MANAGING MULTIMODAL AND MULTILINGUAL SEMANTIC CONTENT

Michael Martin[1], Daniel Gerber[2], Norman Heino[1], Sören Auer[1] and Timofey Ermilov[1]
AKSW/Computer Science Institut, University of Leipzig, Postfach 100920, 04009 Leipzig, Germany

Keywords: Knowledge management, Semantic web, Multimodality, Multilinguality, Semantic Wiki.

Abstract: With the advent and increasing popularity of Semantic Wikis and the Linked Data the management of semantically represented knowledge became mainstream. However, certain categories of semantically enriched content, such as multimodal documents as well as multilingual textual resources are still difficult to handle. In this paper, we present a comprehensive strategy for managing the life-cycle of both multimodal and multilingual semantically enriched content. The strategy is based on extending a number of semantic knowledge management techniques such as authoring, versioning, evolution, access and exploration for semantically enriched multimodal and multilingual content. We showcase an implementation and user interface based on the semantic wiki paradigm and present a use case from the e-tourism domain.

1 INTRODUCTION

With the advent and increasing popularity of Semantic Wikis and Linked Data the management of semantically represented knowledge became mainstream. Oracle, for example, integrated support for semantic knowledge management into their database product (Lopez and Das, 2009), Google started to evaluate annotations using Resource Description Framework attributes (RDFa) and the W3C has lately launched the second revision of the Web Ontology Language (OWL) standard. However, despite this progress certain categories of semantically enriched content, such as multimodal documents as well as multilingual textual resources are still difficult to handle.

Currently knowledge bases primarily contain typed data and a limited amount of textual content, such as short labels, short descriptions or small hypertext fragments. With the increasing maturity of semantic technologies and their wider use in many different application scenarios the representation and interlinking of metadata for multimodal content such as audio, video, compound hypertext or multimedia documents is becoming paramount. Another crucial feature of semantic knowledge representation is the language independence. Ontologies, taxonomies or simple resource descriptions can be easily equipped with multilingual texts and labels. However, the translation and life-cycle of multilingual semantic content is currently insufficiently supported. Examples for the importance of supporting multimodal and multilingual semantic content are in particular, bio-medical semantic information systems and semantics based Web Content Management.

In this paper, we present a comprehensive strategy for managing the lifecycle of both multimodal and multilingual semantically enriched content. The strategy is based on extending a number of semantic knowledge management techniques such as authoring, versioning, evolution, access and exploration for semantically enriched multimodal and multilingual content. With regard to multimedia content we devise a strategy for extracting, semantically representing and interlinking metadata of multimedia documents. For the management of multilingual knowledge bases we developed techniques for supporting the lifecycle of multilingual resources by enabling an efficient semi-automatic translation of individual property values, resources or all textual content stored within a knowledge base. For keeping textual content in a knowledge base in the preferred language in sync with translations into other languages we devise a strategy based on capitalizing the integrated versioning of the Semantic Data Wiki OntoWiki. We showcase an implementation and user interface and present a use case from the e-tourism domain.

The paper is structured as follows: We describe a number of important aspects for managing semantic content in Section 2. We outline our strategy for dealing with large quantities of multimodal content

Martin M., Gerber D., Heino N., Auer S. and Ermilov T..
MANAGING MULTIMODAL AND MULTILINGUAL SEMANTIC CONTENT.
DOI: 10.5220/0003336202900295
In *Proceedings of the 7th International Conference on Web Information Systems and Technologies* (WEBIST-2011), pages 290-295
ISBN: 978-989-8425-51-5

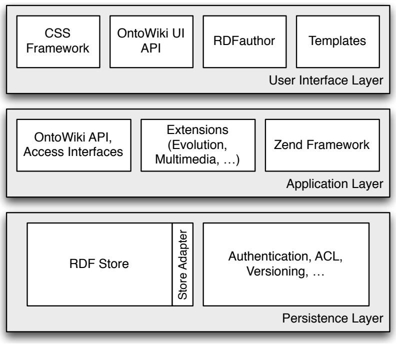


Figure 1: OntoWiki framework architecture.

in Section 3. In Section 4 we present the strategy for supporting the life-cycle of multilingual resources. We showcase an application scenario in Section 5, where both – multimodal and multilingual – strategies for semantic content management are successfully applied.

2 MANAGEMENT OF SEMANTIC CONTENT

Our approach for managing multimodal and multilingual semantic content is based on OntoWiki (Auer et al., 2006). It started as an RDF-based data wiki with emphasis on collaboration but has meanwhile evolved into a comprehensive framework for developing Semantic Web applications (Heino et al., 2009). This involved not only the development of a sophisticated extension interface allowing for a wide range of customizations but also the addition of several access and consumption interfaces allowing OntoWiki installations to play both a provider and a consumer role in the emerging Web of Data. In the sequel we discuss OntoWiki extensions that particularly facilitate authoring and management of multimodal and multilingual data.

Authoring. Semantic content in OntoWiki is represented as resource descriptions. Following the RDF data model representing one of the foundations of the Semantic Web vision, resource descriptions are represented (at the lowest level) in the form of *statements*. Each of these statements (or triples) consist of a *subject* which identifies a resource as well as a *predicate* and an *object* which together represent data about said resource in a fashion reminiscent of key-value pairs. By means of *RDFa*, these statements are retained in the HTML view (i.e. user interface) part and are thus accessible to client-side techniques like JavaScript. Authoring of such content is based on said client-side representation by employing the *RDFauthor* approach (Tramp et al., 2010b): views are declared in terms of the model language (RDF) which allows the underlying model be restored. Based on this model, a user interface can be generated with the model being providing all the domain knowledge required to do so. The RDFauthor system provides an extensible set of authoring widgets specialized for certain editing tasks. In the work at hand, we extended the system by adding capabilities for automatically translating literal object values. Since the semantic context is known to the system, these translation functionality can be bound to arbitrary characteristics of the data (e. g. to a certain property or a missing language).

Versioning. Keeping track of all changes is an important task in order to encourage user participation. OntoWiki applies this concept to RDF-based knowledge engineering in that all changes are tracked on the statement level (Auer and Herre, 2006). These low-level changes can be grouped to reflect application- and domain-specific tasks involving modifications to several statements as a single versioned item. Provenance information as well as other metadata (such as time, user or context) of a particular changeset can be attached to each individual changeset. All changes on the knowledge base can be easily reviewed and rolled-back if needed.

Evolution. The loosely typed data model of RDF encourages continuous evolution and refinement of knowledge bases. With *EvoPat*, OntoWiki supports this in a declarative, pattern-based manner (Rieß et al., 2010). Basic evolution patterns consist of three components (1) a set of variables, (2) a SPARQL select query selecting a number of resources under evolution, (3) a SPARQL/Update query template that is executed for each resulting resource of the select query. In addition, basic patterns can be combined to form compound patterns—suitable for more complex evolution scenarios. In order to facilitate the semi-automatic application of evolution patterns, *bad smells* can be defined that serve as a detection mechanism for ontology design anti-patterns or data modeling problems. If certain conditions are met, this process is even fully automatable.

Access and Exploration Interfaces. In addition to human-targeted graphical user interfaces, OntoWiki supports a number of machine-accessible data interfaces: OntoWiki implements a *SPARQL Endpoint*, allowing all resources managed in an OntoWiki be queried over the Web. According to accepted *Linked*

Data publication principles, OntoWiki makes all resources accessible by its URI. Furthermore, for each resource used in OntoWiki additional triples can be fetched if the resource is dereferenceable. Pingback is an established notification system that gained wide popularity in the blogsphere. OntoWiki adapts the pingback idea known from the blogsphere to Linked Data providing a *notification mechanism* for resource usage (Tramp et al., 2010a). If a *Semantic Pingback*-enabled resource is mentioned (i. e. linked to) by another party, its pingback server is notified of the usage.

For exploring semantic content, OntoWiki provides several exploration interfaces: The compromise of, providing a generic user interface aiming at being as intuitive as possible is tackled by regarding knowledge bases as *information maps*. Each node at the information map, that is, RDF resource, is represented as a Web accessible page and interlinked to related digital resources. The *full-text search* makes use of special indexes if the underlying knowledge store provides this feature.The resulting SPARQL query is stored as an object which can later be modified (e. g. have its filter clauses refined). For *domain-specific use cases*, OntoWiki provides an easy-to-use *extension interface*. By providing such a custom view, it is possible to hide the fact an RDF knowledge base is being worked on. This permits OntoWiki to be used as a data-entry frontend for users with a less profound knowledge of semantic technologies. Via its *facet-based browsing*, OntoWiki allows the construction of complex concept definitions, with a pre-defined class as a starting point by means of property value restrictions.

3 MULTIMODAL SEMANTIC CONTENT

For handling large amounts of multimedia data, automatic processes for managing this kind of content have been developed and integrated into OntoWiki. They allow to import arbitrary multimedia documents (13 different file types are currently supported) or even complete directory structures into a knowledge base and manage them subsequently with OntoWiki, using the techniques presented in Section 2. The workflow for importing multimedia documents is presented in Figure 2 and described in the sequel.

Extracting Multimedia Metadata. We developed a framework, which detects certain formats (from the more than 1000 different registered MIME types).

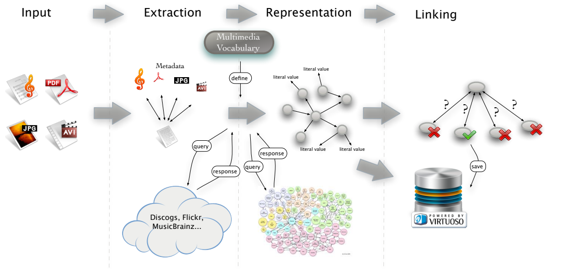


Figure 2: Multimedia metadata extraction, representation and interlinking process.

The framework is highly configurable and easily extensible, thus allowing to easily integrate support for new multimedia types and to configure the properties and classes used to create the semantic metadata. The extraction of multimedia metadata is realized as follows:

1. Extraction of Metadata Attributes. Information about the file name, size or date of creation is extracted. In addition to those information, many multimedia formats already contain metadata specific to their field of use. Such information is most likely arranged in key-values pairs in the file's header. For instance, music files usually contain *ID3 tags*, images taken by digital cameras include an *EXIF* header. The MIME type of the file is determined and subsequently a specialized metadata extractor is initialized. The framework is designed in a way that every metadata extractor manages a set of extensions, each one being responsible for the extraction of a single metadata type on its own. These extensions are executed consecutively, thus giving the opportunity to re-use already extracted metadata and accelerate the extraction process.
Previews of PDF or video documents are created. Other examples of metadata extraction extensions are the number of pages of a PDF document or the geo-coordinates of an image.

2. Integration of Additional Information. The previously extracted metadata is now used to obtain and integrate additional information, which is not explicitly contained in the processed files. For example, an artists name extracted from the music's file ID3 information may be used to look up a URI for this artist on the Data Web. Likewise, traditional non-RDF based web-services may be used to gather additional information (e.g. the album cover for a song).

Representing Multimedia Metadata. To represent the extracted metadata in RDF we reused well established vocabularies (cf. Figure 3). The rationale

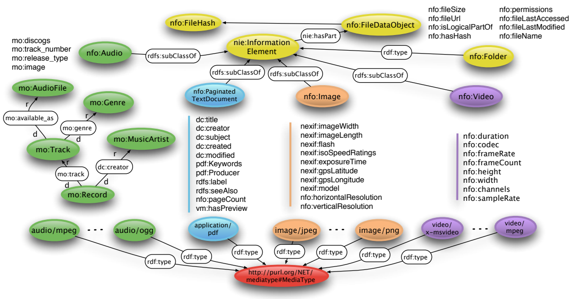


Figure 3: Vocabulary for representing multimedia metadata.

of this representation is the separation between the actual data (`nfo:FileDataObject`) and its interpretation (`nie:InformationElement`) borrowed from the NEPOMUK information element ontology (NIE) and the NEPOMUK file ontology (NFO)[1]. The large number of subclasses of `nie:InformationElement` makes it possible to classify most of the common multimedia types. To interpret a `DataObject`, it needs to be seen in the context of a corresponding `InformationElement` or as one of its subclasses. This is achieved with the properties `nie:hasPart` (or its inverse `nie:isPartOf`). This approach ensures that even complex data structures, like archives in the attachment of emails are processed correctly.

In order to describe the individual `InformationElements` further, the following list of vocabularies has been chosen: For describing **audio documents** the music ontology (Raimond et al., 2007) is used (namespace prefix `mo`), since it allows to represent all information available in ID3, as well as concepts such as *concert* or *festival*. For describing **PDF documents** elements of various vocabularies such as *Dublin Core* (`dc`), NFO and RDFS are used. For describing **images** the NEPOMUK EXIF ontology is used. Currently, we primarily extract low-level information from **videos** (e. g. frame-rate or the used video codec) which are represented using the NEPOMUK file ontology. URIs used to describe the file's MIME type are provided by the *mediatypes* application[2].

Interlinking Metadata. The extracted and RDF represented metadata can now be used to find and create links to arbitrary other resources. We developed the *OntoWiki Linking Module*, which is able to discover possible links between any sort of resources, in particular multimedia documents. The semi-automatic algorithm used to locate the resources for linking can be divided in the following four parts, taking into account that the starting point of this algorithm is an arbitrary resource *r*: (1) Find all properties with the `rdfs:range` or `rdfs:domain` of the type (`rdf:type`) of resource *r*. (2) Created a list of all resources, which are accessible via those properties. (3) All resources found in step two are now compared to *r*. The comparison takes place with the help of string attributes (e. g. `dc:title` or `rdfs:label`). As metric to calculate the probability of interlinking we use the *effectiveness measure*, i. e. the weighted harmonic mean between precision and recall. (4) The found resources are now sorted by their probability of linking to *r* and presented to the user (grouped by the property). For linking multimedia documents with arbitrary resources, we developed vocabularies which are aligned to the different MIME type categories (i.e. application, audio, video, text and image).

4 MULTILINGUAL SEMANTIC CONTENT

The life-cycle of multilingual semantic content (cf. Figure 4) usually starts with the creation and authoring of a semantic resource. Once created textual content can be translated. Subsequently, the original language content attached to the resource might be revised, which has to trigger a revision of the translations as well. The translation of properties of RDF resources in multiple languages is realized with language tags, which can be attached to the string literal property values. Most RDF resources contain at least one label in a preferred language. For the semi-automated translation of literal values we employ the *Google Translation Service* API[3]. It supports the translation between more than 50 languages as well as an automated language detection, which helps if source RDF literal values are not annotated with a language tag. Since not all properties contain translatable content as well as not all literal values are suitable sources for translation, users are able to configure translatable property URIs and possible source languages. In the sequel we sketch OntoWiki extensions for language resource translation and management.

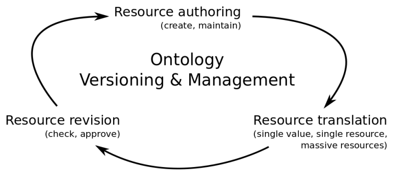
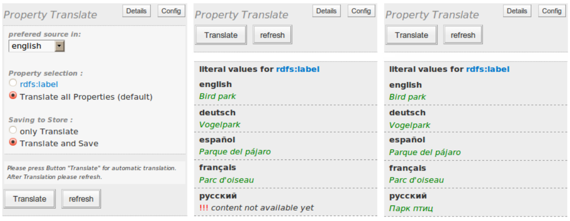


Figure 4: Life-cycle of multilingual resources.

[1] NIE, NOE and NEXIF are available from: http://www.semanticdesktop.org/ontologies/

[2] http://mediatypes.appspot.com

[3] http://code.google.com/apis/ajaxlanguage/



prefered source in: english

Property selection :
○ rdfs:label
● Translate all Properties (default)

Saving to Store :
○ only Translate
● Translate and Save

Please press Button "Translate" for automatic translation.
After Translation please refresh.

literal values for rdfs:label
english
Bird park
deutsch
Vogelpark
español
Parque del pájaro
français
Parc d'oiseau
русский
!!! content not available yet

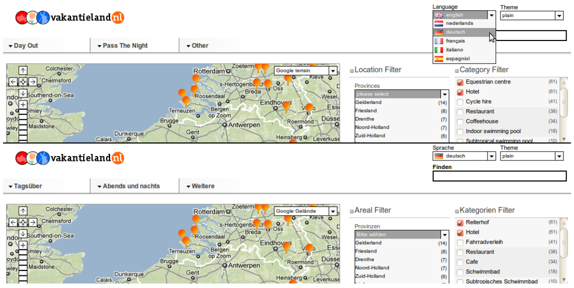
literal values for rdfs:label
english
Bird park
deutsch
Vogelpark
español
Parque del pájaro
français
Parc d'oiseau
русский
Парк птиц

Figure 5: Individual resource translation extension.

Single Resource Translation. To improve the translation support of single resources we integrated the *RDFauthor extension* and the *Individual resource translation extension* in OntoWiki. The first extension auto-suggest translations while editing single literal values. The second extension can be deployed to translate configured parts of complete RDF resources automatically (cf. Figure 5).

Massive Translation Component. In order to translate RDF resources massively, we developed the *Massive translation component* for OntoWiki, which is operating similar to the *Individual resource translation extension*. By using SPARQL, this component requests a set of resources with missing translations. The result is fulfilled automatically by using the translation service and presented to the user as a HTML-form in the GUI. All literal values in the generated HTML-form are editable, due to improve translations manually afterwards. After applying and saving the new translations, this algorithm is looped until all resources contain the specified amount of translated literal values.

Multi-lingual Resource Versioning and Revision. Every change between its creation and deletion of an RDF resource is tracked by OntoWikis versioning component described in Section 2. This versioning component is also used to store information about the translation process. After changing one of the literal values, translations to other languages could be affected. As a consequence, a special entry marking a translation process is stored to the versioning repository. As being depicted in Figure 4, every translation might have to be revised. This flag is used to notify the content author for approving the correctness of all other translated literal values of the particular property of the selected RDF resource. After approving the correctness a further flag is stored to represent an acceptable translation state of the RDF resource.



Provinces
please select
Gelderland (14)
Friesland (8)
Drenthe (7)
Noord-Holland (7)
Zuid-Holland (6)

☑ Equestrian centre (61)
☑ Hotel (61)
□ Cycle hire (41)
□ Restaurant (38)
□ Coffeehouse (34)
□ Indoor swimming pool (18)
□ Subtropical swimming pool (10)

Provinzen
Bitte wählen
Gelderland (14)
Friesland (8)
Drenthe (7)
Noord-Holland (7)
Zuid-Holland (6)

☑ Reiterhof (61)
☑ Hotel (61)
□ Fahrradverleih (41)
□ Restaurant (38)
□ Cafe (34)
□ Schwimmbad (18)
□ Subtropisches Schwimmbad (10)

Figure 6: Excerpt of the Vakantieland user interface.

5 USE CASE: VAKANTIELAND

Both our semantic content management strategies were applied and evaluated in the Semantic Web application *Vakantieland*[4]. Vakantieland publishes comprehensive information about 20,000 touristic points-of-interest (POI) in The Netherlands such as textual descriptions, location information and opening hours. The information is stored in a knowledge base containing almost 2 million triples. It is structured using approx. 1,250 properties as well as 400 classes. Vakantieland was designed according to the model/view/controller principle and uses OntoWiki's Erfurt API as middleware. Almost all of the information presented in Vakantieland is retrieved using SPARQL.

Multimedia Management in Vakantieland. In this use case we applied the multimedia management process, presented in Section 3, to create RDF resources for about 850 PDF documents (i.e. info brochures of POIs) and interlink them accordingly. In particular, we extended the generic multimedia linking vocabulary, in order to specify the `rdfs:domain` to POIs and evaluated the OntoWiki Linking Module. For 100 randomly chosen documents the suggestions of this module have been compared to manually assigned links, created by a domain expert. This evaluation has shown, that for 80% of the documents, the correct suggestion – the POI with the highest probability – was found. The other way around (i.e. suggesting documents for POIs) it was even possible to find the correct one in 90% of the cases. The created links are then used to display those documents, or any other document type for that matter, and additional information like the document's title (translated in various languages), on a POI's details page.

[4] Available at: http://staging.vakantieland.nl

Multilinguality Management in Vakantieland. In order to present the tourism content of Vakantieland in multiple languages as depicted in Figure 6, we translated *rdfs:labels* and *rdfs:comments* of classes, properties and instances. At this time, the tourism RDF content of Vakantieland contains information encoded in different languages such as Dutch, English, French, German, Italian and Spanish. We measured our approach manually by using the *Individual resource translation extension* to translate 50 randomly chosen property labels. In comparison to manually translating these property labels from German to English the usage of the translation extension reduced the required time by more than 80%.

6 CONCLUSIONS

With the increasing maturation of semantic technologies the facilitation of multimodal and multilingual semantic content management became a crucial requirement. In this article we presented two complementary strategies for such content based on the semantic wiki paradigm. Both strategies are based on supporting the lifecycle of respective semantic content. With regard to future work we deem that work with regard to the integration of automatic linking techniques, fine-grained provenance tracking, and facilitation of adaptive previews is promising.

REFERENCES

Auer, S., Dietzold, S., and Riechert, T. (2006). OntoWiki – A Tool for Social, Semantic Collaboration. ISWC, LNCS.

Auer, S. and Herre, H. (2006). A Versioning and Evolution Framework for RDF Knowledge Bases. PSI.

Heino, N., Dietzold, S., Martin, M., and Auer, S. (2009). Developing Semantic Web Applications with the OntoWiki Framework. In *Networked Knowledge - Networked Media*, Studies in Computational Intelligence.

Lopez, X. and Das, S. (2009). Oracle Database 11g Semantic Technologies, Semantic Data Integration for the Enterprise, White Paper. Technical report, Oracle Semantic Technologies Center.

Raimond, Y., Abdallah, S. A., Sandler, M., and Giasson, F. (2007). The music ontology. ISMIR.

Rieß, C., Heino, N., Tramp, S., and Auer, S. (2010). EvoPat – Pattern-Based Evolution and Refactoring of RDF Knowledge Bases. ISWC, LNCS.

Tramp, S., Frischmuth, P., Ermilov, T., and Auer, S. (2010a). Weaving a Social Data Web with Semantic Pingback. EKAW, LNAI.

Tramp, S., Heino, N., Auer, S., and Frischmuth, P. (2010b). Rdfauthor: Employing RDFa for collaborative knowledge engineering. EKAW, LNAI.