public administration case, where different departments (or even ministries) maintain their own sets of vocabularies which often depend on other vocabularies under a different jurisdiction. Other important tasks include term creation and approval workflow, vocabulary versioning, user access control, visualization of relationships between terms, and support for multilingual attributes.

ACKNOWLEDGEMENTS

This work was supported by grant No. SGS19/110/OHK3/2T/13 Efficient Vocabularies Management Using Ontologies of the Czech Technical University in Prague.

REFERENCES

Adida, B., Birbeck, M., McCarron, S., and Herman, I. (2015). RDFa Core 1.1 – Third Edition. W3C recommendation, W3C. https://www.w3.org/TR/rdfa-core/, accessed 2020-02-03.

Bandholtz, T., Schulte-Coerne, T., Glaser, R., Fock, J., and Keller, T. (2010). iQvoc - Open Source SKOS(XL) Maintenance and Publishing Tool. In *Proceedings of the Sixth Workshop on Scripting and Development for the Semantic Web, Crete, Greece, May 31, 2010*. http://ceur-ws.org/Vol-699/Paper2.pdf, accessed 2020-01-23.

Blaško, M., Kostov, B., and Křemen, P. (2016). Ontology-based dataset exploration – a temporal ontology use-case. In *Proceedings of the Intelligent Exploration of Semantic Data (IESD'16)*.

Brickley, D. and Guha, R. (2014). RDF Schema 1.1. W3C recommendation, W3C. https://www.w3.org/TR/rdf-schema/, accessed 2020-01-20.

Cyganiak, R. and Bizer, C. (2007). Pubby – A Linked Data Frontend for SPARQL Endpoints. online. http://wifo5-03.informatik.uni-mannheim.de/pubby/, accessed 2020-01-23.

Cyganiak, R., Wood, D., and Lanthaler, M. (2014). RDF 1.1 Concepts and Abstract Syntax. W3C recommendation, W3C. http://www.w3.org/TR/rdf11-concepts/, accessed 2020-01-20.

Guarino, N., Oberle, D., and Staab, S. (2009). *What Is an Ontology?*, chapter 1, pages 1–17. Springer-Verlag Berlin Heidelberg.

Guizzardi, Giancarlo (2005). *Ontological Foundations for Structural Conceptual Models*. PhD thesis, University of Twente.

Křemen, P. and Nečaský, M. (2019). Improving discoverability of open government data with rich metadata descriptions using semantic government vocabulary. *Journal of Web Semantics*, 55:1 – 20.

Libovický, J. (2016). KER - keyword extractor. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Mascardi, V., Cordì, V., and Rosso, P. (2007). A Comparison of Upper Ontologies. In *WOA 2007: Dagli Oggetti agli Agenti. 8th AI*IA/TABOO Joint Workshop "From Objects to Agents": Agents and Industry: Technological Applications of Software Agents, 24-25 September 2007, Genova, Italy*.

Miles, A. and Bechhofer, S. (2009). SKOS Simple Knowledge Organization System Reference. W3C Recommendation, W3C. http://www.w3.org/TR/skos-reference, accessed 2020-01-20.

Motik, B., Patel-Schneider, P. F., and Parsia, B. (2012). OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax (Second Edition). W3C recommendation, W3C. https://www.w3.org/TR/owl2-syntax/, accessed 2020-01-20.

Musen, M. A. (2015). The Protégé Project: A Look Back and a Look Forward. *AI Matters*, 1(4):4–12.

Rumbaugh, J., Jacobson, I., and Booch, G. (2004). *Unified Modeling Language Reference Manual, The (2nd Edition)*. Pearson Higher Education.

Stellato, A., Rajbhandari, S., Turbati, A., Fiorelli, M., Caracciolo, C., Lorenzetti, T., Keizer, J., and Pazienza, M. T. (2015). VocBench: A Web Application for Collaborative Development of Multilingual Thesauri. In *Proceedings of the 12th European Semantic Web Conference on The Semantic Web. Latest Advances and New Domains - Volume 9088*, pages 38–53, New York, NY, USA. Springer-Verlag New York, Inc.

Suominen, O., Ylikotila, H., Pessala, S., Lappalainen, M., Frosterus, M., Tuominen, J., Baker, T., Caracciolo, C., and Retterath, A. (2015). Publishing SKOS vocabularies with Skosmos. Manuscript submitted for review.

Tudorache, T., Nyulas, C., Noy, N. F., and Musen, M. A. (2013). WebProtégé: A Collaborative Ontology Editor and Knowledge Acquisition Tool for the Web. *Semantic Web*, 4(1):89–99.

Wood, D., Zaidman, M., Ruth, L., and Hausenblas, M. (2013). *Linked Data: Structured Data on the Web*. Manning Publications Co., Shelter Island, NY, USA.