Proceedings of the Fifth Workshop on Semantic Wikis – Linking Data and People

7th Extended Semantic Web Conference Hersonissos, Crete, Greece, June 2010

edited by Jochen Reutelshöfer

May 31, 2010

Preface

Dear Reader, the semantic wiki community is meeting again to have their 5^{th} workshop. For the third year in a row (and fourth time in total) the SemWiki workshop adds the charm of the wiki spirit to the ESWC, thus forming an established event at the conference. The submissions we received are addressing a high diversity of semantic wiki related topics and 19 of them will be presented at the workshop. While most of the contributions again came from European countries, we are glad about semantic wiki research activities in South America, as we also received papers from Chile and Argentina. Beside general semantic wiki topics like knowledge representation, reasoning, refactoring and user interaction, a large number of papers reported on concrete applications in real world domains, as for example Astronomy, Archeology/History, Biology and Pharmaceutics. We think that valuable experiences reported from these projects will help to employ the semantic wiki approach in real world projects even more easily and successfully. Further, we hope that some insights will also be helpful in the larger context of the general semantic web. We wish to thank *all* authors and reviewers who spent a lot of work to contribute to this topic and thereby made this workshop possible. Many thanks also to the ESWC organisation team, which set the stage for this workshop as one out of 9. We are confident that this workshop will again bring semantic wikis one step forward considering research, tools and applications.

May 2010

Christoph Lange, Sebastian Schaffert, Hala Skaf-Molli and Jochen Reutelshöfer

About the Workshop

Wikis are social software transforming visitors into collaborators. Semantic wikis are social semantic software that combines the most salient aspects of wikis with technologies from the Semantic Web. They play an important role in the construction of the social semantic web. They have the mission to gather humans and computers in order to build together the next wave of lightweight ontologies.

Semantic wikis have grown up. Foundational research on them is done in large projects, some enterprise systems are sold commercially, and certain established systems have evolved into operating system like platforms for semantic social software. Classical wikis are starting to adopt basic concepts of semantic wikis, and the "wiki spirit", including easy collaboration and linking of knowledge, is found in more and more innovative applications, such as Google Wave.

Recently, the focus of semantic wiki research has shifted from proofs of concept and hacks to real-world use cases. Besides evaluations of such use cases, foundational research and technical innovation are still needed, as the large-scale application of semantic wikis has unveiled a number of research questions that the academic community now has to answer in a consolidated effort.

The aim of this fifth SemWiki workshop is to exchange ideas, to discuss pressing research questions arising from practical usage of semantic wikis, and to explore integrations of wikis with other semantic web technologies. The outcome of the workshop will be a collection of open research questions, an overview of the structure of the research area (open space session, to be documented at semanticweb.org), experience reports (what works in semantic wikis and what does not yet?), and a state-of-the-art overview of applications of semantic wikis. As semantic wikis contain many of the core Semantic Web challenges in an integrated fashion, they act as "Petri dishes" for the semantic web; thus, the results obtained in this workshop will have a wider impact on Semantic Web research and Web Science.

Previous semantic wiki workshops took place at ESWC 2006, WikiSym 2006, ESWC 2008 and ESWC 2009. More information about these workshops can be found on the SemWiki workshop homepage (*http://www.semwiki.org*).

Contents

Programme

Sunday Night (May 30th): SemWiki Social Event, including ...
Award Ceremony of the INSEMTIVES Gaming Idea Challenge
and Announcement of the KiWi Application Challenge
time and location to be announced

Workshop (Monday, May 31st, room: POLYMNEA):

09:00 – 10:30 Session 1: Opening, Keynote and Best Paper Award

09:00 – 09:10 Opening Ceremony
Christoph Lange, Sebastian Schaffert, Hala Skaf-Molli, and Jochen Reutelshöfer

09:10 – 10:00 Keynote: The limits of Semantic Wikis – and beyond
Denny Vrandečić, KIT, Karlsruhe
—discussion—

10:00 – 10:30 **Best Paper**
PEST: Term-Propagation over Wiki-Structures as Eigenvector Computation
Klara Weiand, Fabian Kneissl, Tim Furche and François Bry
—discussion—

10:30 – 11:00 **Coffee Break**

11:00 – 13:00 Session 2: Lightning Panels

11:00 – 12:00 **Content generation and enhancement**

Jump-starting a Body-of-Knowledge with a Semantic Wiki on a Discipline Ontology
Víctor Codocedo, Claudia Lopez and Hernan Astudillo

TasTicWiki: A Semantic Wiki with Content Recommendation *(short presentation)*
Manuela Ruiz-montiel, Joaquín J. Molina-Castro and José F. Aldana-Montes

—discussion—

Ideator - a collaborative enterprise idea management tool powered by KiWi *(short presentation)*
Rolf Sint, Mark Markus, Sebastian Schaffert and Thomas Kurz

Towards Meta-Engineering for Semantic Wikis
Jochen Reutelshoefer, Joachim Baumeister and Frank Puppe

—discussion—

12:00 – 13:00 **e-science**

Access and Annotation of Archaeological Corpus via a SemanticWiki *(short presentation)*
Eric Leclercq and Marinette Savonnet

Collaborative Editing and Linking of Astronomy Vocabularies Using Semantic Mediawiki *(short presentation)*
Stuart Chalmers, Norman Gray, Iadh Ounis and Alasdair Gray

—discussion—

A Wiki-Oriented On-line Dictionary for Human and Social Sciences *(short presentation)*
Khelifa Lydia, Ilham Nadira Lammari, Hammou Fadili and Jacky Akoka

Automating Content Generation for Large-scale Virtual Learning Environments using Semantic Web Services *(short presentation)*
Ian Dunwell

—discussion—

13:00 – 14:30 **Lunch**

14:30 – 15:30 Session 3: Lightning Panels

14:30 – 15:00 **Bio-medical Applications**

Lab Service Wiki: a wiki-based data management solution for protein production service *(short presentation)*
Antoni Hermoso Pulido, Michela Bertero, Silvia Speroni, Miriam Alloza and Guglielmo Roma

OpenDrugWiki – Using a Semantic Wiki for Consolidating, Editing and Reviewing of Existing Heterogeneous Drug Data *(short presentation)*
Anton Köstlbacher, Jonas Maurus, Rainer Hammwöhner, Alexander Haas, Ekkehard Haen and Christoph Hiemke

—discussion—

15:00 – 15:30 **Coordination and Maintenance**

Enhancing MediaWiki Talk pages with Semantics for Better Coordination - A Proposal *(short presentation)*
Jodi Schneider, Alexandre Passant and John Breslin

Semantic Wiki Refactoring. A strategy to assist Semantic Wiki evolution *(short presentation)*
Martin Rosenfeld, Alejandro Fernández and Alicia Díaz

15:30 – 16:00 Session 4: Demo & Poster Session

15:30 – 15:50 **Demo talks (5 min advertisement)**

DSMW: Distributed Semantic MediaWiki
Hala Skaf-Molli, Gérôme Canals and Pascal Molli

Annotation component for a Semantic Wiki
Marek Schmidt and Pavel Smrž

A Perfect Match for Reasoning, Explanation, and Reason Maintenance: OWL 2 RL and Semantic Wikis
Jakub Kotowski and François Bry

Connecting Semantic Mediawiki to different Triple Stores Using RDF2Go
Manfred Schied, Anton Köstlbacher and Christian Wolff

15:50 – 16:00 Open forum session: Posters and Demos (until 17:00)

Prototyping a Browser for a Listed Buildings Database with Semantic MediaWiki
Anca Dumitrache, Christoph Lange, Michael Kohlhase and Nils Aschenbeck

Human-machine Collaboration for Enriching Semantic Wikis using Formal Concept Analysis
Alexandre Blansché, Hala Skaf-Molli, Pascal Molli and Amedeo Napoli

x 16:00 – 16:30 **Coffee Break (poster/demo session open to visitors)**

16:30 – 18:00 Session 5: Discussion & Closing

16:30 – 17:00	**Poster and Demo session continued**
17:00 – 17:45	**Lightning talks & open discussion**
17:45 – 18:00	**Closing remarks**

Organisation

- Christoph Lange
- Sebastian Schaffert
- Hala Skaf-Molli
- Jochen Reutelshöfer

PEST: Term-Propagation over Wiki Structures as Eigenvector Computation

Klara Weiand, Fabian Kneißl, Tim Furche, and François Bry

Institute for Informatics, University of Munich,
Oettingenstraße 67, D-80538 München, Germany
http://www.pms.ifi.lmu.de/

Abstract. We present PEST, a novel approach to approximate querying of structured wiki data that exploits the structure of that data to propagate term weights between related wiki pages and tags. Based on the PEST matrix, eigenvectors representing the distribution of a term after propagation are computed. The result is an index which takes the document structure into account and can be used with standard document retrieval techniques. This article gives a detailed outline of the approach and gives first experimental results showing its viability.

1 Introduction

Mary wants to get an overview of software projects in her company that are written in Java and that make use of the Lucene library for full-text search. According to the conventions of her company's wiki, a brief introduction to each software project is provided by a wiki page tagged with *"introduction"*.

Thus, Mary enters the query for wiki pages containing *"java"* and *"lucene"* that are also tagged with *"introduction"*. In the semantic wiki KiWi, this can be achieved by the KWQL [5] query `ci(java lucene tag(introduction))`, where `ci` indicates wiki pages, see Section 3.2.

However, the results fall short of Mary's expectations for two reasons that are also illustrated in the sample wiki of Figure 1:

(1) Some projects may not follow the wiki's conventions (or the convention may have changed over time) to use the tag *"introduction"* for identifying project briefs. This may be the case for Document 5 in Figure 1. Mary could loosen her query to retrieve all pages containing *"introduction"* (rather than being tagged with it). However, in this case, documents that follow the convention are not necessarily ranked higher than other matching documents.

(2) Some projects use the rich annotation and structuring mechanisms of a wiki to split a wiki page into sub-sections, as in the case of the description of KiWi in Documents 1 and 2 from Figure 1, and to link to related projects or technologies (rather than discuss them inline), as in the case of Document 4 and 5 in Figure 1. Such projects are not included in the results of the original query at all. Again, Mary could try to change her query to allow keywords to occur in sub-sections or in linked to documents, but such queries quickly become rather

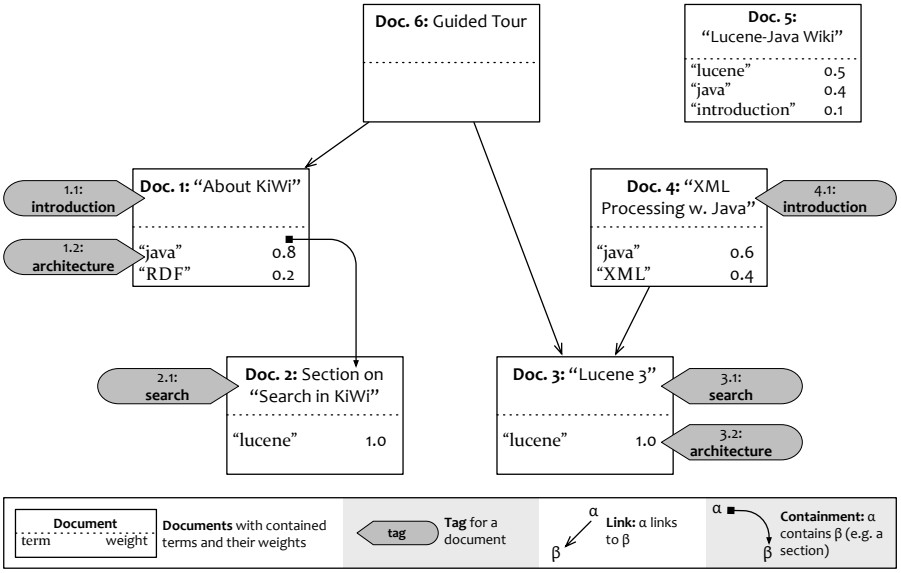


Fig. 1. Link and containment graph for a sample wiki

complex (even in a flexible query language such as KWQL). Furthermore, this solution suffers from the same problem as addressed above: documents following the wiki's conventions are not necessarily ranked higher than those only matched due to the relaxation of the query.

Fuzzy matching over words by means of, for example, stemming is an established technique widely used in Information Retrieval applications such as web search engines. Fuzzy matching over structure however, is only recently gaining attention as the amount of (semi-)structured data on the web increases. When a query explicitly imposes structural constraints on the selection, fuzzy matches are also returned where the structural constraints hold only approximately (e.g., a direct link is approximated by a chain of links).

In this article, we present PEST, short for term-**p**ropagation using **e**igenvector computation over wiki-**st**ructures, a novel approach to approximate or **fuzzy matching over structured data**. PEST is based on a unique technique for propagating term weights (as obtained from a standard vector-space representation of the documents) over the structure of a wiki using eigenvector computation. The eigenvector computation is inspired by, but differs significantly from, Google's PageRank [4].

In contrast to many other fuzzy matching approaches (see Section 2), PEST relies solely on modifying term weights in the document index and requires no runtime query expansion, but can use existing document retrieval technologies such as Lucene. Nevertheless, it is able to solve all above described problems in the context of the semantic wiki KiWi.

To illustrate how PEST propagates term weights, consider again Figure 1. As for PageRank, the "magic" of PEST lies in its matrix, called the PEST *propagation matrix*, or PEST matrix for short. The PEST matrix is computed in two steps:

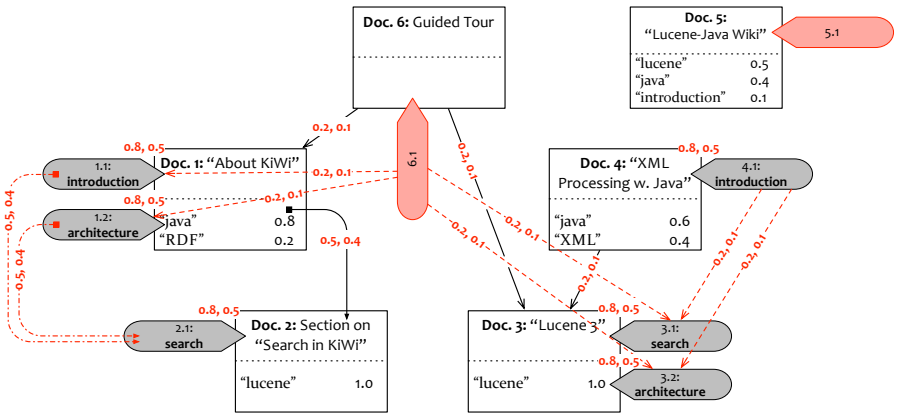


Fig. 2. Edge weights and virtual nodes and edges for Figure 1

(1) Weighted propagation graph: First, we extend and weight the graph of the wiki pages and their tags: These insertions are used to enable direct propagation between tags. Thus, we can configure how terms propagate between tags of related pages independently from term propagation between the pages.

The resulting graph for the sample wiki is shown in Figure 2. We have added the tags 5.1 and 6.1 and containment edges from tag 1.1 and 1.2 to tag 2.1, as well as link edges, e.g., from the tag 6.1 to tag $1.1, 2.1, 3.1$ and 3.2. In the following, we assign edge weights based solely on the type of the edge (link, containment, tagging).

(2) "Informed Leap": The weighted propagation graph, however, does not encode any information about the differences in *term distribution* in the original nodes, but only information about the structure of the wiki graph. To encode that information in the PEST matrix, we use an "informed leap": First, we transpose the weighted adjacency matrix of the weighted propagation graph and normalize it by the highest edge-weight sum over all documents (rather than for each document individually) to preserve differences in overall edge weight between documents. Second, the remaining probability together with a fixed leap parameter α (e.g., 30%) is used for an "informed leap" to an arbitrary node. The probability to go to a specific node A in such an "informed leap" is not random (as in the case of the original PageRank), but informed by the original term weight distribution: A page with a high term weight for term τ is more likely to be the target of a leap than a page with low term weight for τ.

The resulting matrix is called the PEST matrix $\mathbf{P}_\tau$ for term τ. Note that it must be computed for each term individually, but does not depend on the query. A formal description of the PEST matrix computation is given in Section 5.

Finally, the eigenvectors of the PEST matrix for each term τ are combined to form the vector space representation (i.e., the document-term matrix) for the wiki pages and their tags. Section 6 presents, as an example, the computation and the resulting term weights for the wiki from Figure 1.

Keyword *queries* can be evaluated on this representation with any of the existing IR engines using a vector space model (e.g., Lucene). Queries mixing

keywords and structure require an engine capable of combining keyword matches with structural constraints such as the KWQL engine.

Contributions

To summarize, PEST improves on existing fuzzy matching approaches for structured data (briefly summarized in Section 2) in the following aspects:

- It is based on a *simple, but flexible model for structured content* that captures a wide range of knowledge management systems and applications. We introduce the model in Section 4 and discuss how it can represent the core concepts of the semantic wiki KiWi, briefly recalled in Section 3.1. We also briefly recall KWQL (Section 3.2) to illustrate the need for a combination of structure and keyword queries.
- The main contribution of PEST is the PEST matrix for propagating term weights over structured data. The computation of that matrix for a given graph of structured content is formalized in Section 5.
  The PEST matrix allows the propagation of term weights at *index time* and yields a modified vector space representation that can be used by any IR engine based on the vector space model (e.g., Lucene).
  Section 6 gives an extended example of the PEST matrix computation on the sample wiki from Figure 1.
- We prove formally in Section 5.3 that any PEST matrix has 1 as dominant eigenvalue and that the power method converges with the corresponding eigenvector if applied to a PEST matrix.

Though the results from Section 6 as well as further internal testing validate the PEST approach, there are a number of open issues summarized in Section 7.

2 Related Work: Fuzzy Matching on Structured Data

PEST differs from the majority of fuzzy matching approaches including those reviewed in the following in two important ways:

- It is designed for *graph-shaped data* rather than purely hierarchical data as most of the XML-based approaches discussed in the following.
- In essence, PEST can be used with any information retrieval engine based on the vector space model. The only modification to the evaluation process is the computation of the actual vector space model. Otherwise existing technology (such as Lucene or similar search engines) can be utilized. In particular, the PEST matrix is query independent and thus can be computed at *index time*.

Before we consider specific approaches, it is worth recalling that *fuzzy matching*—approaches to include not only strict matches, but also other results which are relevant but do not match the strict interpretation of the query—and *ranking* are closely related. Though they do not have to be used in conjunction, this

is often the case, in particular to allow a fuzzy matching engine to differentiate looser results from results that adhere more strictly to the query.

While fuzzy matching is widely used in web search and other IR applications, conventional query languages for (semi-)structured data such as XQuery, SQL or SPARQL do not usually employ fuzzy matching or rank results. These languages have been applied to probabilistic data, but this is a distinct area from fuzzy querying: In probabilistic data management the data itself introduces uncertainty, in fuzzy matching uncertainty is introduced under the assumption that the user is also interested in matches that do not quite match her query.

As the amount of structured web data increases and the semantic web continues to emerge, the need for solutions that allow for layman querying of structured data arises. Research has been dedicated to combining web querying and web search and to introducing IR methods to querying, for example in the form of extensions to conventional query languages, visual tools for exploratory search, extension of web keyword search to include (some) structure and keyword search over structured data. With the arrival of these techniques, the need for fuzzy querying that does not apply solely to individual terms or phrases but takes the data structure into account arises.

Approximate matching on data structure has been researched mainly in the context of XML data similarity [15]. A wide body of work in this area can be divided into three main classes of approaches:

Tree edit distance: Tree edit distance approaches, e.g., [10, 1, 2] extend the edit distance in such a way that not strings but trees are compared. A number of types of edit operations may be applied repeatedly in order to transform one XML document into another. The similarity between the documents can then be quantified through a cost function taking into account the number of steps and types of operations required.

In contrast to PEST, these approaches are hard to generalize to graph data, require a relaxation loop at query time, and require the evaluation of a (often quite considerable) number of relaxed queries whereas PEST's computation can be performed entirely at index time. The last effect is slightly ameliorated by novel top-k algorithms in [2]. Also it is not obvious how different edge types, as easily treated by PEST, affect tree edit distance.

Approximate tree matching: A small number of approaches modify existing matching algorithms to introduce certain degrees of freedom. In [13], direct relations in the query are allowed to be matched with indirect relations in the document. In [14], a document is considered a good approximate match if it and the query have few paths that are not common (a mismatching).

Again, the contrast to PEST lies (a) in the limitation to tree-shaped data which would be hard to lift at least in the case of [14] due to the reliance on paths and suffix trees and (b) in the need for a new query engine, where PEST can reuse existing information retrieval engines.

Adapting the vector space model: Finally, the largest class of approaches aims, like PEST, to adapt the vector space model, a well-established IR technique, to the application on XML data. In the vector space model, documents and

queries are represented as vectors of weights for each term; similarity is computed as the cosine angle between two vectors.

Pokorny et al. [12] represent paths and terms in an XML tree in a matrix instead of a vector, assigning weights to each combination of path and term. A query, also expressed as an XML tree, is transformed into a matrix of the same form. The score of a query with respect to a possible result is then calculated as the correlation between the two matrices. In an extension, the matrix is adapted to reflect also the relationship between paths.

In [6] (and similarly [9]) document vectors are modified such that their elements are not weights for terms but rather weights for term and context pairs—the context of a term is the path in which it occurs. The vector then consists of a weight for each combination of term and context. Further, the cosine similarity measure is relaxed by computing context similarities which are integrated in the vector similarity measure.

Similarly, [13] and, later, [11] use tree embeddings combined with a vector space representation of XML elements.

Activation propagation is used in [3] for fuzzy matching over structure. Here, a modified version of the vector space model is used to calculate similarity scores between query terms and textual nodes in the data. The calculation of term weights takes into account the structural context of a term as well as its frequency. In a second step, these scores are propagated up in the tree. Finally, the highest activated nodes are selected, filtering out some results which are considered to be unsuitable such as the descendants of results that have already been selected. This approach resembles ours in that activation propagation is used to realize approximate matching over structure. However, in this approach, propagation happens upon query evaluation and is unidirectional. Like the other approaches in this class, it is also limited to tree-shaped data.

Outside of XML, one widely-used method where structural relationship is used for fuzzy matching is the use of anchor-tags in web search [4]. The anchor text of a link to a web page is treated as if it was part of the text of that web page even if it does not appear there. However, the application of this approach is limited to anchor tags and does not apply to general graphs or generalize to different link types.

3 Preliminaries

3.1 KiWi

KiWi[1] is a semantic wiki with extended functionality in the areas of information extraction, personalization, reasoning, and querying. KiWi relies on a simple, modular conceptual model consisting of the following building blocks:

Content Items are composable wiki pages, the primary unit of information in the KiWi wiki. A content item consists of text or multimedia and an optional sequence of *contained* content items. Thus, content item containment provides

[1] `http://www.kiwi-project.eu`, showcase at `http://showcase.kiwi-project.eu/`

a conventional structuring of documents, for example a chapter may consist of a sequence of sections. For reasons of simplicity, content item containment precludes any form of overlapping or of cycles, and thus a content item can be seen as a directed acyclic graph (of content items). **Links** are simple hypertext links and can be used for relating content items to each other or to external web sites.

Annotations are meta-data that can be attached to content items and links, describing their content or properties. They can be added by users, but can also be created by the system through automatic reasoning. Though KiWi supports various types of annotations ranging from informal, freely chosen tags, to semi-formal tags selected from a pre-defined vocabulary, to RDF triples and relationships from an ontology, we consider only tags consisting of phrases (one or several words) in this paper.

To illustrate these concepts, consider again Figure 1: It shows a sample KiWi wiki using the above structuring concepts (for sake of familiarity, we call content items documents). For example, the content item (document) 1 "About KiWi" contains the content item 2 representing a section on "Search in KiWi" and is linked to by the content item 6 "Guided Tour". It is tagged with 1.1 *"introduction"* and 1.2 *"architecture"*.

Structure, within as well as between resources, thus plays an important role for expressing knowledge in the wiki, ranging from simple tags to complex graphs of links or content item containment.

3.2 KWQL

KWQL [5], KiWi's label-keyword query language [16], allows for combined queries over full-text, annotations and content structure, fusing approaches from conventional query languages with information retrieval techniques for search.

KWQL aims to make data contained in a Semantic Wiki accessible to all users—not only those who have experience with query languages. Queries have little syntactic overhead and aim at being only as complex as necessary. The query language is designed to be close to the user experience, allowing queries over the elements of the conceptual model described in the previous section.

Further, KWQL has a flat learning curve and the complexity of queries increases with the complexity of the user's information need. Simple KWQL queries consist of a number of keywords and are no more complicated to write than search requests in web search engines. On the other hand, advanced KWQL queries can impose complex selection criteria and even reformat and aggregate the results into new wiki pages, giving rise to a simple form of reasoning.

Some examples of KWQL queries are given in the following table:

`Java`	Content items containing *"java"* directly or in any of its tags or other meta data
`ci(author:Mary)`	Content items authored by Mary (using `author` meta-data)
`ci(Java OR (tag(XML) AND author:Mary))`	

Content items that either contain *"java"* or have a tag containing *"XML"* and are authored by Mary

```
ci(tag(Java) link(target:ci(Lucene)))
```

Content items with a tag containing *"java"* that contain a link to a content item containing *"lucene"*

4 A Formal Model for Wiki Content: Content Graphs

In this section we formally define a generic graph-based model of structured content that is capable of capturing the rich knowledge representation features of KiWi.

Definition 1 (Content graph). *A* ***content graph*** *is a tuple* $G = (V_d, V_t, E_l, E_n, \mathcal{T}, w_t)$ *where* V_d *and* V_t *are sets of vertices and* $E_l, E_n \subseteq (V_d \cup V_t) \times (V_d \cup V_t)$. V_d *and* V_t *represent documents (content items) and tags.* E_l *and* E_n *describe the directed linking and nesting among documents and tags.*

The textual content of documents and tags is represented by a set $\mathcal{T}$ *of terms and a function* $w_t : (V_d \cup V_t) \times \mathcal{T} \to \mathbb{R}$ *that assigns a weight to each pair of a vertex and a term. We assume that the term weights for each vertex* v *are a stochastic vector (i.e.,* $\sum_{\tau \in \mathcal{T}} w_t(v, \tau) = 1$*).*

We denote the type of an edge e with $type(e) \in \{l, n\}$ and the type of a vertex v with $type(v) \in \{d, t\}$.

The above is an instance of a generic model, that allows for an arbitrary number of vertex and edge sets for flexible typing. Tags can be used to represent any property of a document other than its textual content. Here, we limit ourselves to two vertex and edge types each for sake of clarity. The model allows for different types of links and nestings exist depending on the types of linked and nested nodes. For example, an edge in $E_l \cap (V_d \times V_t)$ represents a link from a document to a tag, whereas an edge $E_l \cap (V_d \times V_d)$ represents a link between documents.

For the sample wiki from Figure 1, the six documents 1 to 6 form V_d, $V_t = \{1.1, 1.2, 2.1, 3.1, 3.2, 4.1\}$, $E_l = \{(6,1), (6,3), (4,3), (1,1.1), (1,1.2), \ldots, (4,4.1)\}$, $E_n = \{(1,2)\}$, $\mathcal{T}$ the set of all terms in the wiki and $w_t = \{(1, \text{"java"}, 0.8), \ldots, (2.1, \text{"search"}, 1), \ldots\}$.

Nesting of tags in documents, $E_n \cap (V_d \times V_t)$, do not occur in our model of a semantic wiki, but may do so in other applications.

5 Computing the PEST Propagation Matrix

Based on the above model for a knowledge management system, we now formally define the propagation of term-weights over structural relations represented in a content graph by means of an eigenvector computation.

A document's tag is descriptive of the content of the text of said content item—they have a close association. Similarly, the tags of a sub-document to

some extent describe the parent document since the document to which the tag applies is, after all, a constituent part of the parent document. More generally, containment and linking in a wiki or another set of documents indicate relationships between resources. We suggest to exploit these relationships for approximate matching over data structure by using them to propagate resource content. A resource thereby is extended by the terms contained in other resources it is related to. Then, standard information retrieval engines based on the vector space model can be applied to find and rank results oblivious to the underlying structure or term-weight propagation.

To propagate term weights along structural relations, we use a novel form of transition matrix, the PEST propagation matrix. In analogy to the *random surfer* of PageRank, the term-weight propagation can be explained in terms of a *semi-random reader* who is navigating through the content graph looking for documents relevant to his information need expressed by a specific term τ (or a bag of such terms). He has been given some—incomplete—information where in the graph τ occurs literally. He starts from one of the nodes and reads on, following connections to find other documents that are also relevant for his information need (even if they do not literally contain τ). When he becomes bored or loses confidence in finding more matches by traversing the structure of the wiki (or knowledge management system, in general), he jumps to another node that seems promising and continues the process.

To encode this intuition in the PEST matrix, we first consider which connections are likely to lead to further matches by weighting the edges occurring in a content graph. Let $\mathbf{H}$ be the transposed, normalized adjacency matrix of the resulting graph. Second, we discuss how to encode, in the leap matrix $\mathbf{L}_\tau$, the jump to a *promising* node for the given term τ (rather than to a random node as in PageRank)

The overall PEST matrix $\mathbf{P}_\tau$ is therefore computed as (where α is the leap factor)

$$\mathbf{P}_\tau = (1-\alpha)\mathbf{H} + \mathbf{L}_\tau.$$

Each entry $m_{i,j} \in \mathbf{P}_\tau$, that is, the probability of transitioning from vertex j to vertex i, is thus determined primarily by two factors, the normalized edge weights of any edge from j to i, the term weight of τ in j.

5.1 Weighted Propagation Graph

To be able to control the choices the semi-random reader performs when following edges in the content graph, we first extend the content graph with a number of additional edges and vertices and, second, assign weights to all edges in that graph.

Definition 2 (Weighted propagation graph). *A **weighted propagation graph** is a content graph extended with a function $w_e : (E_l \cup E_n) \to \mathbb{R}^2$ for assigning weights to edges that fulfills the following conditions:*

- *For each document $v \in V_d$, there is a tag $t_v \in V_t$ with $(v, t_v) \in E_l$.*

– *For each pair of documents $v, w \in V_d$ with $(u, v) \in E_l$ (E_n), if t_v and t_w are tags of v and w respectively, then there is an edge $(t_v, t_w) \in E_l$ (E_n).*

Edge weights are given as pairs of numbers, one for traversing the edge in its direction, one for traversing it against its direction.

The first condition requires that each document must be tagged by at least one tag. The second condition ensures that tags of related documents are not only related indirectly through the connection between the documents, but also stand in a direct semantic relation. For example, a document which contains another document about a certain topic trivially also is about that topic to some extent, since one of its constituent parts is.

Proposition 1. *For every content graph, a weighted propagation graph can be constructed by (1) adding an empty tag ("dummy tag") to each document that is not tagged at all and (2) copying any relation between two documents to its tags (if not already present).*

Consider again the sample wiki from Figure 1, the resulting weighted propagation graph is shown in Figure 2. It contains two "dummy tags" (5.1 and 6.1) as well as a number of added edges between tags of related documents.

We call a weighted propagation graph *type-weighted*, if for any two edges $e_1 = (v_1, w_1), e_2 = (v_2, w_2) \in E_l \cup E_n$ it holds that, if $type(e_1) = type(e_2)$, $type(v_1) = type(v_2)$, and $type(w_1) = type(w_2)$, then $w_e(e_1) = w_e(e_2)$. In other words, the weights of edges with the same type and with start and end vertices of the same type respectively must be the same in a type-weighted propagation graph. In the following, we only consider such graphs.

Let $\mathbf{A}_w$ be the weighted adjacency matrix of a weighted propagation graph G. Then we normalize and transpose $\mathbf{A}_w$ to obtain the transition matrix $\mathbf{H}$ for G as follows:

$$\mathbf{H} = \frac{1}{\max\left(\sum_i w_e((i,j))\right)} \mathbf{A}_w^T$$

Note that we normalize the columns for all vertices with the same maximum sum of outgoing term weights. This preserves differences in weights between nodes with the same number of outgoing edges, but also yields only a sub-stochastic matrix.

5.2 Informed Leap

Given a leap factor $\alpha \in (0, 1]$, a leap from vertex j occurs with a probability

$$P(\text{leap}|j) = \alpha + (1 - \alpha)(1 - \sum_i \mathbf{H}_{i,j})$$

A leap may be *random* or *informed*. In a random leap, the probability of jumping to some other vertex is uniformly distributed and calculated as $l^{\text{rnd}}(i, j) = \frac{1}{|V_d \cup V_t|}$ for each pair of vertices (i, j).

An informed leap by contrast takes the term weights, that is, the prior distribution of terms in the content graph into account. It is therefore term-dependent and given as $l_\tau^{\inf}(i,j) = \frac{w_t(i,\tau)}{\sum_k w_t(k,\tau)}$ for a $\tau \in \mathcal{T}$.

In preliminary experiments, a combination of random and informed leap, with heavy bias towards an informed leap, proved to give the most desirable propagation behavior. The overall leap probability is therefore distributed between that for a random leap and that of an informed leap occurring according to the factor $\rho \in (0,1]$ which indicates which fraction of leaps are random leaps.

Therefore, we obtain the leap matrix $\mathbf{L}_\tau$ for term τ as

$$\mathbf{L}_\tau = \Big(P(\text{leap}|j) \cdot \big((1-\rho) \cdot l^{\inf_\tau}(i,j) + \rho \cdot l^{\text{rnd}}(i,j)\big) \Big)_{i,j}$$

5.3 Properties of the PEST Matrix

Definition 3 (PEST matrix). *Let $\alpha \in (0,1]$ be a leap factor, $\mathbf{H}$ be the normalized transition matrix of a given content graph (as defined in Section 5.1) and $\mathbf{L}_\tau$ the leap matrix (as defined in Section 5.2) to $\mathbf{H}$ and term τ with random leap factor $\rho \in (0,1]$. Then the* PEST *matrix $\mathbf{P}_\tau$ is the matrix*

$$\mathbf{P}_\tau = (1-\alpha)\mathbf{H} + \mathbf{L}_\tau.$$

Theorem 1. *The* PEST *matrix $\mathbf{P}_\tau$ for any content graph and term τ is column-stochastic and strictly positive (all entries > 0).*

Proof. It is easy to see that $\mathbf{P}_\tau$ is strictly positive as both α and ρ are > 0 and thus there is a non-zero random leap probability from each vertex to each other vertex.

$\mathbf{P}_\tau$ is column stochastic, as for each column j

$$\begin{aligned}
\sum_i (\mathbf{P}_\tau)_{i,j} &= \sum_i \big((1-\alpha)\mathbf{H}_{i,j} + (\mathbf{L}_\tau)_{i,j}\big) \\
&= (1-\alpha)\sum_i \mathbf{H}_{i,j} + \Bigg(\big(\alpha + (1-\alpha)(1 - \sum_l \mathbf{H}_{l,j})\big) \cdot \\
&\qquad \big((1-\rho) \cdot \underbrace{\sum_i l_\tau^{\inf}(i,j)}_{=1} + \rho \underbrace{\sum_i \cdot l^{\text{rnd}}(i,j)}_{=1}\big) \Bigg) \\
&= (1-\alpha)\sum_i \mathbf{H}_{i,j} + (1-\alpha)(1 - \sum_l \mathbf{H}_{l,j}) + \alpha = 1 - \alpha + \alpha = 1
\end{aligned}$$

Corollary 1. *The* PEST *matrix $\mathbf{P}_\tau$ has eigenvalue 1 with unique eigenvector $\mathbf{p}_\tau$ for each term τ.*

The resulting eigenvector $\mathbf{p}_\tau$ gives the new term-weights for τ in the vertices of the content graph after term-weight propagation. It can be computed, e.g., using the power method (which is guaranteed to converge due to Theorem 1).

	1	2	1.1	1.2	2.1	4	3	4.1	3.1	3.2
1	0.1463	0.4091	0.4848	0.4848	0.1054	0.2556	0.1873	0.1873	0.2146	0.2146
2	0.1630	0.0109	0.0088	0.0088	0.3165	0.0130	0.0095	0.0095	0.0109	0.0109
1.1	0.2019	0.0109	0.0088	0.0088	0.1998	0.0130	0.0095	0.0095	0.0109	0.0109
1.2	0.2019	0.0109	0.0088	0.0088	0.1998	0.0130	0.0095	0.0095	0.0109	0.0109
2.1	0.0074	0.2054	0.1644	0.1644	0.0054	0.0130	0.0095	0.0095	0.0109	0.0109
4	0.1116	0.1637	0.1324	0.1324	0.0804	0.1949	0.1817	0.4540	0.1637	0.1637
3	0.0074	0.0109	0.0088	0.0088	0.0054	0.0908	0.0095	0.0095	0.3220	0.3220
4.1	0.0074	0.0109	0.0088	0.0088	0.0054	0.2074	0.0095	0.0095	0.0498	0.0498
3.1	0.0074	0.0109	0.0088	0.0088	0.0054	0.0130	0.2040	0.0873	0.0109	0.0109
3.2	0.0074	0.0109	0.0088	0.0088	0.0054	0.0130	0.2040	0.0873	0.0109	0.0109

Table 1. Excerpt of PEST matrix for *"java"* with $\alpha = 0.3$ and $\rho = 0.25$

The vector space representation of the content graph *after term-weight propagation* is the document-term matrix using the propagation vectors $\mathbf{p}_\tau$ for each term τ as columns.

6 Structure Propagation with PEST Matrix: An Example

In order to confirm that the described propagation approach performs as expected, a prototype implementation of the PEST matrix construction has been implemented and experiments computing the resulting vector space representation after term-weight propagation have been conducted. The implementation is available from `http://www.pms.ifi.lmu.de/pest`.

Here, we present the results for the sample wiki from Figure 1. We use a leap factor of $\alpha = 0.3$ and a random leap factor of $\rho = 0.25$. Using these factors, the PEST matrix is computed for each term $\tau \in \{\text{"java"}, \text{"lucene"}, \ldots\}$. The edge weights are derived by intuition of the authors as shown in Figure 2.

The resulting matrix for the term *"java"* is shown in Table 1, omitting Documents 5 and 6 and their tags for space reasons.

Note that the matrix contains high probabilities for propagation to 1 and 4 throughout thanks to the informed leap. This preserves their higher term-weight for *"java"* compared to other nodes that do not contain *"java"*.

Using the PEST matrix, we compute for each term the resulting PEST vector $\mathbf{p}_\tau$. Together these vectors form a new document-term matrix representing the documents and tags in our wiki, but now with propagated term weights, as shown in Table 2.

To verify the veracity of our approach, let us consider a number of desirable properties an approach to fuzzy matching on a structured knowledge management systems such as KiWi should exhibit:

1. Documents containing a term directly (e.g., *"java"*) with a significant term weight should still be ranked highly after propagation. This should hold to

	RDF	*XML*	*architecture*	*introduction*	*java*	*lucene*	*search*
1	0.46	0.03	0.11	0.11	0.26	0.08	0.07
1.1	0.11	0.02	0.05	0.23	0.07	0.04	0.07
1.2	0.11	0.02	0.24	0.04	0.07	0.04	0.07
2	0.10	0.02	0.05	0.05	0.06	0.21	0.09
2.1	0.06	0.02	0.06	0.06	0.04	0.06	0.24
3	0.02	0.08	0.09	0.04	0.04	0.22	0.09
3.1	0.02	0.04	0.03	0.04	0.02	0.06	0.22
3.2	0.02	0.04	0.23	0.04	0.02	0.06	0.03
4	0.01	0.53	0.02	0.08	0.17	0.02	0.02
4.1	0.01	0.12	0.02	0.22	0.04	0.02	0.02
5	0.01	0.02	0.01	0.03	0.11	0.11	0.01
5.1	0.01	0.01	0.01	0.02	0.03	0.03	0.01
6	0.03	0.02	0.03	0.02	0.03	0.03	0.02
6.1	0.03	0.02	0.04	0.03	0.02	0.02	0.03

Table 2. Document-term matrix after term-weight computation

guarantee that direct search results (that would have been returned without fuzzy matching) are retained.

Indeed Documents 1, 4, and 5, all containing *"java"* are highest ranked for that term, though the tags of Document 1 come fairly close. This is desired, as Document 1 contains *"java"* with high term weight and tag-document associations are among the closest relations.

2. A search for a term τ should also yield documents not containing τ but directly connected to ones containing it. Their rank should depend on the weight of τ in the connected document and the type (and thus propagation strength) of the connection.

   Again, just looking at the results for *"java"* the two tags of Document 1 as well as the contained Document 2 receive considerable weight for term *"java"*.

3. Searching for a KWQL query such as `ci(architecture introduction)` should also rank highly documents that do not include these terms, but that are tagged with *"architecture"* and *"introduction"*.

   Document 1 is such a case and is indeed the next highest ranked document for such a query after the three documents directly containing *"architecture"* or *"introduction"* (using either boolean or cosine similarity).

Though this evaluation can, by design, only illustrate the effectiveness of the proposed term-weight propagation approach for fuzzy matching, we believe that it is a strong indication that it will prove efficient and effective also for larger and more diverse document collections.

7 Conclusion and Open Questions

PEST is a unique approach to fuzzy matching that combines the principles of structural relevance from approaches such as PageRank with the standard vector space model. Its particular strength is that it runs entirely at index time and results in a modified vector space representation.

However, the present paper is just the first step in exploring the potential and research issues on term-weight propagation as eigenvector computation over structured data.

First, and most obvious, extensive *experimental evaluation* of the approach including a comparison with existing methods is called for. In particular, we are currently estimating the values for α and ρ as well as for the edge weights "by the seat of our pants" rather than empirical observation. A guide to choosing these values might be possible to derive from studying the behavior of PEST on various kinds of data. Edge values, in particular, could also be amenable to various machine learning approaches, using, for example, average semantic relatedness as a criterion, or to semi-automatic approaches through user-feedback.

We have also considered a number of *different algorithmic approaches to term-weight propagation*, e.g., where propagation is not based on convergence but on a fixed number of propagation steps. Techniques for spreading activation [7, 8] might be applicable and a comparison study is called for. Furthermore, the computation of the PEST matrix is just one of several alternatives for finding a stochastic propagation matrix.

There are also a number of *specific areas for improving* PEST:

1. In PEST, propagation between documents and between tags and documents influence each other: E.g., a document with many tags will propagate only a relatively smaller amount to its children than a document with few children. For extreme cases, a model where each of these kinds of propagations is at least each given a minimal amount might prove superior to the basic version of PEST described here.

2. The model in this paper does not address the representation of tagged links. One simple way to do this would be to represent a tagged link between two documents as a direct link and in addition a tag that is connected via links to both documents. Alternatively, *typed links* could be introduced. They create the possibility of dynamically determining the weight of a connection based on the link type and term being propagated, for example depending on their semantic similarity as determined through their Google distance or distance in an *ontology*.

3. Links to *external resources* such as Linked Open Data or ontologies are currently not considered in PEST. Their inclusion would allow to enrich the content graph and thereby enhance the results of term propagation. This extension seems particularly promising in combination with aforementioned typed links.

4. Another, wiki-specific, extension is observing how the term scores of a document change over several *revisions* and taking this into account as a factor when ranking query answers.

5. Any fuzzy matching approach suffers from non-obvious *explanations* for returned answers: In the case of a boolean query semantics, the answer is obvious, but when term propagation is used, a document might be a highly-ranked query result without as much as containing any query terms directly. In this case, providing an explanation, for example that the document in question is closely connected to many documents containing query terms, makes the matching process more transparent to users. However, automatically computing good, minimal explanations is far from a solved issue.

Acknowledgments

The research leading to these results is part of the project *"KiWi—Knowledge in a Wiki"* and has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 211932.

References

1. S. Amer-Yahia, S. Cho, and D. Srivastava. Tree pattern relaxation. In *EDBT*, 2002.
2. S. Amer-Yahia, L. V. S. Lakshmanan, and S. Pandit. FleXPath: flexible structure and full-text querying for XML. In *SIGMOD*, 2004.
3. V. N. Anh and A. Moffat. Compression and an IR approach to XML retrieval. In *INEX Workshop*, pages 99–104, 2002.
4. S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *WWW*, 1998.
5. F. Bry and K. A. Weiand. Flavors of KWQL, a keyword query language for a semantic wiki. In *SOFSEM*, 2010.
6. D. Carmel, Y. Maarek, Y. Mass, N. Efraty, and G. Landau. An extension of the vector space model for querying XML documents via XML fragments. In *SIGIR Workshop on XML and Information Retrieval*, pages 14–25, 2002.
7. A. Collins and E. Loftus. A spreading-activation theory of semantic processing. *Psychological review*, 82(6):407–428, 1975.
8. F. Crestani. Application of spreading activation techniques in information retrieval. *Artificial Intelligence Review*, 11(6):453–482, 1997.
9. T. Grabs and H.-J. Schek. Flexible information retrieval on XML documents. In *Intelligent Search on XML Data*, 2003.
10. S. Guha, H. V. Jagadish, N. Koudas, D. Srivastava, and T. Yu. Approximate xml joins. In *SIGMOD*, 2002.
11. V. Kakade and P. Raghavan. Encoding xml in vector spaces. In *ECIR*, 2005.
12. J. Pokorný. Vector-oriented retrieval in XML data collections. In *DATESO*, 2008.
13. T. Schlieder and H. Meuss. Querying and ranking xml documents. *J. Am. Soc. Inf. Sci. Technol.*, 53(6):489–503, 2002.
14. D. Shasha, J. T.-L. Wang, H. Shan, and K. Zhang. Atreegrep: Approximate searching in unordered trees. In *SSDBM*, 2002.
15. J. Tekli, R. Chbeir, and K. Yetongnon. An overview on XML similarity: Background, current trends and future directions. *Computer Science Review*, 3(3):151 – 173, 2009.
16. K. Weiand, T. Furche, and F. Bry. Quo vadis, web queries? In *Int'l. Workshop on Semantic Web Technologies (Web4Web)*, 2008.

Jump-starting a Body-of-Knowledge with a Semantic Wiki on a Discipline Ontology

Víctor Codocedo, Claudia López, and Hernán Astudillo

Universidad Técnica Federico Santa María,
Avenida España 1680, Valparaíso. Chile
{vcodocedo,clopez,hernan}@inf.utfsm.cl
http://www.usm.cl/

Abstract. Several communities have engaged recently in assembling a Body of Knowledge (BOK) to organize the discipline knowledge for learning and sharing. BOK ideally represents the domain, contextualizes assets (e.g. literature), and exploits the Social Web potential to maintain and improve it. Semantic wikis are excellent tools to handle domain (ontological) representations, to relate items, and to enable collaboration. Unfortunately, creating a whole BOK (structure, content and relations) from scratch may fall prey to the "white page syndrome"[1], given the size and complexity of the domain information. This article presents an approach to jump-start a BOK, by implementing it as a semantic wiki organized around a domain ontology. Domain representation (structure and content) are initialized by automatically creating wiki pages for each ontology concept and digital asset; the ontology itself is semi-automatically built using natural language processing (NLP) techniques. Contextualization is initialized by automatically linking concept- and asset-pages. The proposal's feasibility is shown with a prototype for a Software Architecture BOK, built from 1,000 articles indexed by a well-known scientific digital library and completed by volunteers. The proposed approach separates the issues of domain representation, resources contextualization, and social elaboration, allowing communities to try on alternate solutions for each issue.

Key words: semantic wiki, body of knowledge, automated domain ontology, digital assets contextualization

1 Introduction

In recent years, several professional and academic communities have undertaken to organize and systematize their knowledge with a "Body Of Knowledge" (BOK for short). BOK's have been created most famously for project management

[1] Colloquial name for writers' mental block when starting a new piece from scratch

(PMBOK [2] by the PMI [3]) and for software engineering (SWEBOK [4] [5]), but also for IT architecture (ITABOK [6] by IASA [7] [8] [9]), and other related disciplines.

Body-of-Knowledge (BOK) requirements typically include representing the domain, contextualizing resources (e.g. literature), and relying on Social Web members to maintain and improve it. Semantic wikis are excellent tools to handle domain (ontological) representations, to relate items, and to enable collaboration. Unfortunately, creating a whole BOK (structure, content and relations) from scratch may easily lead to the "white page syndrome", given the size and complexity of the domain information.

This article presents an approach that differs from most current BOK's in exploiting a formal discipline description to maintain the knowledge organization. It also presents several tools to automate the creation of a domain conceptualization (in concepts of a populated ontology), a semantic wiki to manage the domain representation and its assets, stylized wiki elements, and a timeline-based browser to explore the domain.

The reminder of the article is structured as follows: section 2 summarizes earlier related work; section 3 introduces the proposed approach for building a BOK; section 4 explains how the wiki structure, content and linking are initialized; section 5 describes the ConcepTion tools that implement the proposal; section 6 suggests some future work; 7 summarizes and concludes.

2 Related Work

Several strands of work are directly related to this approach.

2.1 Semantic Wiki

Semantic Wikis are designed to allow collaborative creation of content using a fixed syntax and semantics to improve searching and querying. In traditional wikis it is possible to find basic *building blocks* to create content (on most wikis only a set of pages each one with a set of links). Semantic wikis provides an expanded set of *building blocks* such as relations, entity types and RDF or OWL annotations [4].

[2] PMBOK - Project Management Body Of Knowledge: `www.pmi.org/Resources/Pages/Library-of-PMI-Global-Standards.aspx`
[3] PMI - Project Management Institute: `www.pmi.org/`
[4] SWEBOK - Software Engineering Body of Knowledge: `www.computer.org/portal/web/swebok`
[5] ACM - Association for Computing Machinery: `www.acm.org/`
[6] ITABOK - IT Architect Body of Knowledge: `www.iasahome.org/web/home/skillset`
[7] IASA - International Association of Software Architects: `www.iasahome.org/`
[8] EABOK - Enterprise Architecture Body Of Knowledge: `www.mitre.org/work/tech_papers/tech_papers_04/04_0104/index.html`
[9] CBK - Common Body Of Knowledge: `www.cissp.com/`

Semantic Media Wiki [11] is a semantic wiki implementation that supports semantic templates creation, allowing to create fixed representations for each concept of the BOK. Semantic Media Wiki is an extension of the popular Media Wiki project[10], the platform on which Wikipedia works on. By this reason it provides a large set of useful extensions like SIMILE Timeline [11], an interactive Timeline browser.

The Kiwi wiki [19] (a EU-funded project) is another semantic wiki implementation that provides some advanced semantic annotation features, allowing a deeper granularity of the information (this feature was inherited from its predecessor IkeWiki [18]). It also provides what they call "Content Versatility", which are different views over the same content implemented by different applications. Unfortunately, Kiwi does not provides as many extensions as Semantic Media Wiki does. By using Kiwi, we think that we will lose some time on building them.

2.2 Semantic Digital Libraries and Ontology-based Approaches

Angelo di Iorio et al. [9] proposed WikiFactory to automatically create a domain semantic wiki from a domain ontology. Their work is based on customizing a semantic wiki from an ontology definition to add the content afterwards.

Jerome DL [13] is a semantic digital library whose main requirements are: provide user-oriented browsing features and allow efficient searching using semantic tools. The description of resources is based on Dublin Core[12]and FOAF[13]. Unfortunately, this two ontologies are quite simple on their specification. In that way, documents cannot be contextualized to a domain specific categorization for searching purposes.

ScholOnto [20] is a discourse ontology for describing Digital Libraries designed to support searching, tracking and analyzing concepts from academic perspectives. It is focused on expressing the *claims* that authors make on their documents. Although this is an interesting perspective we realize that such an approach leads to the "white page syndrome" as authors lack on time and motivation to fill templates with this information.

2.3 Bodies of Knowledge

There is not a single, common structure for all BOK's:

- The SWEBOK [22] is organized into ten knowledge areas (KAs): requirements, design, construction, testing, maintenance, configuration management, engineering management, engineering process, engineering tools and

[10] http://www.mediawiki.org
[11] SIMILE: www.simile-widgets.org/timeline/
[12] Dublin Core: `www.dublincore.org/`
[13] FOAF - Friend of a Friend Project: `www.foaf-project.org/`

methods, and quality. The SWEBOK contents were authored under the guidance, coordination and editing of a committee, originally composed of members of several professional societies; and benefited from systematic revision by hundreds of individuals.
- The PMBOK [17] identifies 44 processes, organized into five process groups and nine knowledge areas; the process groups are: Initiating, Planning, Executing, Controlling and Monitoring, and Closing; and the knowledge areas are: Project Integration Management, Project Scope Management, Project Time Management, Project Cost Management, Project Quality Management, Project Human Resource Management, Project Communications Management, Project Risk Management, and Project Procurement Management.
- The ITABOK [14], also called The Aspiring Architect Skills Library, is organized around a taxonomy of IT architect skills, proposed by IASA as well; the taxonomy categories are: Bussiness Technology Strategy, Design, Human Dynamics, Infrastructure, IT Environment, Quality Attributes, and Software. The ITABOK holds several articles in each category; topics were defined by a Training Committee, and bid on by practitioners.

Clearly, there are alternative notions of what a BOK is and how it should be written. But some generalizations can be made:

- A BOK is not just another textbook (an authoritative view by an individual or a committee); if so, it runs the risk of quickly becoming (or being born already) obsolete.
- A BOK can be created from resource collections, but it is more than their sum; otherwise, an overall "big picture" does not emerge.

Although digital assets (e.g. papers, learning objects, Web sites...) are important, a BOK cannot be just a search engine for assets.

3 Proposal

Building a body of knowledge (BOK) is expensive in human resources and time: it demands not only defining concepts and relations among them, but also requires a management system capable of support a whole community that will collaborate to create knowledge and enable inexperienced members of the community to understand the domain. To simplify and speed-up these requirements, we propose an ontology-based BOK which is semi-automatically populated from authoritative documents (such as articles). The BOK is enriched socially using the wiki, and is presented on a timeline to help better understand topics evolution in the community.

[14] `www.iasahome.org/web/home/skillset`

3.1 Ontology-based Body of Knowledge

There is a link between ontologies and BOK's: an ontology is a knowledge representation in which concepts are organized in hierarchies and are related to each other through relations, and a BOK is also a knowledge organization in which a discipline is presented through definitions of concepts. (REFERENCIA A MAX VOLKEL). Both ontologies and BOK's are knowledge organizations, their difference being for whom they are constructed: ontologies are intended to be machine-readable whereas BOK's are intended to be used and understood by humans. It is not only a format difference that arises here (structured information v/s free text).

Our approach tries to balance the trade-off between representation accuracy and usability of the organization [1] by maintaining a simple ontology that represents the Software Architecture discipline. Thus, we benefit from the good representation given by ontologies and the "good" user experience provided by BOKs. The ontology is created from authoritative documents, and the BOK presented to the user is based on a software architecture thesaurus and the manual organization provided by Software Architects.

3.2 An Ontology for Software Architecture from the Literature

From a very simplistic point of view, the more papers of a given domain a researcher is able to read, the more understanding he will have of what is happening with that domain. It should be possible to aid this process by automating the analysis of publications, using basic *Information Extraction* [6] techniques and *Concept frequency analysis*. Although clearly the process of understanding a discipline is not yet automatable, current technologies allow to jump-start the creation of a knowledge model such as an ontology. For this work we used and extended SKOS ontology [15] to model the Concepts of a domain. We added a new Class called *DigitalAsset* that represents a digital artifact that contains *explicit knowledge* about a Concept (REFERENCIA A VOLKEL DE NUEVO). The simplicity of the ontology we chose owes much to the design criteria for *Minimal Ontological Commitment* [8].

The publication full body is not used for analysis since it would require a much more complex and expensive process for extracting information. Instead, we analyze publications' metadata since simple, structured and also freely available on Internet from Web sites such as DBLP[16], CiteSeer[17] or ScienceDirect[18].

[15] SKOS - Simple Knowledge Organization System: `www.w3.org/2004/02/skos/`
[16] `www.dblp.org`
[17] `www.citeseer.org`
[18] `www.sciencedirect.org`

Table 1. Papers per Concept

no.	Concept	Digital Assets Set	Frequency
1	Architecture Rationale	p1,p2,p3,p4,p5,p6,p7	7
2	Reusability	p0,p2,p4,p5,p6	4

Mining digital assets metadata to extract Concepts The following excerpt is a typical Bibtex[19]entry provided by ScienceDirect[20].

```
@article{Kazman2005511,
title = "From requirements negotiation to software architecture decisions",
year = "2005",
...
author = "Rick Kazman and Hoh Peter In and Hong-Mei Chen",
keywords = "Requirements negotiation", "Architecture analysis",...
abstract = "Architecture design and requirements..."}
```

Three main fields may contain information of the Software Architecture discipline: keywords, title and abstract. We use keywords as a primary data source, since it is the simplest information available (tags of no more than 3 words). The analysis is based on two properties of the keywords:

- Keyword Frequency: If a keyword is present on several papers (that is, a keyword was used to tag several papers) that keyword represent an important Concept for the discipline that is being analyzed.
- Co-occurrence: If a subset of keywords is present on several papers, all the keywords in the subset are likely to be related to each other.

We extended the analysis to the Abstract field, which contains a short text comprising the main ideas of the content of the document. This text was used as a search-base for the Keywords (processed with Named Entity Recognition [21]).

This analysis yields a thesaurus with Concepts related to each other but with no hierarchy among them.

Creating a hierarchy of Concepts Given two Concepts related by co-occurrence analysis, we would like to know which Concept is broader and which one is narrower in the discipline, to add semantics to their relation. We proposed to identify and compare all digital assets associated to the Concepts. Table 1 shows two Concepts, each with an associated collection of digital assets.

[19] Bibtex is a tool and file format to describe and process references - see `www.bibtex.org`

[20] ScienceDirect: `www.sciencedirect.com`

[21] Named Entity Recognition is an Information Extraction technique used to identify entities on texts

Both Concepts co-occur on 4 different digital assets so we could say that they are related by co-occurrence. However, an 80% of the digital assets of the Concept #2 are contained on the set of concept #1, and only a 57% of the digital assets of concept #1 are in the concept #2 set (we call these percentages *co-ocurrence factors*). We can make the simple assumption that 80% of the literature of the concept *Reusability* is part of the literature of the concept *Architecture Rationale* and thus, *Reusability* represents something in the subdomain of *Architecture Rationale*. Since we cannot know what is this "something" that it represents we use a shallow relation stating only that *Reusability* is a narrower concept than *Architecture Rationale* (actually, *Reusability* of design rationale documents is a major goal of *Architecture Rationale*).

Applying this technique to every pair of co-occurrent concepts yields a hierarchy that emerges from the flat thesaurus built by mining the digital assets metadata. We can choose the minimal *co-ocurrence factor* to create the "narrower" relation between two concepts. We call this the *co-ocurrence filter*. Notice that a concept is not constrained to be *narrower* of only one concept (*Reusability* also is narrower than *Non-functional requirement*).

Enriching Keywords with a thesaurus The ontology built is used as a backbone of the BOK. That means that it should be as complete as possible to cover all the main aspects of the discipline on research. Nevertheless, using only the keywords provided by the authors of papers yields some drawbacks:

- Ambiguous Concepts: Authors often get too creative to tag their documents. Ambiguity is a main problem of tagging as author s will tag using their own knowledge (different from shared knowledge) (*architecture design, architectural design*).
- Too Generic Concepts: Some Concepts are too generic for the discipline and may not appear in the collection of Keywords since they do not represent a good tag for categorization. For instance, the word *System* is never used as a Keyword to tag a Software Architecture paper.
- Too Specific Concepts: Many Keywords are too specific and do not add useful information that can be used on the BOK. For example, proper names, identificators, etc. These kind of Keywords add noise to the final ontology.

To overcome these issues, the initial dictionary of concepts to search on abstracts is created over a thesaurus (we use a Software Architecture thesaurus presented by Fraga et al.[7]). The thesaurus plays a triple role in the process:

- Using tools such as lemmatization, we can anchor different tags to a single concept within the thesaurus ($\{$*architecture design, architectural design*$\} \Rightarrow \{$*Software Architecture Design*$\}$) reducing ambiguity.
- It adds words that, for being too generic, will not appear as Keywords on papers (*System* is a main concept in the thesaurus).

Too Specific Concepts need to be managed on a different way. We cannot just simply ignore all Keywords from papers' metadata and use only those on the

hand-made thesaurus because we would lose the capacity to discover information or new trends and topics. Specific concepts that cause noise are avoided by filtering them by the frequency they have. The idea is simple, the more specific a concept is, the less frequency it will have. Only concepts that appear in more than X papers will be used. We called X the *frequency filter*.

4 Use of Semantic Wiki for a BOK

The configuration of a wiki for the identified metamodel implies creating two kinds of pages: those representing domain concepts, and those representing digital assets. Both kinds of pages make use of specialized Infoboxes, allowing a standardized visual representation of the (concept or asset) attributes. The relationship between assets and concepts is represented by inter-pages referencing.

4.1 Discipline Exploration: Page per Concept

The ontology is later used on a semantic wiki, where a single wiki page is created for each concept (a little program in java was used to do such labor). At this point is necessary to understand that the we are providing a jump-start approach for the SABOK, but of course, the definitions and contents of this knowledge representation remains in the hands of the Software Architecture Community. Of course, some information is provided on the semantic wiki for each concept: Broader Concepts, Narrower Concepts, Associated Digital Assets, and Topic Category. According to the properties of the concept, we have created specific types of topics.

The semantic wiki allows the community to create and maintain content collaborative, populating and enriching the SABOK; explaining such tools is out of the scope of this work. Searching concepts can be done either with the free-text searching tool provided by the wiki framework, or by browsing the thesaurus used to build the ontology.

4.2 Resource Contextualization: Page per Asset

Since each concept on the SABOK has several digital assets associated (the same used to build the ontology), it can be used as a digital asset search tool as well. The ontology behind the SABOK allow us to use inference on answering queries. We have identified two inference levels: basic, and based on concepts.

Basic transitivity. Since the concepts are arranged on a hierarchy we can provide transitivity inference level for digital assets associated on a branch of concepts. For instance, all digital assets associated to *Reusability* will be answered to the query *"digital assets for Architecture Rationale"*.

As it can be seen, the SABOK besides from organizing the discipline knowledge, provides a searching capability of Digital Assets associated to each concept based on inference powered by its ontology.

4.3 Subject-based Exploration with Timelines

The generated BOK can be browsed with a timeline-based tool, which shows the evolution of concepts and how they relate to each other. A timeline-based visualization tool can show which concepts concite attention currently. Crosscuting concepts can be visually identified because they have a constant participation in the timeline over the years. Users can access the community-created information of concepts and the wiki itself to edit and manage it.

The information that tool requires resides as *year of publication* in the digital assets information (see section 3.2). In the timeline, the concepts are presented with the dates of the first and (currently) last publication that use it.

The timeline can also be used to present Digital Assets evolution around a concept. This should be really useful for researchers looking for the last publications according certain subject, for example.

Finally, new Digital Assets can be added to the SABOK, such as lessons, presentations, posters, video, etc.

5 Case Study: A Software Architecture BOK

The proposed approach has been implemented in a system named *ConcepTion* [22], composed of three main tools: a Miner, a Hierarchizer, and a Visualizer.

The approach was validated with a case study for the Software Architecture domain.

5.1 Software Architecture(s) Descriptions

Several efforts have been carried out to build a vocabulary for Software Architecture (SA). However, most of them are not intended to describe the entire Software Architecture discipline but systems and parts thereof (i.e. the discipline subject matter, not the discipline itself).

The SA community has recently focused on describing and recording architecture knowledge (AK) that supports the architecting process (e.g. adopted and discarded decisions, rationale, tradeoffs), and several metamodels and ontologies has been proposed to systematize it (PAKME [2], ADDSS [5], Archium [10], AREL [21], NDR [16], [12], among others). Also, Liang et al. [15] tackled the measuring of semantic distance among several proposals to describe AK, and defined a set of characteristics to categorize all AK concepts.

Unfortunately, only a couple of articles have proposed a broader description of the entire software architecture discipline. Babu et al. [14] introduced ArchVoc, the most cited software architecture ontology, which was generated with combined manual and semi-automatic techniques to identify software architecture concepts. The manual technique used the back-of-the-book index of major software architecture books, and the semi-automatic technique parsed

[22] `www.toeska.cl/conception/wiki/`

architecture-related Wikipedia[23]pages. The first approach yield 480 concepts, and the second one, 1650 concepts; they were organized into 9 overall categories, which were also sorted according to architecting phases.

Fraga et al. [7] also employed both an automatic and a manual technique to generate a software architecture thesaurus. The corpus of both generation techniques were the back-of-the-book index of major software architecture books (in 2005). The manual process yield a 500-concept thesaurus, and the automatic technique generated a 1200-concept thesaurus. Both thesauri were combined yielding 27 top-level concepts.

Although these two thesauri are good vocabularies to classify existing SA knowledge, there are several challenges that have not been already tackled in building a software architecture discipline vocabulary:

- Both thesauri have been manually manipulated to better classify SA knowledge, so their hierarchies and relationships are usually very influenced by existing conceptual frameworks present in the discipline. This aspect certainly helps to create good thesauri for information search, but it usually hampers its ability to describe real connections among concepts. For example, they group "fault-tolerance", "performance" and "usability" into a single category ("Quality Requirements" or "Non-Functional Requirements"), but in practice all three concepts are rarely present in the same article; indeed, most papers (and communities) focus on only one of them. Also, "fault-tolerance" is more frequently related to "validation" and "formal methods" than to any other quality requirement.
- The starting corpus of these thesauri did not include published research or industry articles, either; they used back-of-the-book indices and/or SA-related Wikipedia pages. This corpus selection reduces the vocabulary scope to those topics already published in books, omitting new trending topics or novel techniques that might be being discussed in major refereed SA conferences or journals. For example, none of these thesauri mention "design rationale" or "software architecture rationale", both dealt with in several recent mainstream articles.

5.2 Mining

The ontology was populated using 1,000 Bibtex files (including abstract) returned by ScienceDirect [24] for the "Software Architecture" search concept. Extracted metadata was stored in RDF [25]. Table 2 shows some statistics generated by the Miner.

Over 10% of the articles do not have an abstract in their Bibtex file, so we can only rely on the keywords that the authors used to tag them. Interestingly, only

[23] Wikipedia: `www.wikipedia.org`

[24] ScienceDirect: www.sciencedirect.com

[25] RDF: Resource Description Framework, the industry standard to store Semantic Information; see `www.w3.org/RDF/`.

Table 2. Statistics from ConcepTion Miner

Name	Value
Quantity of Papers	1000
Quantity of Papers with Abstract	886
Quantity of unique Concepts	2203
Concepts over 50%	47

47 tags account for more than the 50% of the matches produced by comparing searching dictionary concepts in abstracts. These are the most important and which we focused on.

5.3 Hierarchizer

The Hierarchizer compares every pair of concepts and calculates a co-occurrence factor between them, (see section 3.2). We can lower the *co-ocurrence filter* to find more relations among concepts, but of course, the lower it is, the more false positives we will find. We have found empirically that a *co-ocurrence filter* of 80% is appropriate to discover new relations and maintain false positives on a low level.

The *co-occurrence filter* and *frequency filter* (see section 3.2) are the two parameters that can be used to adjust the quality of the hierarchy obtained, and thus, the ontology's instances.

After creating the hierarchy, it can be visualizated with Graphviz[26]to draw the concepts and their relations, allowing Software Architecture experts to audit it and manually filter false-positives. Some samples of hierarchies can be found on Toeska's Website[27].

5.4 The prototype SABOK

The prototype SABOK was implemented using the semantic wiki platform Semantic Media Wiki [28] (SMW). A simple ad-hoc tool adds a wiki page for each concept in the hierarchy.

A timeline browser was also built with the MIT SIMILE Timeline[29]allowing to use HTML and JavaScript to use XML data, namely, a Knowledge Base with the ontology created.

Figure 1 shows a screenshot of the prototype SABOK. The evolution of the Concept *Architecture* is shown. Each line represents a narrower Concept displayed from the year of the first paper published with this Concept to the last paper. Figure 2 shows the wiki page for the Concept *Reusability*. Along with

[26] `www.graphviz.org/`
[27] Toeska Research Group, Universidad Técnica Federico Santa María: `www.toeska.cl`
[28] `www.semantic-mediawiki.org`
[29] SIMILE Project: `http://simile.mit.edu/timeline/`

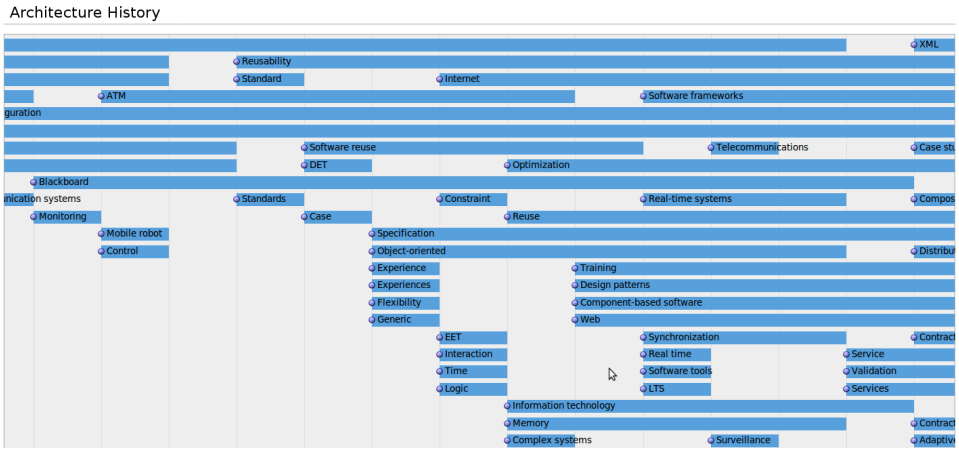


Fig. 1. Screenshot of ConcepTion SABOK Timeline - Architecture Concept Evolution

the information of broader concepts and narrower Concepts a Timeline of the publications using this Concept is provided. The Timeline is fully interactive and allow user to browse research papers. Figure 3 shows two infoboxes: Digital Asset and Concept. Digital Asset's infobox displays useful information such as title, author and Concepts used on this paper. It also provides information of inferenced Concepts related to the paper. Concept's infobox the upper and lower concepts in the hierarchy. It also displays inferred concepts and Digital Assets associated to the concept.

6 Further Work

Along with adding more advanced NLP tools and adding more papers to the analysis to improve our hierarchy, we believe that there are two topics that could add a lot of value to the SABOK presented.

- Cluster Analysis: Through cluster analysis we can understand better which are the areas the discipline is divided into. Also, it should be possible to acknowledge some useful intersections of areas and define them as different elements in the ontology to improve searching capability. We think that using Formal Concept Analysis tools would allow us to find this clusters of information by identifying classes of concepts as shown on PACTOLE methodology [3].
- Emerging Topics Tracking: With our approach is possible to find which are the most newer topics in the discipline and how they are related to each other. However, that does not mean that these are emerging topics. We think that emerging topics have a low frequency and thus, they will not emerge on our hierarchy. Besides that, we think that emerging topics appears on publications with a high impact factor and that's how we think that they

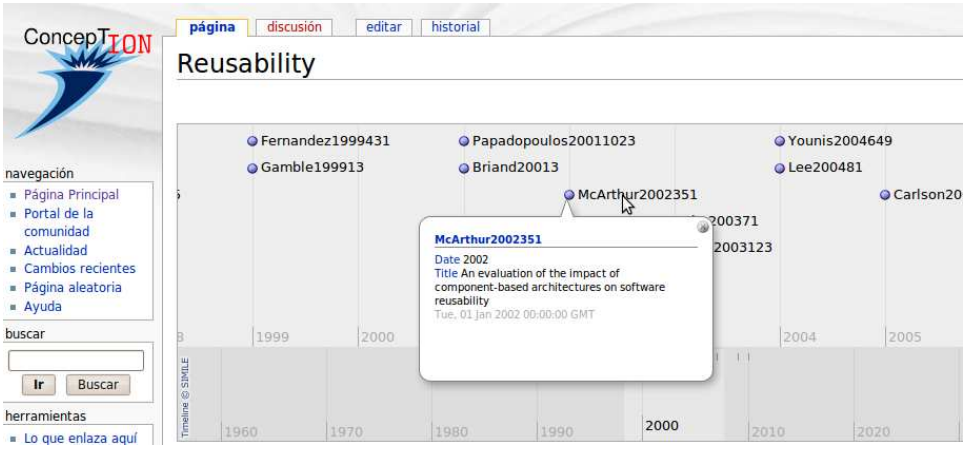


Fig. 2. ConcepTion wiki - Concept Reusability

should be identified. Though, that kind of information is not available on bibtex files and should be obtained on a different way.

Although we think the best validation for our SABOK should be made by the community, we are planning on making validation tests with Software Architects and Software Engineering students in the following months.

7 Conclusions

This article has presented a novel method to jump-start the creation of an ontology-based Body of Knowledge (BOK).

Using authorative documents from a community, we can mine and extract information about a discipline to hierarchize it and create an ontology. The ontology is used to organize the BOK and search Digital Assets (research publications in our example) using inference. The resulting BOK provides contextualization allowing document discovering and search inference.

The *ConcepTion* set of tools allows to extract, mine, hierarchize and display a BOK using a semantic wiki to manage information and a timeline tool to show evolution of topics in the discipline. The community is then asked to feed the BOK with definitions and their own Digital Assets. Future work will be focused on improving the quality of the resulting BOK and adding more features.

References

1. H. Astudillo. Maximizing object reuse with a biological metaphor. *TAPOS*, 3(4):235–251, 1997.

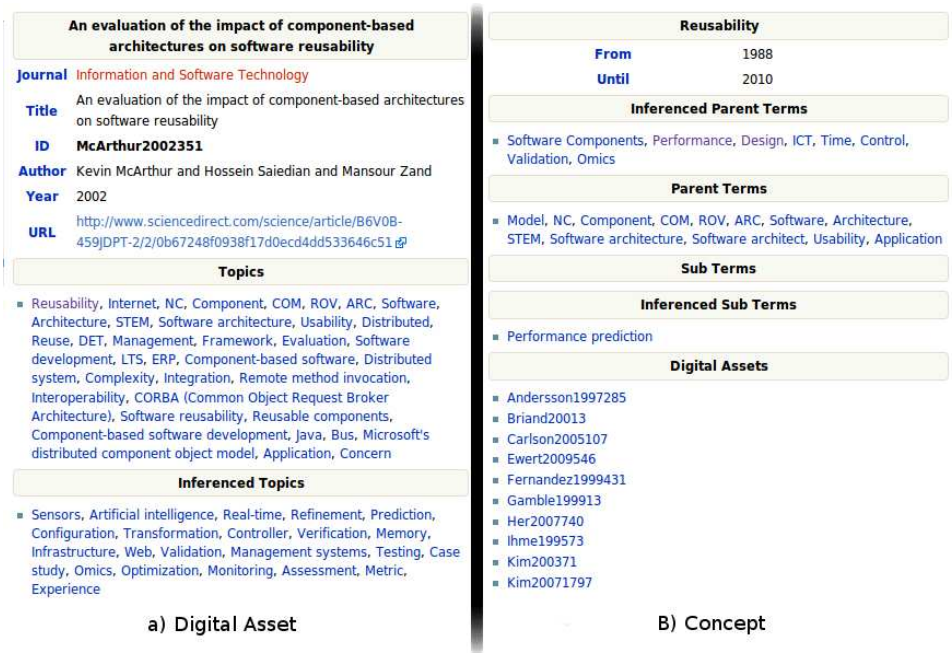


Fig. 3. Screenshot of ConcepTion Infoboxes

2. M. A. Babar, I. Gorton, and B. Kitchenham. Rationale management in software engineering. In *A Framework for Supporting Architecture Knowledge and Rationale Management*, pages 237–254. Springer Berlin Heidelberg, 2007.
3. R. Bendaoud, Y. Toussaint, and A. Napoli. Pactole: A methodology and a system for semi-automatically enriching an ontology from a collection of texts. 5113:203–216, 2008.
4. F. Bry, M. Eckert, J. Kotowski, and K. A. Weiand. What the user interacts with: Reflections on conceptual models for semantic wikis. In C. L. 0002, S. Schaffert, H. Skaf-Molli, and M. Völkel, editors, *SemWiki*, volume 464 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.
5. R. Capilla, F. Nava, S. Pérez, and J. C. Dueñas. A web-based tool for managing architectural design decisions. *SIGSOFT Softw. Eng. Notes*, 31(5):4, 2006.
6. H. Cunningham. *Encyclopedia of Language and Linguistics*, chapter Information Extraction, Automatic, pages 665–677. 2nd edition, 2005.
7. A. Fraga, S. Sánchez-Cuadrado, J. Lloréns, and H. Astudillo. Knowledge representation for software architecture domain by manual and automatic methodologies. *CLEI Electron. J.*, 9(1), 2006.
8. T. R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *Int. J. Hum.-Comput. Stud.*, 43(5-6):907–928, 1995.
9. A. D. Iorio, V. Presutti, and F. Vitali. Wikifactory: An ontology-based application for creating domain-oriented wikis. In Y. Sure and J. Domingue, editors, *ESWC*, volume 4011 of *Lecture Notes in Computer Science*, pages 664–678. Springer, 2006.
10. A. Jansen, J. van der Ven, P. Avgeriou, and D. K. Hammer. Tool support for architectural decisions. In *WICSA '07: Proceedings of the Sixth Working IEEE/IFIP*

Conference on Software Architecture, page 4, Washington, DC, USA, 2007. IEEE Computer Society.
11. M. Krötzsch, D. Vrandecic, M. Völkel, H. Haller, and R. Studer. Semantic wikipedia. *J. Web Sem.*, 5(4):251–261, 2007.
12. P. Kruchten, P. Lago, and H. van Vliet. Building up and exploiting architectural knowledge. In *QoSA'05: Second International Conference on Quality of Software Architectures*, pages 43–58. Springer Berlin / Heidelberg, 2006.
13. S. R. Kruk, M. Cygan, A. Gzella, T. Woroniecki, and M. Dabrowski. Jeromedl: The social semantic digital library. In S. R. Kruk and B. McDaniel, editors, *Semantic Digital Libraries*, pages 139–150. Springer, 2009.
14. B. T. Lenin, S. R. M., P. T. V., and R. D. ArchVoc-Towards an ontology for software architecture. In *SHARK-ADI '07: Proceedings of the Second Workshop on SHAring and Reusing architectural Knowledge Architecture, Rationale, and Design Intent*, page 5, Washington, DC, USA, 2007. IEEE Computer Society.
15. P. Liang, A. Jansen, and P. Avgeriou. Selecting a high-quality central model for sharing architectural knowledge. *Quality Software, International Conference on*, 0:357–365, 2008.
16. C. López, P. Inostroza, L. M. Cysneiros, and H. Astudillo. Visualization and comparison of architecture rationale with semantic web technologies. *Journal of Systems and Software*, 82(8):1198–1210, 2009.
17. Project Management Institute. *A Guide to the Project Management Body of Knowledge (PMBOK Guide) - Third Edition, Paperback*. Project Management Institute, 2004.
18. S. Schaffert. Ikewiki: A semantic wiki for collaborative knowledge management. In *WETICE '06: Proceedings of the 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pages 388–396, Washington, DC, USA, 2006. IEEE Computer Society.
19. S. Schaffert, J. Eder, S. Grünwald, T. Kurz, and M. Radulescu. Kiwi — a platform for semantic social software (demonstration). In *ESWC 2009 Heraklion: Proceedings of the 6th European Semantic Web Conference on The Semantic Web*, pages 888–892, Berlin, Heidelberg, 2009. Springer-Verlag.
20. S. B. Shum, E. Motta, and J. Domingue. Scholonto: an ontology-based digital library server for research documents and discourse. *Int. J. on Digital Libraries*, 3(3):237–248, 2000.
21. A. Tang, Y. Jin, and J. Han. A rationale-based architecture model for design traceability and reasoning. *J. Syst. Softw.*, 80(6):918–934, 2007.
22. L. L. Tripp. *Guide to the Software Engineering Body of Knowledge: 2004 Version*. 2005.

TasTicWiki: A Semantic Wiki with Content Recommendation

Manuela Ruiz-Montiel, Joaquín J. Molina-Castro, and José F. Aldana-Montes

Departamento de Lenguajes y Ciencias de la Computación,
University of Málaga, España
manuela.ruiz.montiel@gmail.com,{jmolina,jfam}@lcc.uma.es

Abstract. Wikis are a great tool inside the Social Web, as they provide the chance of creating collaborative knowledge in a quick way. Semantic wikis are becoming popular as Web technologies evolve: ontologies and semantic markup on the Web allow the generation of machine-readable information. Semantic wikis are often seen as small semantic webs as they provide support for enhanced navigation and searching of their contents, just what the standards of the Semantic Web aim to offer. Moreover, the great amount of information normally present inside wikis, or any web page, creates the necessity of some kind of filtering or personalized recommendation in order to lighten the search of interesting items. We have developed TasTicWiki, a novel semantic wiki engine which takes advantage of semantic information in order, not only to enhance navigation and searching, but also to provide recommendation services.

Key words: semantic wikis, recommender systems, ontologies

1 Introduction

A wiki is a web site with collaboratively edited pages. Users of the wiki perform these editions through the browser, in a quick way and without restrictions. Each page or article has an unique identifier, so they can be referenced from anywhere inside or outside the wiki. The general features of wikis are the following [1]: editing via browser with a simplified syntax -rather than HTML tags-, collaborative editing, non-lineal navigation thanks to a large number of hypertext links to other wiki pages, search functions and support for uploading non-textual contents.

We have developed a wiki engine, TasTicWiki, which seizes semantic technologies in order to offer sophisticated functionalities as well as semantic-enhanced recommendation services, in order to enlighten the tedious searching tasks derived from the potential existence of a vast amount of articles. In the next sections we will explain how this objectives are achieved as well as the architecture and features of TasTicWiki.

2 Semantic Wikis

Semantic Wikis are traditional wikis extended with semantic technologies like OWL or RDF. The goal of this enrichment is to make the available information machine-readable, so presentation, navigation, searching and even edition can be improved in a sophisticated way. This is usually done by adding meaning to the strong linking present in every wiki: the links are not mere hypertext anymore, as they represent meaningful relations among articles, or between articles and data types.

Common features of all approaches to semantic wikis are the following [3]: typing/annotating of links, context-aware presentation, enhanced navigation, semantic search and reasoning support. Some of the existing semantic wikis delegate the responsibility of creating the knowledge base to the final users of the wiki, allowing them to define meaningful relations practically without restrictions. Others rely on already defined ontologies that form the knowledge base, so the relations to be used are defined and restricted from the beginning. In `http://semanticweb.org/wiki/Semantic_Wiki_State_Of_The_Art#Active` we can find a list of the currently active semantic wikis.

3 Semantic Recommender Systems

In this section we briefly introduce how semantics can be taken advantage of in the context of recommender systems, and how it improves the results as they take into account the truly underlying reasons that determine the users satisfaction or dissatisfaction about the items.

Traditional Collaborative Filtering algorithms proceed by calculating similarities between users or between items [6]. These similarities are based on the ratings given to the items by the users. In the first case (user based), a user will receive recommendations made up of the items that similar users liked best. In the second case (item based), the recommended items will be those that are similar to the ones that the user loved in the past. This latter approach is known to be more efficient, since the similarities can be calculated off line [7].

If semantic features are taken into account, then the similarities could be computed according to them. This is what we call *Semantic Filtering Recommendation*. Indeed, the semantic features are the underlying reasons owing to which the items are similar or not. As we will see in section 4.5, our item-based approach for developing for a semantic recommender systems is based on domain ontologies containing the semantic attributes for the items. We use OWL ontologies and a reasoner able to classify the described resources.

4 TasTicWiki

TasTicWiki is a wiki engine that supports the creation and management of semantic wikis with recommendation services. This engine is born from the mixture between the ideas of semantic wikis and semantic recommender systems. Both

of them are utterly better off with the addition of semantic annotations, and we have developed an architecture where this extra information can be used in an homogeneous way, both for the semantic wiki and the recommendation system sake. In fact, we can see the recommendation services as an enhancement of wiki search, which is purely one of the leading leitmotivs of adding semantics to wikis.

4.1 TasTicWiki architecture

In figure 1 we illustrate the architecture of TasTicWiki.

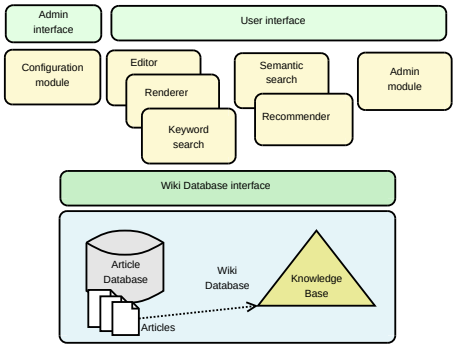


Fig. 1. TasTicWiki architecture

Every article in TasTicWiki is stored inside the database and it also corresponds to an instance inside the knowledge base, i.e., the ontology. The semantic metadata is thus stored separately from the page content, but we have set up a cache inside the database that will serve basic semantic information at the time of rendering and making certain type of queries, for the sake of reducing time processing. We use the knowledge base only when an article is firstly classified and when the users request queries involving complex axioms. The modules over the database interface are the ones that implement the functionality of the wiki. The *admin module* is devoted to administrative tasks such login, logout, registration, management of user profiles, etcetera.

Knowledge Base. TasTicWiki relies on a background ontology preloaded in the knowledge base. This domain ontology depends on the specific topic of the wiki. For example, we have developed a domain ontology in the field of tourism, since we have implemented a wiki[1] for a tourist information system. This background ontology has to fulfill some conditions in order to be used as a logic model in our knowledge base. It needs two main classes or concepts: one for storing the articles and another one for the different features the articles may have. We need

[1] `http://khaos.uma.es/wikitrip`

at least one role connecting the former with the latter, i.e., a *hasFeature* role -but nothing prevents the existence or others roles.

As an example, we briefly explain the skeleton of the tourism ontology we have developed. It has a *Tourist Service* class devoted to store the instances of the regular articles inside the wiki. These instances are related to the instances inside the class (or subclasses of) *Tourist Service Feature*, via some roles including *hasFeature* -we have three more roles as sub roles of the last one: *hasTradeActivity*, *hasSportActivity* and *hasSpecialty*. In addition, we count with some data roles establishing properties like the price, opening and closing times, etcetera.

In order to make the ontology expressive enough, we have defined some sub classes of the article class (i.e., the Tourist Service one). They are in much cases defined with complex axioms, e.g., there is a class called *Department Stores* which is defined as a service with at least two different trade activities. Another example can be *Inexpensive Accommodation*, defined as every *Accommodation Service* whose price is lower than thirty euros. The idea behind these definitions is that the Knowledge Base will perform some reasoning over the annotations the users include inside the text of the articles.

4.2 Semantic Annotation

When creating or editing an article, users in TasTicWiki may include two kinds of semantic annotations. This is done by special wikitext commands, and they consist in: a) annotations about features and b) annotations about categories the article belongs to. In a), the system needs the user to specify both some role and some feature value (or equivalently, some data role and some data value). In b), only the name of the category is needed. In 4.6 we will show an example of the wiki text used to add this semantic annotations.

4.3 Enhanced navigation and presentation

When rendering an article, TasTicWiki provides a Semantic Box, which summarizes all the available semantic metadata. The kind of information present in the Semantic Box depends on the type of article that is being rendered. Indeed, inside TasTicWiki exists a clean classification of articles depending on their concrete roles, as we explain in the next section. In figure 2 we can see an article with its Semantic Box.

Types of articles. In this section we describe each type of article in TasTicWiki and some details about them.

- **Regular articles**: they are the standard articles of the wiki, i.e., those whose purpose is just spreading knowledge about some particular topic. In our example, they would be the articles standing for Tourist Services. Users are allowed to insert semantic annotations in their wikitext. The information contained in their Semantic Boxes are: asserted features and categories

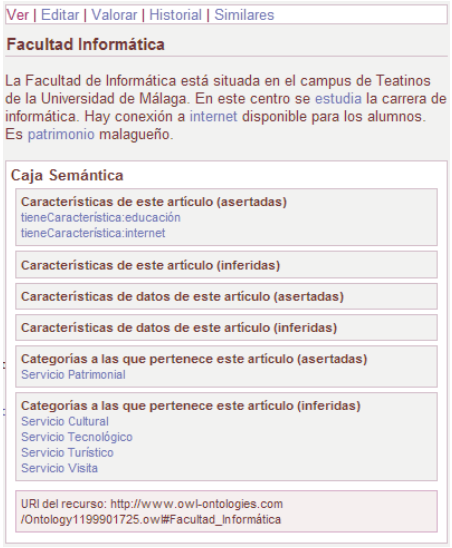


Fig. 2. An article inside the wiki, with its Semantic Box

(i.e., those explicitly specified by the users with semantic annotations) and inferred features and categories (those inferred by the reasoner).
- **Special articles**: they represent ontology entities like categories, feature concepts (in which we can found lists with feature values), feature values themselves, roles and data roles. Users are not allowed to add semantic annotations on them, but they can edit the wikitext in a pure textual way. Their semantic boxes show structural information like sub and super classes, domains and ranges, etcetera.

Users may create regular articles and feature value ones, but not the articles corresponding to concepts or roles. In other words, they are not allowed by the moment to edit the architecture of the background ontology (only their instances). This is considered as future work on the TasTicWiki system.

4.4 Enhanced search

Traditional wikis normally offer some kind of keyword, textual search. This sometimes is not powerful enough to retrieve the articles we need, as keywords do not really grasp the semantics underneath. In TasTicWiki we have developed a semantic search module, in which users, through a friendly, graphical interface, will be able to build and share complex queries based on complex ontological axioms.

It is not only about typical database search like *tell me all the services with a price lower than thirty*. It goes beyond, as complex axioms aim to recover articles

following not only the explicitly provided information, but implicit knowledge as well. As we are working with OWL ontologies, these axioms are the ones who exist in OWL DL: cardinality restrictions, universal and existential quantifiers, value axioms, negation axioms and membership axioms, with logical connectives as glue. In figure 3 we can see the interface for building complex queries.

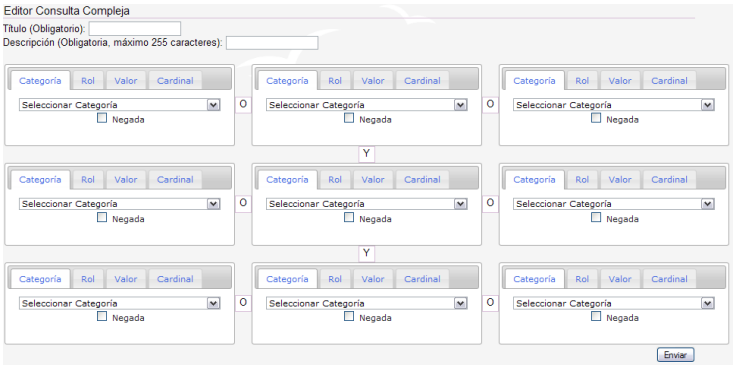


Fig. 3. Interface for complex queries. It is a matrix of atoms in which the user can specify some logic axioms that the obtained articles have to fulfill.

4.5 Recommendation

In Semantic Filtering Recommendation [10], we compute similarities between articles depending on the available semantic metadata. Then, given a set of well rated articles in the past, we compute the final recommendations. In next sections we go through the details of this process.

Analyzing users interactions. Inside a wiki we have several sources of information that can be taken into account at the time of computing the satisfaction of users. The direct one is collecting explicit ratings about the articles, asking for a numeric evaluation. But we also can take advantage of the previous behavior of the user inside the system: searchings, readings and editions. These last source is somewhat wiki-specific and, though by the moment is only used as a numeric coefficient (i.e., we only focus on the *quantity* of editions), an immediate future work way is taking into account the *quality* of the editions, mostly the semantic

ones. This source of information could be used not only for the recommendations sake (e.g., we could infer semantic categories or features of which the user is a connoisseur), but also for supporting the edition tasks, offering suggestions of possible annotations that would go well with the current wiki text.

The *configuration module* allows the administrator of the system to decide a weighting coefficient of all these factors in order to compute the satisfaction degree that an article has for an user (e.g., we could consider that explicit ratings are more important than the rest of factors). We need this degrees in order to build the input for the recommendation algorithm described in the next section.

Recommendation process. Providing we have a set of articles that satisfied a given user to some extent, computed from the study of the past interactions that the user has performed inside the wiki as we explained in the previous section, we are now able to compute the final recommendations. Given a well-rated article, its neighborhood is the set of the n most similar articles in the system. The similarity between two articles is calculated as follows:

$$sim_{i,j} = \frac{|SIP(i) \bigcap SIP(j)|}{max(|SIP(i)|, |SIP(j)|)}$$

Where *SIP*(i) is the *Semantic Item Profile* of the item (article) i, calculated by means of the Article Ontology -i.e, it is the set of semantic categories the item i belongs to. Note that similarities range from 0 to 1.

Once we have computed all the neighborhoods of the well-rated articles, we recommend those items in the union of all the neighborhoods that fulfill the next two conditions: the article has not been read by the selected user and the *Recommendation Factor*, which is a measure of how good the recommendation will be for an user, is bigger than a certain number, called *Recommendation Threshold*[2]. The Recommendation Factor is calculated as follows:

$$RF(i) = r(father) * sim_{i,father}$$

Where *father* is the article from which the neighborhood was calculated. If an article belongs to more than one neighborhood, then we take into account the biggest factor of all the possible recommendations. The *Recommendation Threshold* that we use to filter the items depends on the ratings domain and could be parametrized, as well as the size of the neighborhoods -in terms of percentage of the total number of articles in the system.

4.6 An example

Let us imagine an user who is going to use the wiki for a while. We will see through a simple, brief example how this experience will be like. In `http://khaos.uma.es/wikitrip` we can find the concrete wiki used for this example, called Wikitrip, developed in the topic of tourism services inside Malaga, Spain.

[2] This threshold can go from 0 to the upper limit of the ratings, e.g., from 0 to 5

Editing an article. The user wants to create an article about a hotel where he stayed during his last holidays. It was a three-star hotel with lift, private bathroom and a price of forty euros. Moreover, he consider that its category is medium. Among other textual information, the user wants to specify this four semantic annotations, task which will be performed by special wikitext commands:

> ...Astoria Hotel has [[feat:hasFeature:with lift/lift]], [[feat:hasFeature:private bathroom]] and a price per night of [[dfeat:hasPrice:30]] euros. Is is a [[cat:Medium Category]] service and...

As we can see, special, different commands are used depending on whether we are specifying features (*feat*), categories (*cat*) or data features (*dfeat*).

Navigation and presentation. Once the user has saved this article, it will be presented with links in the places where the semantic annotations were inserted.

- For features (i.e., with *lift* and *private bathroom*) a link to the corresponding *feature value article* will be rendered.
- For categories (i.e., *Medium Category*) a link to the *category article* will be rendered.
- In the case of data features, it makes nonsense to render a link to the value *30 euros*. Instead, an special type of link is generated: a query of all the articles inside the wiki which have a price of 30 euros.

Inside the Semantic Box of the article the user will find the most interesting pieces of information. Here, the system shows the implicit information extracted from the semantic annotations the user has inserted. Specifically, we will find that the article belongs to four categories: one explicitly inserted by the user (Medium Category) and three inferred by the reasoner, this is: *Tourist Service*, *Accommodation Service* and *Inexpensive Service*.

The information about the features will not be rendered in the semantic box as links to articles. Instead, these links lead to special queries which retrieve all the articles in the system related to the same value through the same role. In the case that implicit feature relations are inferred, they will be shown inside the Semantic Box as well.

Searching. The user can do some searching inside the wiki. It could be in a pure textual way, as in many traditional wikis, but also in a semantic way. Thanks to the underlying reasoner and a proper interface, the user will be able to make queries like: All the Catering Services with either a price lower than thirty or with at least three different specialties. Once the result list is computed, the user can read, edit and rate the given articles.

Recommendations. Once the user has read, searched, edited or rated some articles inside the system, the recommendation module will be able to compute a list of recommended items as we explained in section 4.5. If the user does not have any experience inside the system, then this list will be made up of the most popular articles (measured by explicit ratings).

5 Related work

Fred Durao and Peter Dolog have proposed a tag-based recommendation [11] as an extension for KiWi [5], with three slightly distinct approaches which offer different levels of performance and quality. In the more complex approach, they compute similarities between articles according to the tags the users have used to annotate them. Basically, this system differs from ours in the sense that we use reasoning in order to compute the *tags* -categories, indeed- and they only rely on the users criteria. Nevertheless, they plan to develop some reasoning to infer semantic similarities between tags, but even in that case, our approach turns in another flavor, since we extract the tags or categories from the Knowledge Base. For example, if we have an article a talking about a catering service with a price of ten euros, and an article b about an accommodation service with a price of forty euros, our system will tag both of them with the concept *Inexpensive Service*, and we will use that information in order to compute the final recommendation.

6 Work status and Future work

TasTicWiki is currently in its beta version, providing the services we pointed out in previous sections. Some future work is actually needed: an internationalization module, some improvements in the edition interface -as well as taking advantage of semantics in the edition tasks-, OWL/RDF export, and of course, the possibility of editing the underlying ontology in a collaborative way.

Other issues like performance are also to be studied, since we are using rich, expressive ontologies that do not go well with complexity. Complex queries are hard to solve, and we need scalable reasoning able to respond within a tolerable time. DBOWL [12] is a persistent and scalable reasoner which stores the underline ontology using a relational database which could be integrated with our current repository, allowing the composition of complex queries with the right level of abstraction thanks to a special, ad-hoc query language.

Moreover, when complex queries are requested, we need the knowledge base to be prepared and adapted to every previous change in the annotations of articles. This means, in reasoning terms, that the underlying ontology needs to be classified regularly in order to show complete results, so more solutions in this field are to be investigated. Furthermore, an evaluation of the recommender system inside the wiki needs to be done.

7 Conclusions

TasTicWiki is a wiki engine that allows the creation and management of semantic wikis with recommendation services. Semantic metadata improves presentation and navigation inside the wiki. TasTicWiki relies on background, rich ontologies that make possible advanced reasoning tasks, improved searching and some sophisticated functionalities as content recommendation.

Acknowledgments. This work was supported by the ICARIA Project Grant, TIN2008-04844 (Spanish Ministry of Education and Science), the pilot project *Formación y desarrollo de tecnología aplicada a la biología de sistemas*, P07-TIC-02978 (Innovation, Science and Enterprise Ministry of the regional government of the *Junta de Andalucía*) and the project *Creación de un soporte tecnológico para un sistema de información turística*, FFI 055/2008 (Tourism, Sport and Commerce Ministry of the regional government of the *Junta de Andalucía*).

References

1. Wikipedia: Wiki. `http://en.wikipedia.org/wiki/Wiki#Characteristics` (2009)
2. Vlkel, M., Krtzsch, M., Vrandecic, D., Haller, H., Studer, R.: Semantic Wikipedia. Proceedings of the 15th international conference on World Wide Web. Pages:585 - 594 (2006)
3. Schaffert, S.: IkeWiki: A Semantic Wiki for Collaborative Knowledge Management. 15th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises. Pages 388 - 396 (2006)
4. Kuhn, T. AceWiki: AceWiki: A Natural and Expressive Semantic Wiki. Semantic Web User Interaction at CHI. (2008)
5. Schaffert, S., Eder, J., Grnwald, S., Kurz, T., Radulescu, M: KiWi - A Platform for Semantic Social Software (Demonstration). ESWC 2009: 888-892 (2009)
6. Adomavicius, G., Tuzhilin, A.: Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Trans. Knowl. Data Eng. 17(6): 734-749 (2005)
7. Mobasher, B., Jin, X., Zhou, Y.: Semantically Enhanced Collaborative Filtering on the Web. EWMF 2003: 57-76 (2003)
8. Rui-Qin Wang, Fan-Sheng Kong: Semantic-Enhanced Personalized Recommender System. International Conference on Machine Learning and Cybernetics. Volume: 7, On page(s): 4069-4074 (2007)
9. Linden, G., Smith, B., York, J.: Industry Report: Amazon.com Recommendations: Item-to-Item Collaborative Filtering. IEEE Distributed Systems Online 4(1): (2003)
10. Ruiz-Montiel, M., Aldana-Montes, J.F.: Semantically Enhanced Recommender Systems, On the Move to Meaningful Internet Systems: OTM 2009 Workshops, volume 5872/2009, pages 604 - 609 (2009)
11. Durão, F., Dolog, P.: Analysis of Tag-Based Recommendation Performance for a Semantic Wiki. 4th Semantic Wiki Workshop at the 6th European Semantic Web Conference, volume 464 (2009)
12. Roldán, M., Aldana-Montes, J. F. : Complete OWL-DL Reasoning Using Relational Databases. Database and Expert Systems Applications, volume 5690, pages 435 - 442 (2009)

Ideator - a collaborative enterprise idea management tool powered by KiWi*

Rolf Sint, Mark Markus, Sebastian Schaffert, Thomas Kurz

{firstname.surname}@salzburgresearch.at
Salzburg Research
Jakob Haringer Str. 5/3
5020 Salzburg
Austria

Abstract. *"The most difficult thing with ideas is not to have them. It's to find out if they're good [1]."* This position paper demonstrates the requirements for an idea management application and presents the idea management tool Ideator. The Ideator is a software tool which is currently under development and which offers innovative and flexible solutions to idea management in company environments. It is based on the semantic wiki KiWi that is a framework for semantic social software applications. We present several functionalities of the Ideator and show which modifications and extensions of KiWi are necessary for their realisation.

1 Idea management

Idea management as a part of innovation management is an important factor to increase the productivity of companies. It makes the development of new products more efficient and helps to structure the ideation process within the company. This saves costs and keeps a company competitive. Different stakeholders, like employees, customers, suppliers or business partners may create new ideas that appear in different forms. Some are small optimizations of processes within a company and others are hot topics like ideas for innovative products. Companies that support the idea management benefit from the accumulated knowledge of its people [2]. In big companies, which support idea management like Deutsche Post, hundreds of ideas are collected every day. The managements challenge is the identification of the relevant ones from the whole amount of ideas. For this purpose all submitted ideas have to be evaluated according to different criterias, e.g. costs, benefits, innovativeness or the strategic relevance for the company.

* The research leading to these results is part of the project "KiWi - Knowledge in a Wiki" and has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 211932.

2 Enterprise 2.0 Idea management

Since the existence of web2.0 sites like Facebook[1], Flickr[2] and Wikipedia[3] it is common practice that people use the internet to interact, collaborate and share things with each other. Users publish their content on blogs and wikis, discuss with each other and form online communities. According to Michael Koch and Alexander Richter in [3] more than 750 million users worldwide spend a high amount of their free time on social networking sites. The effects of the growth and the high acceptance of social software are relevant for companies, too. The term enterprise2.0 describes the use of social software in the environment of a company. They benefit from the high acceptance of these sorts of applications. Michael Platt describes in [4] that Web2.0 applications *"...represent a significant opportunity for organizations to build new social and web-based collaboration, productivity, and business systems, and to improve cost and revenue returns."* Currently some web2.0 based idea management tools exist on the market. Representatives are BlueKiWi[4], BrainR[5] and Ideascale[6]. They have in common the easy creation of ideas and offer special support for communities.

3 Ideator - a collaborative idea management tool based on the Semantic Wiki KiWi

The Ideator is a web based idea management tool that combines the web2.0 philosophy with semantic web technology. The Ideator has its name from idea and motor and allows an innovative way of exploring and navigating within content and an effective filtering, search and visualization of ideas. We decided to use the semantic wiki KiWi as a framework for our tool. The KiWi system offers a flexible extension mechanism that allows the creation of semantic social software applications based on the KiWi core system. The important point is that most of the required functionalities of semantic social software applications are provided by KiWi and can be easily adopted for specific applications in different domains [5].
The KiWi core system offers

... several functionalities to support communities and which are typical for semantic social software applications, e.g. dashboard, social networking functionalities, collaborative tagging, ...
... a wiki based way to create, edit and link content
... forms which allow a structured acquisition of data and the possibility to transform unstructured wiki based data into structured form based data

[1] http://www.facebook.com
[2] www.flickr.com
[3] www.wikipedia.com
[4] http://www.bluekiwi-software.com/
[5] http://www.brainr.de/
[6] http://www.ideascale.com/

... several ways to classify and navigate within content based on semantic web technologies
... an easy way to add domain specific functionalities
... relevant features for enterprise applications like permission management, web services, versioning

We consider these points as key success factors for an idea management tool. The following section will describe functionalities of the Ideator in more detail. In addition, there will be a description of the needed modifications of the KiWi system.

3.1 Ideator workflow and user roles

For the creation of a new idea the Ideator offers alternatively a wiki or a form based approach. This will be described in the next section in more detail. An idea manager has an overview over all ideas and has the possibility to search for ideas according to different criterions. Furthermore the idea manager has the possibility to sort out the relevant ideas from the irrelevant ones. Additionally he/she can redirect an idea for evaluation to a reviewer. Beside the official reviewing process the community has the possibility to vote for an idea, too. Figure 1 illustrates the different user roles and their relations within the Ideator.

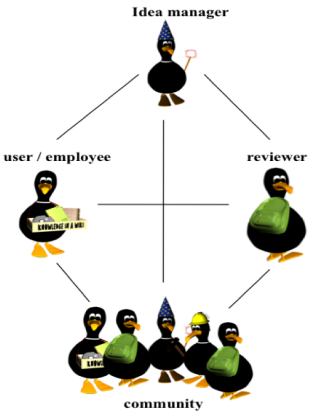


Fig. 1: user roles

3.2 Creating ideas the semantic wiki way ...

The Ideator tool allows the easy creation of new ideas according to the wiki philosophy and enables its systematically enrichment afterwards. The revolutionary

thing is that the Ideator focuses on the user and the ideas instead of the processes within a company: Everyone can use the Ideator to create ideas without the restrictions that appear in common idea management systems. Ideas can be acquired as unstructured textual data within the wiki and no forms and processes that limit the creativity are necessary. This is a very important aspect for motivating people to publish their ideas. Additionally the Ideator wiki supports versioning and the upload of different types of media content. The benefit of creating ideas the wiki style is that each idea can be linked to other ideas or related articles.
Furthermore the Ideator offers...

- a structured way to create new ideas
- the transformation of unstructured wiki data into structured data

Creating content the wiki way does not guarantee that all necessary information is given. It is uncertain whether the costs of the ideas realisation are included in the wiki text and whether the idea is categorized. Only forms can guarantee the entry of data according to a specification by telling the user what he/she has to fill in. On the one hand this guarantees a complete acquisition of data and on the other hand it is the reason why several enterprise applications are exclusively based on forms. In contrast to traditional wikis, where no structure of content exists, the Ideator is based on the semantic wiki KiWi and its data appears in a semi-structured form [6].
The Ideator allows people to create ideas according to the wiki philosophy and offers additionally a form-based approach to enrich systematically the information. A user can choose forms from a pool and use them to add additional information during the runtime of the application. Unstructured wiki text can be annotated using RDFa.
The RDFa primer describe RDFa in [7] as a *"...set of XHTML attributes to augment visual data with machine-readable hints."* It allows the annotation of free text according to concepts in an ontology. Some paragraphs or entities in the wiki article can be annotated with RDFa tags and as a consequence their values appear in the form. The Ideator supports the user in entering RDFa tags with a simple user interface that allows the selection of all possible RDFa properties. Additionally several entities are automatically detected by the system based on information extraction. Figure 2 illustrates the transformation of unstructured wiki text to structured and form based information.

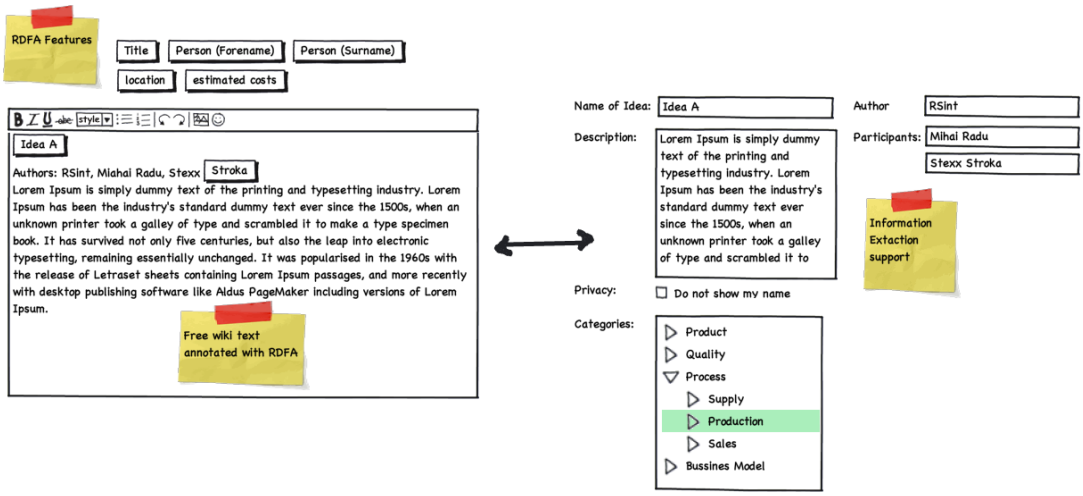


Fig. 2: unstructured wiki text combined with forms

3.3 Extended Community support

Like several other social networking websites the Ideator allows the user to administer the personal profile and to add other users as friends. In this way a user will be notified about all submitted ideas of friends. Each idea can be assigned to a user and it is possible to comment and rate ideas of others. Furthermore an idea can be put on a personal watchlist and the user will be informed about all activities of the containing ideas, e.g. changes of the content, new comments, new ratings, tagging, viewing, etc. A special functionality of the Ideator is the possibility to analyse and increase the activity of the community. This is provided by the integrated reputation mechanism Community Equity[7]. By using this model each activity on an existing idea increases its activity value and as a consequence the reputation of a user gets higher, too. The Community Equity mechanism is an integral part of the KiWi core system and a detailed description about the algorithm can be found in [8]. In this way the most relevant ideas and the most active users are identified.

[7] http://kenai.com/projects/community-equity

3.4 Dashboard

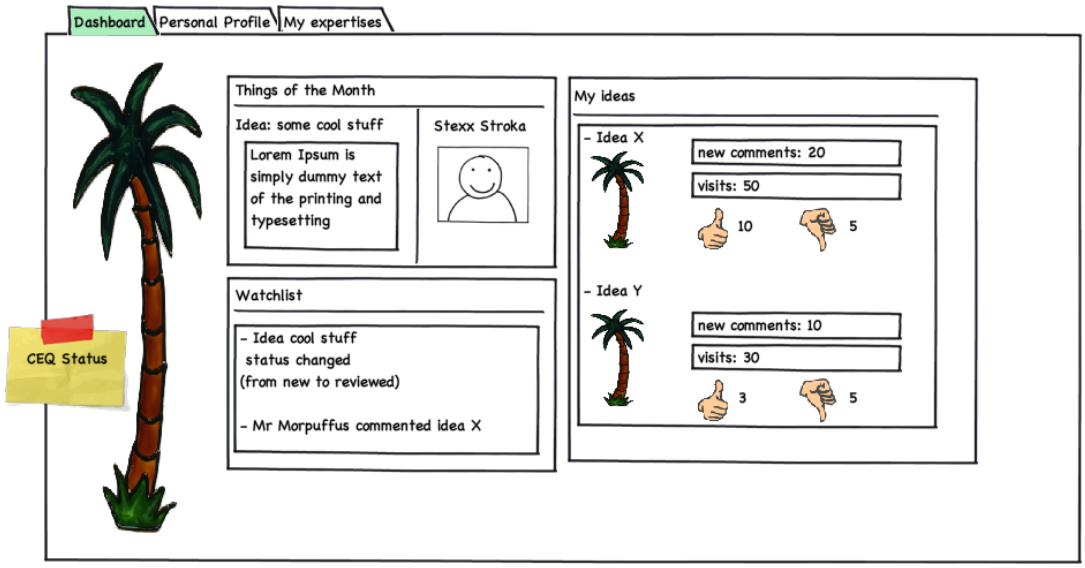


Fig. 3: The ideator dashbaord

After login a user is redirected to the dashboard that allows a personalized view on the content and provides a quick overview of the activities in the community. That informs the user about

-all new submitted ideas
-the best rated ideas
-the activities in the community based on the community equity mechanism described above
-the history of a users visited wiki pages

The dashboard is used to manage a users personal profile and friendlist, which is the primary way to use the social networking functionality of the Ideator. Furthermore the dashboard motivates a user by illustrating its activity in form of a palm. The more active the user submits, rates or comments ideas the bigger the palm is. The same visualisation technique is used to visualize the activity of each idea. Additional to the official reviewing workflow each user has the possibility to vote for an idea: The user can support an idea by clicking the like button, illustrated in figure 3 in form of a thumb.

3.5 Navigation of content / Exploring new ideas

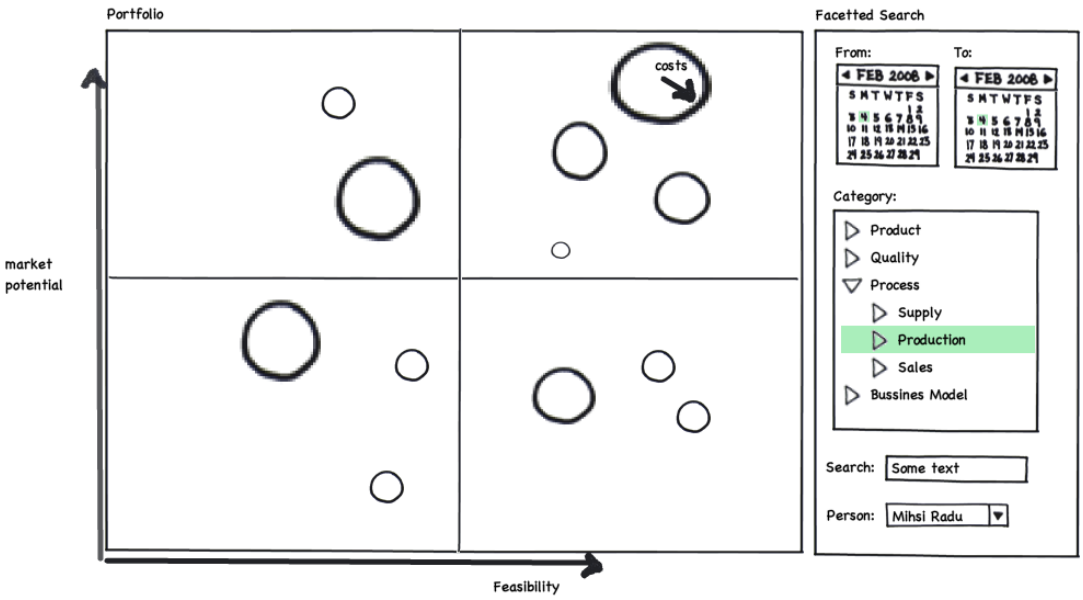


Fig. 4: Idea Portfolio

An idea manager needs to have an overview over all ideas. For this purpose the Ideator offers a facetted search combined with an attractive and informative result representation. Each idea is visualized according to three dimensions within the portfolio matrix: potential for the market, feasibility and costs. Ideas, which have a high feasibility and a high potential for the market, are in the upper right corner of the portfolio. Ideas with a low feasibility and a low potential for the market are in the lower left. The size of the bubble illustrates the costs of the idea, i.e. the bigger the bubble the higher the costs. Each bubble itself deals as a link to the corresponding idea. This visualisation helps an idea manager to sort out relevant ideas from irrelevant ones. All visualized ideas can be filtered according to different criterions by a facetted classification mechanism. The facets are illustrated on the right side of figure 4. The semantic wiki KiWi offers the basis for the facetted search and allows an easy adoption of the visualisation for the search results.

3.6 Conclusion

In this paper we introduce into idea management and demonstrate how KiWi can be used to build an enterprise2.0 application. For this purpose we present the idea management tool Ideator. Especially in the domain of innovation management a high user participation is very important. Only if employees, suppliers and customers participate actively in the idea management process, good and economic ideas can grow. The Ideator is a very user centered application which offers several functionalities which are typically for semantic social software applications like the user profile, the dashboard and the possibility for a community to vote for an idea. People can put ideas on a watchlist and get informed about changes on this idea. Additionally a very innovative reputation mechanism motivates users to participate in the idea management process. Finding the idea is the one thing, sorting out the relevant ideas from the irrelevant ones is the other. The real strengths of the Ideator are the several possibilities to create, structure, navigate and search for data, which are enabled through the semantic basis of the KiWi framework.

References

1. Howland, C.: Zitat von Chris Howland. (`www.zitate-online.de/autor/howland-chris` (9.03.2010))
2. Eggert, J.: What Is Idea Management? (`http://www.idealeadership.com/About_IL.htm` (9.03.2010))
3. Koch, M., Richter, A.: Enterprise2.0. Ouldenburg, Germany (2007)
4. Platt, M.: The architecture journal. (`http://msdn.microsoft.com/en-us/library/bb735306.aspx` (9.03.2010))
5. Schaffert, S., Eder, J., Sint, R., Grünwald, S., Stroka, S., Kurz, T., Radulescu, M.: KiWi - A Platform for Semantic Social Software, Fourth Workshop on Semantic Wikis, ESWC2009 (2008)
6. Sint, R., Schaffert, S., Stroka, S., Ferstl, R.: Combining Unstructured, Fully Structured and Semi-Structured Information in Semantic Wikis, Fourth Workshop on Semantic Wikis, ESWC2009 (2009)
7. Adida, B., Birbeck, M.: RDFa Primer. `http://www.w3.org/TR/xhtml-rdfa-primer/` (9.03.2010) (2007)
8. Schaffert, S., Siorpaes, K., Reiser, P., Radulescu, M., Riachtchentsev, D.: Community Equity - A Reputation and Incentive System for Vibrant OnlineCommunities. (to appear)

Towards Meta-Engineering for Semantic Wikis

Jochen Reutelshoefer, Joachim Baumeister, Frank Puppe

Institute of Computer Science, University of Würzburg, Germany
{lastname}@informatik.uni-wuerzburg.de

Abstract. Building intelligent systems is a complex task. In many knowledge engineering projects the knowledge acquisition activities can significantly benefit from a tool, that is tailored to the specific project setting with respect to domain, contributors, and goals. Specifying and building a new tool from scratch is ambitious, tedious, and delaying. In this paper we introduce a wiki-based meta-engineering approach allowing for the smooth beginning of the knowledge acquisition activities going along with tool specification and tailored implementation. Meta-engineering proposes that in a wiki-based knowledge engineering project not only the content (the knowledge base) should be developed in evolutionary manner but also the tool itself.

1 Introduction

The development of knowledge-based systems still suffers from the *Knowledge Acquisition Bottleneck* yielding high development costs with respect to the knowledge acquisition efforts. Usually, a knowledge acquisition tool poses several constraints to the engineering process. The key feature of such tools are predefined user interfaces, the degree of formalization, and the way of how the knowledge is organized. However, often it appears, that for a given project, considering its contributors, domain and goal, a more appropriate solution might be imaginable, but not yet existing. Customizable and extensible (non wiki-based) tools for building knowledge-based systems are available today (e.g., Protege [1]). Another approach is building a customized knowledge acquisition tool from scratch. However, both approaches do not allow for evolutionary incremental specification and implementation of the tool in parallel with the (beginning) knowledge acquisition activities, but require the specification and implementation in advance. The specification of a knowledge engineering tool in advance bears another challenge: At the beginning of the cooperation of knowledge engineers and domain experts a sound specification is ambitious and risky. As the knowledge engineers initially are not familiar with the domain and the contributors (experience level, number, availability) to be able to define the best possible knowledge formalization method. We propose an incremental and agile approach, that allows for the smooth but immediate startup of knowledge formation and breaks down the constraints and entry barriers to a minimum. The (semantic) wiki-based approach demands from the contributors at the beginning only the capability of mastering the basic wiki workflow, which is browsing and modifying wiki pages. Its only

constraint is, that the knowledge can be entered or imported as wiki content and partitioned/organized into wiki pages. We argue, that this method retains a high level of flexibility being able to support a large number of requirements. However, the gap between a wiki filled with domain knowledge and an executable knowledge-based system, using a formal knowledge representation, is still large. In this paper, we discuss how this gap (emerging on conceptual and on technical level) can be bridged in an agile and incremental manner with reduced delay of the knowledge acquisition phase, and at moderate (software) engineering costs. The knowledge engineering tasks, we focus on, are the development of decision-support systems where solutions, based on entered (formal) problem descriptions in sessions, are proposed. We demonstrate the meta-engineering process for this task by sketching its implementation by several case studies.

The rest of the paper is organized as follows: In Section 2 the meta-engineering process is explained in more detail, considering the conceptual and the technical level. Further, we discuss in Section 3 how the Semantic Wiki KnowWE supports the meta-engineering idea on the technical level. Demonstrating the applicability of the proposed approach, we report on experiences made in different projects in Section 4. We conclude the paper by giving a summary and pointing out directions for future work.

2 The Meta-Engineering Process

The meta-engineering process proposes to model a knowledge acquisition process, that is free from any knowledge engineering tool or knowledge representation at the beginning. The initial phase tries to envision the optimal knowledge formalization environment for the project without regarding any technical constraints. The result is then developed towards a project tailored specification of a knowledge engineering tool. We argue that a wiki poses very low constraints, only demanding that the knowledge can be defined in some textual form, and because of this forms suitable basis for this approach. Thus, the question in the initial phase is, how the knowledge can be entered in a wiki in a *formalizable* way, regarding domain, contributors, startup knowledge, and goal. We call the result of this process the *wiki-based formalization architecture* optimizing criterias such as understandability, maintainability, and acquisition efficiency - yet disregarding any technical constraints. Figure 1 shows the cooperative phases of the meta-engineering process.

Fig. 1. The cooperative phases of the wiki-based meta-engineering process.

At first, in the experimental phase a small toy prototype is implemented using formalization methods (markup, reasoner) already provided by the tool. Even though, the available markups may be not optimal for this task, this phase gives the domain specialists an impression of wiki-based knowledge acquisition. Then, in the design phase possible markups and partitionings for the knowledge are developed in an iterative manner, forming the wiki-based formalization architecture. Small parts of the domain knowledge are entered in the wiki by using the currently specified architecture. Although, the tool cannot compile and process the knowledge at this point, discussing these prototypical knowledge artifacts can catalyze the exchange of expertise. Knowledge engineers obtain an impression of the knowledge that needs to be formalized, and domain specialists experience the general idea of wiki-based knowledge formalization. This allows for a better estimation of the formalization architecture criterias *understandability*, *maintainability*, and *acquisition efficiency*. That way, the formalization architecture can be revised and refined iteratively in joint discussion sessions. Due to the flexibility of the wiki approach these phases can easily overlap resulting in an agile specification and development process of the tool. The process finally leads to the main phase when design and implementation activities get finished, featuring a thoroughly tailored knowledge engineering environment.

2.1 Conceptual Level: Bridging the Gap of Expertise

As already stated in the introduction, it is often difficult to completely specify the most appropriate acquisition method and tool at project startup. Often, either the knowledge engineer or the domain specialist are not familiar enough with the other discipline respectively at the beginning. The wiki-based meta-engineering approach tries to overcome this gap of expertise by the two cooperative phases of experimentation and design. The *wiki-based formalization architecture*, forming the result of these phases, contains the following aspects:

1. **Identification and representation of important domain concepts:** This task defines how the domain concepts are represented in the wiki. Informal "support knowledge" (e.g., textual descriptions, images, links) is added to the concepts, describing the concept, defining a common grounding of their meaning, and documenting its role in the formal model of the domain. When possible, the support knowledge is streamed into the wiki by reusing legacy documents of the project context.
2. **The distribution of the formal knowledge:** The knowledge formalization architecture defines how the formal knowledge is organized in the wiki. The derivation knowledge typically connects the input concepts (findings of the problem description) with the requested output concepts. In general, the derivation knowledge is distributed according to a domain-dependent partitioning and attached to the wiki pages of the most related concepts. However, there is not yet a canonical receipt to select or create the optimal knowledge distribution for a given project in one step.

3. **Definition of the markup:** When defining the appropriate markup general design principles for domain specific languages (DSL) should be adhered to. Spinellis [2] emphasizes, for example, to include the domain experts closely into the design process. Karsai et al. [3] report about design guidelines like the reuse of existing language definitions and type systems or limitation of the number of language elements. In the context of wikis, also the intuitive and seamless integration with the informal support knowledge has to be considered. Hence, the documents are not exclusively created by using the DSL, which is in this case only forming the markup for fragments of the overall wiki content. Knowledge markups in wikis can be designed for use at different (syntactical) granularities: For example, large sections like table- or list-based markups or small relation atoms, that can be distributed as single items within the (informal) page content, are possible [4]. The first allows for a more comprehensive view on the formal knowledge fragments in an aggregated form. The latter in general allows for a better integration with the informal knowledge each relation being injected at the most suitable location in the text. In this case, additional views should be generated from the spread relations to provide concise overviews, for example by using inline query mechanisms.

In cooperative sessions the knowledge engineers together with the domain specialists try out different possibilities for the described aspects. For each idea some demo wiki pages, covering a very small part of the domain, are created, disregarding that the knowledge is not (yet) processed by the wiki.

When this iterative design process has lead to a promising result the implementation phase of the meta-engineering process begins, aiming at modifying the tool to be able to parse and compile the knowledge, according to the designed formalization architecture. At this point, the already inserted knowledge of the demo pages on the one hand can be kept, forming the first part of the knowledge base, and on the other hand serves as a specification for the markup.

2.2 Technical Level: Bridging the Gap of Tool Support

The design phase on the conceptual level identifies a specification of an appropriate (wiki-based) knowledge formalization architecture, for which in most cases no tool support is existing at that point. The gap between a standard wiki or even a standard semantic wiki to the envisioned tool in most cases is still large, for example if the support of production rules entered by a custom markup should be supported. However, the general process of parsing and compilation the knowledge in a wiki-based workflow is always similar. Figure 2 shows an outline of the wiki-based knowledge formalization process chain from the knowledge source (domain specialists or legacy documents) on the left to the productive running system on the right. There are four main steps involved, which are discussed in the following.

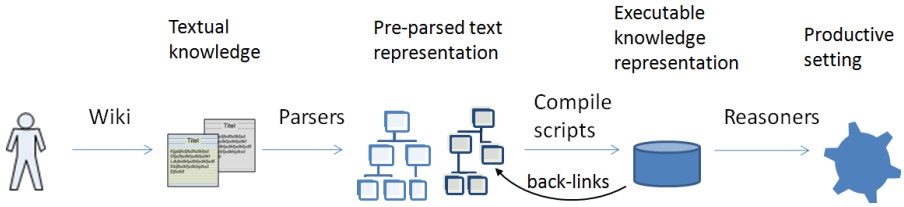


Fig. 2. The wiki-based knowledge engineering process chain.

1. **A wiki to create textual knowledge:** This is the essential part of a wiki application. The wiki interface is used to create textual documents. In general, any standard wiki engine can be used to accomplish this task.
2. **Parsers to create a pre-parsed representation of the textual knowledge:** To create formalized knowledge from text documents, markup needs to be defined. For the specific markup the corresponding parser components are integrated into the system. They create a (concrete) syntax tree of the wiki pages (also containing large nodes of free text). In this parse tree the structure of the formal relations, i.e., references on formal concepts and their relations, are contained.
3. **Compilation scripts to create executable knowledge:** The compile scripts transform the pre-parsed representation of the markup into an executable knowledge format, that can be interpreted by the reasoners. The compile scripts need to be defined with respect to the markup and its syntax tree representation and the target data structure defined by the intended reasoning engine.
4. **Reasoners to test the knowledge base:** Any reasoner that can solve the intended reasoning task of the application can be integrated. For the evolutionary development, testing of the knowledge base is necessary. Hence, components for the execution of the reasoner with the knowledge base need to be provided.

In general, the steps of parsing (2) and compilation (3) could be considered as one step in the process chain. However, separating them into two steps by the use of some structured text-representation has important advantages: Back-links from the formal knowledge artifacts to the corresponding, original text entities become possible. This allows to identify for each formal relation the exact location in the text that it was generated from. One can make use of this for the implementation of important tasks:

- **Explanation:** Explanation components presenting the text slices, that are responsible for the current reasoning result, can be created.
- **Validation:** For many knowledge representations and reasoners validation methods exist, detecting deficiencies like redundant or inconsistent knowledge.

Without the back-links the text location of the corresponding knowledge artifacts can not be identified to correct or tidy the wiki content, being the source

of the compilation process. In general, these two techniques—explanation and validation—are truly necessary to build up large well-formed knowledge bases using an agile methodology. Further, algorithms for refactoring of the knowledge heavily benefit from a pre-parsed text representation, and when exchanging the target reasoning engine only the compile scripts need to be modified and the parsing components remain untouched.

To help bridging the technical gap spanned by the designed formalization architecture and some existing tool, we propose the design of a framework. It needs to allow for the easy integration of missing elements and to provide a library of reusable components to fill the gaps in the formalization chain.

3 Meta-Engineering with KnowWE

KnowWE [5] has been designed to support the implementation of tailored formalization architectures in a process chain like sketched in 2. It provides a library of components that can be reused in the contexts of different projects and allows for the definition and integration of custom components at low implementation costs. KnowWE connects several free and open source software components to provide the knowledge formalization capabilities sketched in Figure 2. As basic wiki engine JSPWiki[1] is used. We integrated the reasoning engines *OWLIM*[2] for RDF reasoning and *d3web*[3] for diagnostic reasoning. To extend the system with additional components (e.g., parser, compile scripts, renderers,...), we provide a flexible plugin mechanism based on *JPF (Java Plugin Framework*[4]*)*. Besides the interconnection of these components forming a semantic wiki with problem-solving capabilities, the major technical contribution of KnowWE is the generic typed data-structure for the pre-parsed representation of the textual content as shown in Figure 2, called *KDOM* (Knowledge Document Object Model). The wiki documents are parsed according to the extensible *KDOM schema*, where all allowed textual entities are defined in a declarative way. A detailed explanation of the parsing algorithm creating the parse-tree of a document using a specified KDOM schema can be found in [6].

3.1 Parsing

The KnowWE core library contains components to configure a project specific KDOM schema. While parser components for common useful syntactical structures such as tables, bullet lists, dash-trees, or XML are provided by the system, for domain specific languages own parser components need to be defined. Figure 3 shows different applications of the dash-tree markup as decision tree, concept hierarchy and property hierarchy.

[1] http://www.jspwiki.org
[2] http://www.ontotext.com/owlim/
[3] http://www.d3web.de
[4] http://jpf.sourceforge.net/

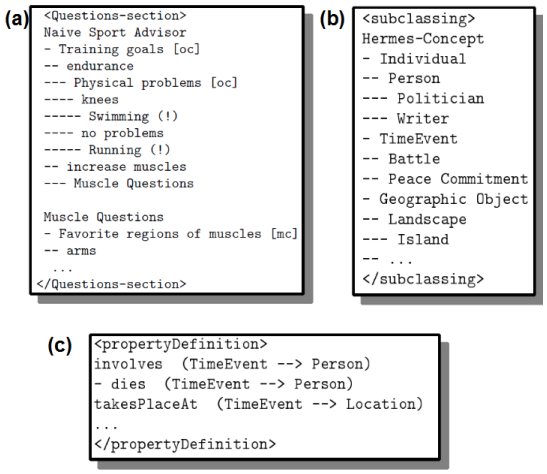


Fig. 3. Different applications of dash-tree based markups: (a) as decision tree in a sports advisor knowledge base; (b) as concept hierarchy in the historical domain; (c) as property hierarchy with domain and range definition

Customized reuse of predefined markup is possible by small modifications in the dash-tree KDOM schema components. Figure 4 shows the basic KDOM schema of the dash tree markup. The blue nodes represent KDOM types from the KnowWE core library, provided with the corresponding parsing component (i.e., regular expressions). Only small modifications at the dash-tree leaf of the schema are necessary to enable specific parsing and compilation tasks using dash-trees.

Recently, a bridge to the UIMA Framework[5], which is an open source framework for unstructured/semi-structured information processing, was integrated. Currently, experiments extracting formal knowledge from legacy documents, using different of the large number of UIMA analysis engines available, are run. The configuration of tailored information extraction components as a meta-engineering task, alternatively to the development of markup, aims at semi-automation of the knowledge acquisition process.

3.2 Compilation and Reasoning

Compile scripts can be attached to the KDOM schema being executed automatically after the corresponding subtree has been instantiated by the parsing process of a page. They walk the KDOM parse-tree and instantiate the executable knowledge in the corresponding repository, depending on the targeted reasoning engine. By the use of the unique IDs of the nodes of the KDOM nodes, the back-links described in Section 2 can be created.

[5] http://uima-framework.sourceforge.net/

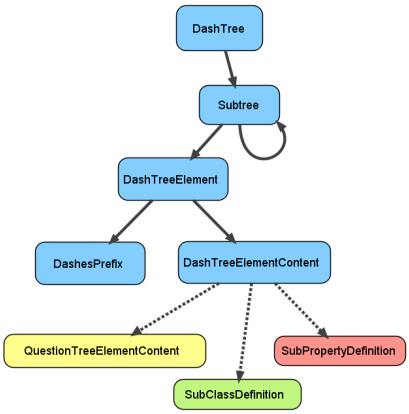


Fig. 4. A KDOM schema defining dash-tree syntax. Parsing and compilation of the leaf contents can be customized.

A general include mechanism, which is independent of markup or target representation, allows for specific compiling tasks. By using the include mechanism, any composition of knowledge fragments from various pages can be assembled to one wiki page for testing, generation of views, creation of variants, or export of knowledge bases. The KnowWE core currently integrates (swift-) OWLIM for RDF reasoning. Further, we integrated the d3web engine for diagnostic problem-solving together with basic markups [4] for knowledge formalization in d3web. The integration of the open source rule engine *Drools*[6] is currently in progress. For many applications these reasoners should provide sufficient possibilities to build at least a small prototypical executable knowledge base for demonstration supporting the initial cooperative design phase. The potential later integration of an optimized reasoning engine (e.g., that better scales on the specific reasoning task) only demands to modify the compile scripts and to provide an endpoint for the execution of the engine for testing purposes.

3.3 Refactoring

Meta-engineering proposes an evolutionary approach with respect to the (wiki-based) knowledge formalization architecture implying experimentation with different partitionings of the knowledge over the wiki pages and different markups. Therefore, for transferring content from one page to another or transforming one markup to another, automated support is crucial to prevent repetitions in the knowledge acquisition activities. To allow for efficient scripting of these transformations, we integrated a Groovy[7] endpoint into the wiki, accessible only to project admins. It allows to create and execute transformation scripts on the

[6] http://labs.jboss.com/drools
[7] http://groovy.codehaus.org

wiki content. The Groovy scripts makes use of a predefined API to access, scan and modify the KDOM data-structure. Figure 5 shows a refactoring script embedded in the wiki. At the top of the wiki page a GUI widget for the execution of the refactoring script is shown. Underneath, a Groovy script for renaming an object is located. As these refactoring script operations are complex and dangerous, they should be performed in offline mode (i.e., blocking wiki access for the general users at execution time).

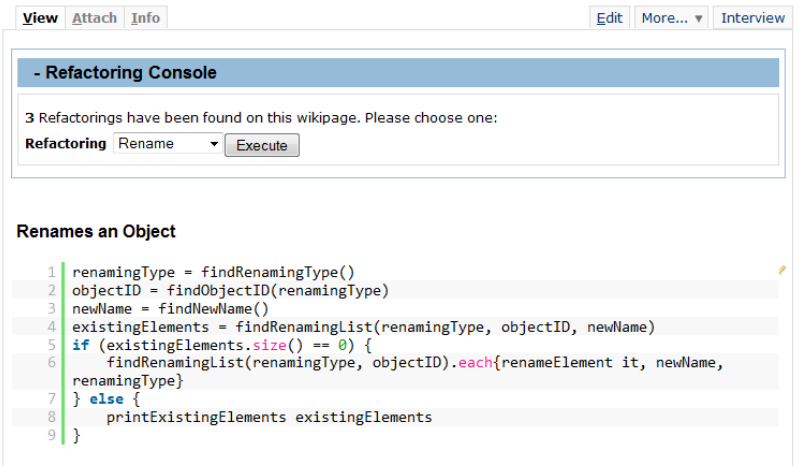


Fig. 5. A refactoring script in the wiki written in Groovy, which renames an object globally.

4 Case Studies

We are currently evaluating the meta-engineering approach within several industrial and academic projects using the KnowWE system. They are addressing a wide range of different domains like biology, chemistry, medicine, history, and technical devices. In some projects, the customizations only make small changes to the existing features while in others large components are created. In the following, we briefly introduce the projects and explain the employed meta-engineering methods.

4.1 Clinical Process Knowledge with CliWE

In another recent project, KnowWE is extended by diagnostic workflow knowledge in the context of the CliWE project[8]. This project considers the development of a medical diagnosis system for a closed-loop device. Documents describing the domain knowledge already exist and these are imported into the wiki as textual and tabular information about the domain. The particular wiki articles are focusing on special aspects of the diagnosis task, for example the assessment of the current patient's state. At predefined milestones the knowledge in the wiki is exported into a single knowledge base in order to evaluate the development on real-time devices.

In general, the core knowledge formalization methods from the d3web-plugin are used. However, to implement closed-loop systems, the need for an additional knowledge representation and reasoning support was identified at an early stage of the project. Thus, a flowchart-like visual knowledge representation has been designed. The flowcharts can be edited in the wiki using the integrated visual flowchart editor *DiaFlux*. To allow for modularization the flowcharts can be organized hierarchically. A (sub-) flowchart can be included into another flowchart as one (box-) component by defining and connecting the input/output interface. Due to this hierarchical organization, partitioning of the different aspects of the domain knowledge over the wiki, is possible. A first prototype of this extension is reported in Hatko et al. [9].

4.2 Fault Diagnosis for Special Purpose Vehicles

Another project considers the development of a diagnostic system for special purpose vehicles. The goal is to reduce the repair time and costs of the vehicles by determining the faulty element by a cost-minimal sequence of (potentially tedious) examinations. The system is build based on existing construction plans and heuristic knowledge of experienced mechanics. After an analysis phase the wiki formalization architecture has been defined. It contains one structural model of the vehicle, one state model of the current diagnosis session and for each technical subcomponent fault causes and malfunctions. For each of this knowledge base components own markup has been defined allowing logical distribution of the knowledge base over different wiki pages and seamless integration with support knowledge, such as technical documents and construction plans. The knowledge will be compiled into set-covering knowledge containing also the cost values for any examination at some given state for a sophisticated interview strategy calculated by an additional problem-solver. This wiki-based formalization architecture has been defined after an initial phase where one Excel-based approach and one wiki based approach using existing markup have been evaluated. Including a initial phase with iterative cooperative sessions and finally the technical implementation of the defined formalization architecture, the project shows a successful application of the Meta-Engineering approach.

[8] CliWE (Clinical Wiki Environment) is funded by Drägerwerk, Germany and runs from 2009-2012.

4.3 WISEC

The WISEC (**W**iki for **I**dentified **S**ubstances of **E**cological **C**oncern) project[9] investigates the management and detection of substances with respect to its bio-chemical characteristics. Here, substances of very high concern (SVHC) under environmental protection considerations are investigated and managed using the multi-modal approach of a Semantic Wiki: The knowledge about each substance is represented by an wiki article containing informal descriptions of the substance and its relations to external sources (via semantic annotations). The overall knowledge base also integrates already known lists of critical substances and explicit domain knowledge of a specialists combining the particular characteristics of the criticality of substances.

Tailored markups were created to capture the relation of the substances to already known critical substance lists. Thus, a list of critical substances in the wiki is still human-readable, but is also automatically compiled as a collection of ontological properties relating the substance concepts with the list concept. Furthermore, special properties (such as different toxic contributions) are also parsed as formal properties of the list concepts. Due to the explicit representation of the relational knowledge in OWL, different properties of substances can be queried over the wiki knowledge base.

4.4 Medical Decision-Support with CareMate

The *CareMate* system is a consultation system for medical rescue missions, when the problem definition of a particular rescue service is complex and a second opinion becomes important. The major goals of the project were the rated derivation of suitable solutions and the implementation of an efficient interview technique for busy rescue service staff in the emergency car. Thus, the user can be guided through an interview focusing on relevant questions of the current problem. With more questions answered the current ranking of possible solutions improves in relevance, and the interview strategy targets the presentation of reasonable follow-up questions.

For the CareMate project, the core entities of the formalization architecture are the cardinal symptoms, i.e., coarse findings describing vaguely the problem of the currently examined patient. The organization according to the cardinal symptoms is motivated by the observation, that in practice the emergency staff also tries to divide the problem by first identifying the cardinal symptom. Subsequently, the applicable domain knowledge can be easily partitioned with respect to the cardinal symptoms. The domain specialist provided the domain knowledge (interview strategy and solution derivation/rating) for each cardinal symptom in form of MS-Visio diagrams. Each cardinal symptom is represented by a distinct wiki article, and the corresponding derivation knowledge is defined using the knowledge formalization pattern *heuristic decision tree* [7]. In Figure 6 the wiki article of the cardinal symptom stomach pain ("Bauchschmerzen") is shown.

[9] in cooperation with the Federal Environment Agency (UBA), Germany

Here, the wiki text describes that the decision tree logic was divided into two decision trees handling the diagnosis of stomach pain for women and for men, separately. For both decision trees an image is shown (can be enlarged on click), that gives an overview of the general structure of the questionnaire and the inference. The lower part of the browser window also shows an excerpt of the formalized knowledge base, where first the sex ("Geschlecht") of the patient is asked.

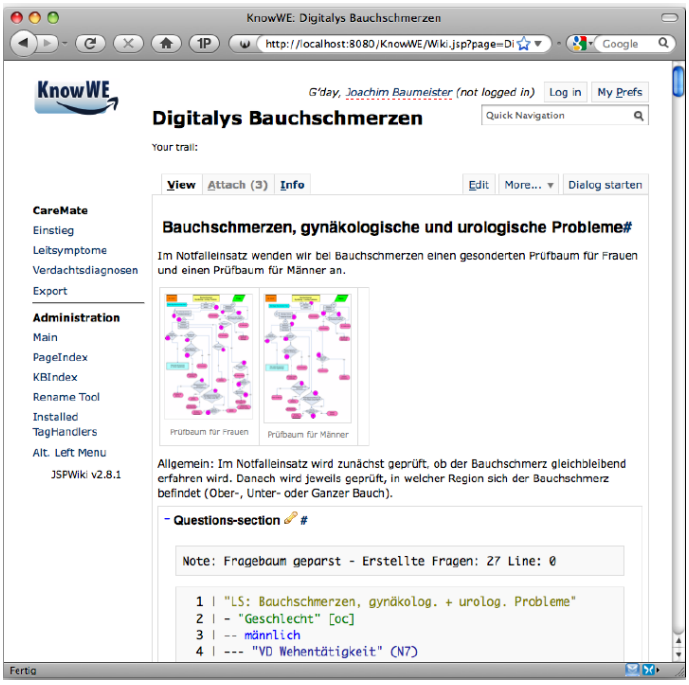


Fig. 6. The wiki page about the cardinal symptom "Bauchschmerzen" (stomach pain). (Screenshot in German language)

The CareMate system is commercially sold by the company Digitalys[10] as part of an equipment kit for medical rescue trucks.

4.5 Biodiversity with BIOLOG

The BIOLOG Europe project[11] aims at integrating socio-economic and landscape ecological research to study the effects of environmental change on managed ecosystems. To make the results of the research accessible for domain

[10] http://www.digitalys.de
[11] www.biolog-europe.org

specialists as well as diverse people with a different background, they decided to build a knowledge system (decision-support system). BIOLOG Wissen (BIOLOG Knowledge) is based on KnowWE and serves as a web-based application for the collaborative construction and use of the decision-support system in the domain of landscape diversity. It aims to integrate knowledge on causal dependencies of stakeholders, relevant statistical data, and multimedia content. In addition to the core formalization methods of the d3web-plugin, we introduced some domain specific features: One major challenge for the researchers in this domain is to find related work about similar studies. For this reason the research community (of ecology) has defined an extensive XML-schema for the description of meta-data about ecological work called *EML* (Ecological Meta-Data Language[12]). We defined a sub-language of EML which is suited to support capturing the relevant meta-data in this project. The research results and examinations of the different BIOLOG sub-projects are provided in EML and are entered into the wiki. Then, the EML data sets are visualized and can be accessed through a (semantic) search interface. Figure 7 shows the BIOLOG wiki depicting (part of) the visualization of an EML data set that describes work about *perception and appreciation of biodiversity* ("Wahrnehmung und Wertschätzung Biodiversität").

Fig. 7. Visualization of the EML data set about *perception and appreciation of biodiversity* (Screenshot in German language).

As the domain specialists are used to model concept hierarchies in the mind-mapping tool *FreeMap*, we integrated support for the FreeMap XML format in the wiki. Thus, a hierarchy can be created or modified externally with FreeMap and then pasted into the wiki to be translated into OWL-concepts.

[12] http://knb.ecoinformatics.org/software/eml/eml-2.0.1/index.html

To manage the publications of the project efficiently, we integrated support of BibTeX data. The wiki serves as a bibliography data base for the publications that have been created within the project scope.

4.6 Ancient History with HermesWiki

The HermesWiki [8] is a semantic wiki in the domain of Ancient Greek History. It is developed in cooperation the Department of Ancient Greek History of the University of Würzburg, Germany. Even though the HermesWiki does not develop a decision support system the meta-engineering approach has been applied successfully on both levels — conceptual and technical.

- **Conceptual Level** As the project at the beginning used a regular (non-semantic) wiki, at first the knowledge acquisition process clearly was free from any constraints due to knowledge formalization. After it was clear how the domain experts structured the content in the wiki in a natural way, we began to integrate formalization mechanisms tailored to the content already given in the wiki and to the workflow of the contributors. For example, we discovered that often multiple (related) time events were described on one page and defined a markup allowing to formalize a text paragraph as a time event by adding a title, a time stamp, and references to (historical) sources.
- **Technical Level** The HermesWiki has been implemented as a plugin for KnowWE reusing as much of the provided core components as possible. While some standard markups (e.g., annotation of pages being instance of some class) could be reused others had to be added (e.g., for time events or locations). Further, the dash-tree markup could be reused in different ways to define hierarchical structures. While the dash-tree parser from the core is used to parse the tree structure, the plugin only needs to specify how to process the (dash-tree) nodes during the compile process with respect to their father or children nodes.

Further details about the formalization methods of the HermesWiki can be found in Reutelshoefer et al. [8]. There, also the use cases for the formalized knowledge in the context of e-learning are described.

5 Conclusion

In this paper we introduced the idea of meta-engineering as an alternative approach to using 'out of the box' systems and building new ones from scratch. We motivated how the meta-engineering approach can help to bridge the two worlds of knowledge engineers and domain specialists and catalyzes the creation of a project tailored knowledge acquisition tool. Semantic Wikis are the appropriate technical platform to implement that approach, as a wiki builds a flexible basis and is customizable to a wide range of knowledge acquisition scenarios. They further allow for initial design phases, where specification and knowledge

acquisition can run in parallel. We discussed how the Semantic Wiki KnowWE supports the meta-engineering idea and we reported about several projects from different domains where the method proved to be helpful for the overall knowledge acquisition efforts. The introduced meta-engineering approach in principle also can be applied by the use of other Semantic Wiki systems, which are also designed with component based extension mechanisms, like for example Semantic MediaWiki [10] and KiWi [11]. However, no other systems known provide components for explicitly building intelligent decision-support systems. We still need to gather more experiences on how to determine the most appropriate wiki-based formalization architecture for a given project. Further, we will improve the technical infrastructure of KnowWE to allow meta-engineering to be applied with even lower implementation efforts.

References

1. Noy, N.F., Sintek, M., Decker, S., Crubezy, M., Fergerson, R.W., Musen, M.A.: Creating Semantic Web contents with Protege-2000. IEEE Intelligent Systems **16**(2) (2001) 60–71
2. Spinellis, D.: Notable Design Patterns for Domain Specific Languages. Journal of Systems and Software **56**(1) (February 2001) 91–99
3. Karsai, G., Krahn, H., Pinkernell, C., Rumpe, B., Schneider, M., Völkel, S.: Design guidelines for domain specific languages. In Rossi, M., Sprinkle, J., Gray, J., Tolvanen, J.P., eds.: Proceedings of the 9th OOPSLA Workshop on Domain-Specific Modeling (DSM09). (2009) 7–13
4. Baumeister, J., Reutelshoefer, J., Puppe, F.: Markups for Knowledge Wikis. In: SAAKM'07: Proceedings of the Semantic Authoring, Annotation and Knowledge Markup Workshop, Whistler, Canada (2007) 7–14
5. Baumeister, J., Reutelshoefer, J., Puppe, F.: KnowWE: A Semantic Wiki for Knowledge Engineering. Applied Intelligence (2010)
6. Reutelshoefer, J., Baumeister, J., Puppe, F.: A Data Structure for the Refactoring of Multimodal Knowledge. In: 5th Workshop on Knowledge Engineering and Software Engineering (KESE). CEUR workshop preceedings, Paderborn, CEUR-WS.org (September 2009)
7. Puppe, F.: Knowledge Formalization Patterns. In: Proceedings of PKAW 2000, Sydney Australia. (2000)
8. Reutelshoefer, J., Lemmerich, F., Baumeister, J., Wintjes, J., Haas, L.: Taking OWL to Athens – Semantic Web technology takes Ancient Greek history to students. In: ESWC'10: Proceedings of the 7th Extended Semantic Web Conference, Springer (2010)
9. Hatko, R., Belli, V., Baumeister, J.: Modelling Diagnostic Flows in Wikis. In: LWA-2009 (Special Track on Knowledge Management), Darmstadt (2009)
10. Krötzsch, M., Vrandecic, D., Völkel, M.: Semantic MediaWiki. In: The Semantic Web - ISWC 2006. Volume 4273 of Lecture Notes in Computer Science., Heidelberg, DE, Springer (2006) 935–942
11. Schaffert, S., Eder, J., Grünwald, S., Kurz, T., Radulescu, M.: KiWi - A Platform for Semantic Social Software (Demonstration). In: ESWC'09: The Semantic Web: Research and Applications, Proceedings of the 6th European Semantic Web Conference, Heraklion, Greece (June 2009) 888–892

Access and Annotation of Archaeological Corpus via a Semantic Wiki

Éric Leclercq and Marinette Savonnet

University of Burgundy
Le2I Laboratory - UMR 5158
B.P. 47 870, 21078 Dijon Cedex - France
Firstname.Lastname@u-bourgogne.fr

Abstract. Semantic wikis have shown their ability to allow knowledge management and collaborative authoring. They are particularly appropriate for scientific collaboration. This paper details the main concepts and the architecture of WikiBridge, a semantic wiki, and its application in the archaelogical domain. Archaeologists primarily have a document-centric work. Adding meta-information in the form of annotations has proved to be useful to enhance search. WikiBridge combines models and ontologies to increase data consistency within the wiki. Moreover, it allows several types of annotations: simple annotations, n-ary relations and recursive annotations. The consistency of these annotations is checked synchronously by using structural constraints and or asynchronously by using domain constraints.

1 Introduction

Document analysis is crucial to archaeologists when trying to understand the evolution of patrimonial buildings and sites. Documentary sources provide partial evidences from which researchers will infer possible scenarios on how a building may have been transformed through the ages. The aim of the CARE project (Corpus Architecturae Religiosae Europeae - IV–X saec.) is the constitution of an integrated corpus of the French Christian buildings dated from the 4th to the beginning of the 11th century. It aims at facilitating work of comparisons, exchanges and discussions with numerous foreign researchers and specialists. The project has been launched in France on January 1st, 2008 after acceptance of the French National Agency for Research and will last 4 years (2008-2011). More than sixty researchers from about twenty universities, diverse research institutions and heritage management institutions are working on. Various categories of staffs are involved: field archaeologists, historians, art historians, draftsmen, topographers, PhD students, etc. They are collecting and analyzing data concerning approximately 2700 monuments. The corpus of multimedia documents (including texts, maps, and photographies) concerning every known building will be gradually published in the form of classic books.

The request of a Web 3.0 application with a collaborative component and the need of document management led us to choose a solution based on a wiki

rather than a database. A prototype is available at `http://care.u-bourgogne.fr`. Despite the power of wiki (free input, rich user-interface, traceability, bi-directional links between pages, etc.), it is difficult to answer a specific query because of the purely textual information stored. Consequently, an approach which can provide a semantic annotation of the content is necessary. In addition, requirements for interoperability and data exchange must be taken into account since the design phase of the application. The semantic web thereby provides such kind of solutions by increasing the expressiveness of data representation, and by allowing reasoning tools and semantic search.

The computer application part of the project has started in September 2008, a prototype has been held with MediaWiki and Semantic MediaWiki through May to July 2009. After this prototyping phase we notice that some functionalities are missing in Semantic MediaWiki. For example n-ary relations are not fully supported and the scope of a tag is generally a document. As in Semantic MediaWiki, annotation can be enhanced as the knowledge evolves. In most of semantic wiki approaches, subjects of annotation are the whole document, we propose a recursive annotation model to cope with different levels of knowledge granularity as well as extension of domains. In [8], authors propose an equivalent representation between OWL concepts and Semantic MediaWiki constructs. WikiBridge approach allows to annotate an element with different annotations in several parts of documents. This functionality can be used to highlight a specific object described in a document. In [7], authors provide facilities to ensure the content quality of a wiki, including constraint and auto-epistemic operators. They introduce semantic checking with three kinds of constraints that are mostly structural: 1) domain and range; 2) concept cardinality; and 3) property cardinality. In WikiBridge, structural constraints checking is included in the annotation process while domain dependent constraints are checked asynchronously.

The rest of the paper is organized as follow: Section 2 describes our architecture through the physical and logical structure, the semantic layer, the information access layer. Section 3 concludes the paper.

2 WikiBridge's architecture

Our proposal is to use MediaWiki to develop a numerical corpus by integration of individual contributions. We have extended MediaWiki with some DBMS capabilities and semantic tools: form based acquisition interface, annotations, query engine.

2.1 Document Structuring

The archaeologists' work is focused on documents: documentary sources and documents of excavations are used to analysis of buildings; in result paper forms are produced. Moreover, document exchange, information retrieval and

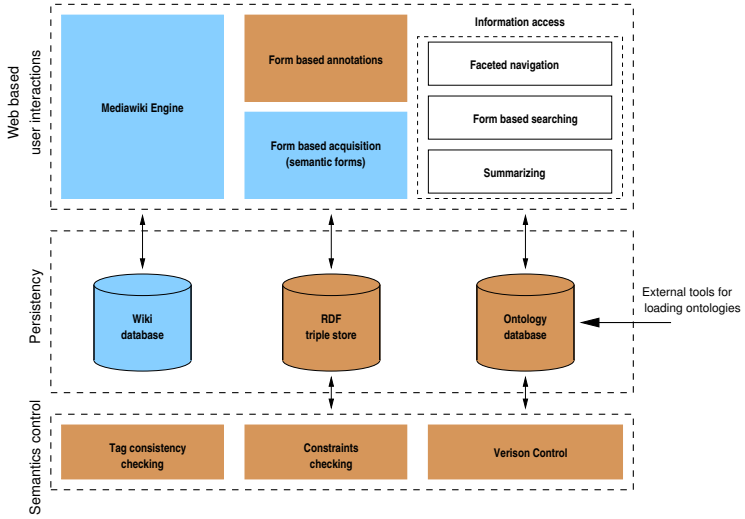


Fig. 1. WikiBridge's Architecture

integration are uses of these various documents. The multitude of purposes and the diversity of document content types led to different structuring needs. Standards such as Open Document Architecture and SGML (Standard Generalized Markup Language) consider that document has at least two structures of representation.

The physical structure defines the document presentation. This structure consists of physical elements such as style sheets (CSS, XSLT).

The logical structure defines an organization (relationship of composition, sequence) of information contained in the document. This organization represents the different parts of the document. It is composed of titles, chapters, paragraphs, notes, figures, etc. Organization of a document in the CARE corpus is as follows: topography, documentary sources, a succession of states describing the evolution of architectural building. In each state, plan of building with concepts of space, architectural elements and function are known from elements of relative dating such as construction techniques, building materials, sepulchers etc. This logical structure can not structure the knowledge and therefore does not allow easy information access.

The logical structure of the document could be stored in a database with attributes of type LONG, but a specific tool must be developed to display, to edit the different documents and their structure. Wiki is a suitable tool for representing these two structures.

The semantic structure has been introduced by other authors [3]. It represents the information itself i.e. the meaning of document content. The semantic structure describes information that a user or an agent asks when searching.

It is superimposed on the document and allows to manipulate the rules and not chapters or paragraphs.

2.2 Physical and logical structure layers

The physical structure is covered by MediaWiki and the logical structure is managed by Semantic Forms extension[1] for MediaWiki. Corresponding modules are described in light grey in figure 1. Each part of the paper document – a word file – (figure 2.a) is represented by a model (figure 2.b), models can be composed. A model is defined by using a mini-scripting language and forms are created on-the-fly on the basis of models. Two types of acquisition form have been created: a form for entering a record corresponding to atomic building and a form corresponding to a group of buildings. Some specific fields (select lists) and free text based fields are proposed. For instance, they are respectively used for selecting administrative regions of a building and describing liturgical installations in a building. Finally, a non-expert in archeology can easily feed the wiki (figure 2.c), by copying and pasting, from paper forms already made by archaeologists. Results are stored in the wiki database.

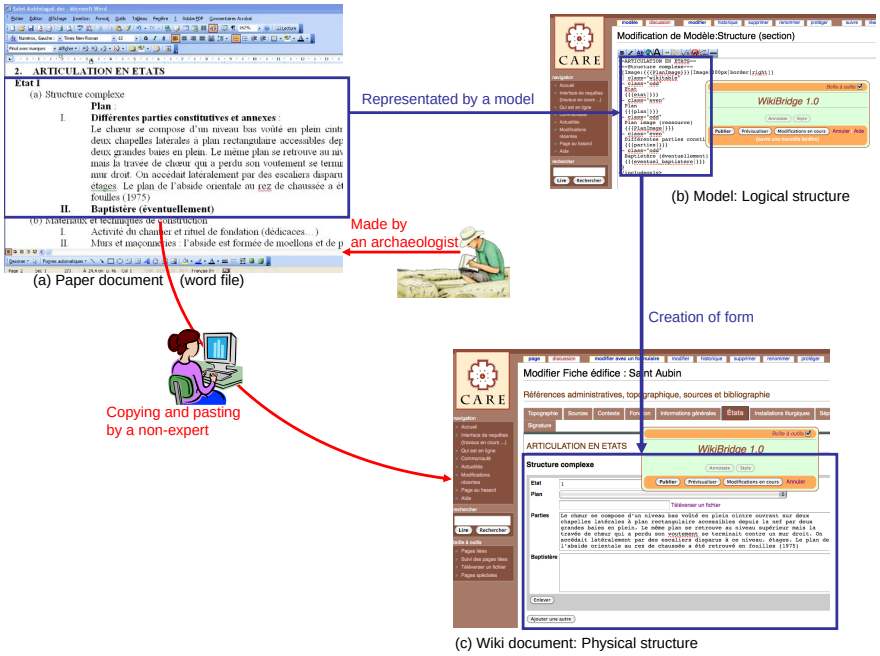


Fig. 2. Acquisition form and document model

[1] http://www.mediawiki.org/wiki/Extension:Semantic_Forms

2.3 Semantic layer

To improve quality of search, we expanded MediaWiki with semantic components (medium grey box in figure 1). Annotations, made by experts, are guaranteed by a domain ontology. Experts directly enter and modify annotations through an extension of the wiki's editing interface (figure 3) which relies on the form based annotation component. We restrict access to ontological knowledge management to a predefined set of Wiki users: we argue that implementing such functionality without adequate process-level support might have uncontrolled consequences on the operation of the overall wiki system. Knowledge engineers interacting with archaeologists create the domain ontology with standard tools like Protégé. The scope of domain ontologies includes concepts and relations of thematic area. Specific extensions of domain ontologies are defined in the context of a distinct usage of the more general knowledge model [4]. CIDOC Conceptual Reference Model [2] [2] is a domain ontology intended to facilitate the integration, mediation and interchange of heterogeneous cultural heritage information. We have made a specific extension of CIDOC ontology for the European Christian buildings. It consists of:

- found objects : type of buildings, architectural elements (e.g. nave); liturgical installations (e.g. altar), wall structures and pavements . . .
- religious aspects of these objects: function, consecration;
- spatial aspects: relative position of an object with another;
- architectural evolution of objects: creation, destruction and modification by adding or deleting element.

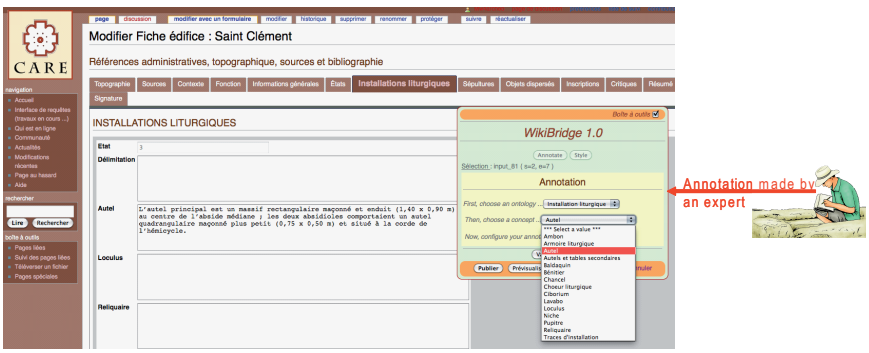


Fig. 3. Annotation Interface

[2] http://cidoc.ics.forth.gr/

Persistency Two persistency levels have been distinguished:

- A persistence level related to knowledge which includes ontology and annotations. We explicitly store in a relational database the conceptual model defining the structure of the domain ontology (Figure 4). Ontology is loaded from Protégé by a specific program. As a result, annotations are stored in RDF data in the RDBMS.
- A persistence level related to document structure is realized by MediaWiki with the Semantic Forms extension.

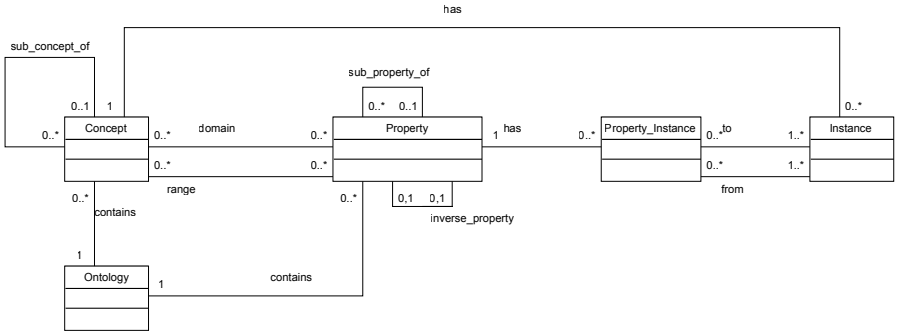


Fig. 4. Database schema for ontology

Annotations and consistency checking Two types of annotations have been identified:

- Simple annotation allows to tag a subject by describing some of its properties by attributes values (literal) couples. These kind of annotations can be compared to a restriction on attribute's domain in the database context. Theses annotations are mostly related to the ABox level.
- Complex annotation references TBox and ABox levels:
  - n-ary relation allows to map a subject with two or more values and references to other elements (subjects). In this case, some values properties reference another subject. For example we can annotate an altar with its dimension, its building material, its location in the nave. The nave is detailed in another part of the document.
  - recursive annotation allows to explain or clarify an attribute by a sub-annotation which is a simple or a complex annotation.

Moreover, annotations related to the same subject can be expressed in different parts of a document or in different documents. We propose a mechanism

to merge annotations and to visualize all the annotations related to one subject in the annotation interface.

In order to implement our annotation mechanism, we choose to use the model of semantic values proposed by Sciore et al. [5] for mediation of relational databases. They define recursively semantic values by the association of a context to a simple value. A context is a set of elements which are assignment of a semantic value to a property. We extend this model by allowing values to be references to other elements (part of documents, subjects). For the aforementioned altar example, the annotations are:

$$1.3(dimension = "width", unit = "m") \quad (1)$$
$$0.95(dimension = "height", unit = "m") \quad (2)$$
$$2.4(dimension = "length", unit = "m") \quad (3)$$
$$marble(buildingMaterial = "stone") \quad (4)$$
$$\#nave143(spatialRelation = "contained"(spatialPosition = "center")) \quad (5)$$

Annotations (1), (2), (3) should be merged but the semantic values model treats them as separated annotations. We can introduce an intermediary annotation such as 3D to allow combination of multiples semantic values. The value is then a specific attribute of the annotation.

$$\begin{aligned} dimension = "3D"(&dimensionY = "width"(unit = "m", value = "1.3"),\\ &dimensionZ = "height"(unit = "m", value = "0.95"),\\ &dimensionX = "length"(unit = "m", value = "2.4")) \end{aligned}$$

Annotation modeling using semantic values allows automatic conversion of units (for example between meters and inches). The same type conversion can be used for dates from centuries to values interval. Conversion can be used in query processing or for multi-lingual support.

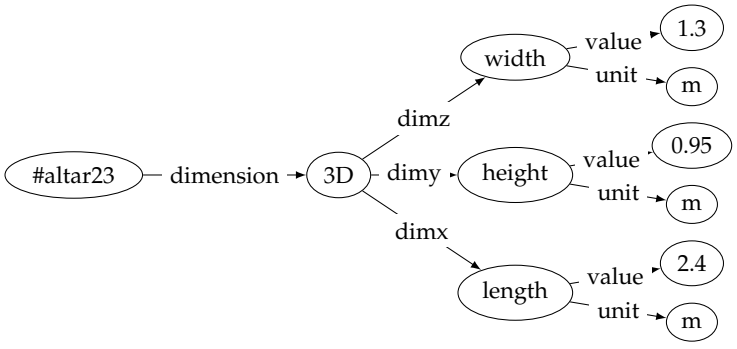


Fig. 5. RDF like transformation of semantic values

In WikiBridge, semantic values are reified in a list of atomic annotations i.e. couples (property, value), related to an object (subject), that are stored as triples in the database. An identifier is given to each atomic annotation allowing recursive semantic values. Annotation tuples can be translated in RDF and displayed as graph (figure 5).

Annotations are defined by users through a wizard that controls two kinds of constraints: 1) Domain values of properties using ABox capabilities; and 2) Structural consistency of properties using TBox capabilities (for instance, a cathedral can have a nave but cannot have an atrium).

This two kinds of constraints can be checked using the ontology structure in OWL format. Nevertheless, some domain dependent constraint cannot be embedded in the structure. For example "a building cannot be dedicated to a saint before is death date" is represented by the following rule:

`isConsecrated(?b,?p)` $\leftarrow$ `hasConstructionDate(?b,?d1)` $\wedge$ `hasDateDead(?p,?d2)` $\wedge$ `d1` $\geq$ `d2`.

OWL is mainly based on description logics [1] (DL). Some features of DL make it difficult to use for validating data annotations through integrity constraints (IC): 1) OWL-DL works in open world assumption; 2) OWL does not use the unique name assumption. Finding inconsistent annotations require to evaluate OWL rules in a closed world assumption to detect violation. The domain dependent constraints are checked when users validate an annotation while domain values and structural properties are checked when users build the annotation through wizard (figure 3). Three approaches are described in [6] : 1) skolemisation-based semantics, some constraints are tagged as IC; 2) ruled-based semantics based on interaction with logic programming that provides negation as failure under the closed world assumption and 3) query-based semantics that relies on boolean epistemic queries for expressing constraints.

In order to implement constraints two solutions have been tested: 1) translation of constraints in a programming language such as procedural SQL or PHP and 2) use of a reasoner and a set of constraints stored in a file. The second solution was chosen because it allows to define and to add dynamically new constraints as knowledge evolve.

2.4 Information access layer

Information access layer has been built with taking into account some features about users. We have thus identified a usage typology in accordance to 1) kind of usage: reader, investigation, clarification; 2) knowledge degree of the domain: domain specialists like archaeologist researchers and non specialists. On this basis, we can distinguish: 1) general public with a general knowledge of the area who wants to find information on the known elements; 2) experts understanding meaning of annotations who need access to detailed information; and 3) researchers who need to make analysis i.e. cross-checking data from multiple articles and make emergence of new knowledge.

To handle these different types of users, we offer three types of queries:

1. faceted browsing allows users to explore by filtering available information through an ontology tree;
2. form based searching provides semantic search by filling in parameters associated with ontology concepts. Two types of interfaces (figure 6) for building semantic queries are developed: a wizard lets users to specify search parameters to engine and users can create query models that are then stored;
3. aggregate view for each article as factbox.

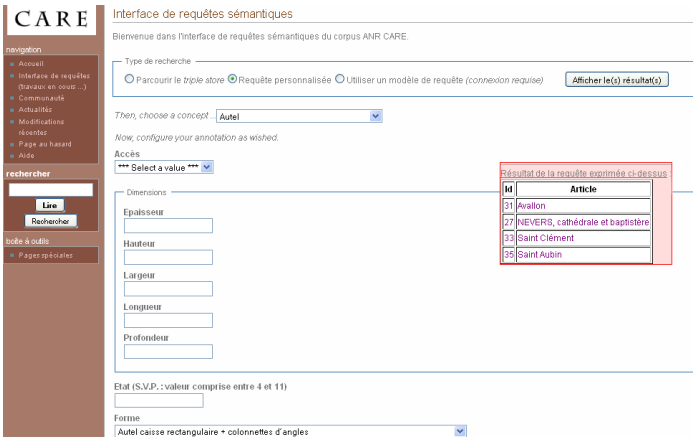


Fig. 6. Query interface

Three kinds of results can be displayed: 1) results can appear in a list containing links to articles, at the right annotation place, so where the information is given; 2) user can then manually navigate through articles interlinked; and 3) users can select annotation to be displayed in the result. From this result, users can obtain the list of the articles in which have the same annotation. This third kind of display is a mix of result list and factbox and allows more sophisticated analysis.

3 Conclusion

A feasible combination of wiki and Semantic Web technologies should preserve the key advantages of both technologies: the simplicity of wiki systems as shared content authoring tool, and the power of Semantic Web technologies w.r.t. structuring and retrieving knowledge. In this article, we have demonstrated that flexibility and data quality required by scientific applications can be achieved by using wiki with semantic web technologies.

We use annotations to make links between logical layer and semantic layer. The semantics of annotation is guaranteed by an ontology including constraints which allows to describe accurately domain knowledge. Our dual approach allows to cope with evolution of knowledge by modifying the ontology and annotations dynamically without modifying database schema.

Actually, we only verify structural constraints in a synchronous mode when users annotate the document. The next version of WikiBridge will automate verification of integrity constraints by Pellet reasoning engine and annotations will be marked by an ontology version. Remain the problem of inter-ontologies version consistency.

Some geomaticians of the Social Sciences and Humanities Research Institute of Dijon will conduct specific spatial analysis by providing GIS tools from end of 2010. For thorough analysis, specialized tools (GeoMondrian[3] and PostGIS [4]) interconnected by Web Services will be proposed to specifically address the spatio-temporal aspect. For simple spatial analysis, OpenLayers[5] applications will be developed.

Acknowledgments This work is supported by the ANR (ANR-07-CORP-011).

References

1. Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
2. M. Doerr. The cidoc crm - an ontological approach to semantic interoperability of metadata. *AI Magazine*, 24:2003, 2003.
3. Line Poullet, Jean-Marie Pinon, and Sylvie Calabretto. Semantic Structuring Of Documents. *3rd Basque International Workshop on Information Technology (BIWIT)*, pages 118–124, 1997.
4. Sebastian Schaffert, Andreas Gruber, and Rupert Westenthaler. A semantic wiki for collaborative knowledge formation. In *In Semantics*, 2005.
5. Edward Sciore, Michael Siegel, and Arnon Rosenthal. Using semantic values to facilitate interoperability among heterogeneous information systems. *ACM Trans. Database Syst.*, 19(2):254–290, 1994.
6. Evren Sirin, Michael Smith, and Evan Wallace. Opening, Closing Worlds - On Integrity Constraints. In *OWLED*, 2008.
7. Denny Vrandecic. Towards automatic content quality checks in semantic wikis. In *Social Semantic Web: Where Web 2.0 Meets Web 3.0*, AAAI Spring Symposium 2009, Stanford, CA, march 2009. Springer.
8. Denny Vrandecic and Markus Krötzsch. Reusing ontological background knowledge in semantic wikis. In Max Völkel and Sebastian Schaffert, editors, *Workshop on Semantic Wikis*, volume 206 of *CEUR Workshop Proceedings*, 2006.

[3] `http://www.spatialytics.org/projects/geomondrian/`
[4] `http://postgis.refractions.net/`
[5] `http://openlayers.org/`

Collaborative Editing and Linking of Astronomy Vocabularies Using Semantic Mediawiki

Stuart Chalmers[1], Norman Gray[2], Iadh Ounis[1], and Alasdair Gray[3]

[1] Computing Science, University of Glasgow, Glasgow, UK
[2] Physics and Astronomy, University of Glasgow, Glasgow, UK
[3] School of Computer Science, Manchester University, Manchester, UK

Abstract. The International Virtual Observatory Alliance (IVOA) comprises 17 Virtual Observatory (VO) projects and facilitates the creation, coordination and collaboration of standards promoting the use and reuse of astronomical data archives. The Semantics working group in the IVOA has repurposed five existing vocabularies (modelled using SKOS), capturing concepts within specific areas of astronomy expertise and applications. A major task however, is to promote the uptake of these semantic representations within the Astronomy community, and further, to let astronomers model (and in turn create links from) their own custom vocabularies to use these existing definitions. In this paper we show how Semantic Mediawiki (SMW) can be used to support expert interaction in the lifecycle of vocabulary creation, linking, and maintenance.

1 Introduction

Astronomy as a discipline incorporates a broad range of topics and data analysis across the wavelength spectrum, from gamma-rays to radio waves, and a wide range of expertise from professional researchers to amateurs. Because of the collaborative nature of astronomy working groups and projects, and a culture where sharing data is the norm, there is a well-established need for consensus definitions describing data (mostly image and object catalogue data). To this end a number of standardised vocabularies have emerged, which are mostly, at present, focused on the search for and retrieval of resources, primarily data and journal articles.

Thus, multiple independent controlled vocabularies have evolved to meet the various terminological needs of these different sub-communities (Table 1). The most widely-known of these is the keyword list maintained jointly by the three main astronomy journals A&A, ApJ and MNRAS (these keywords are used to tag journal articles, so that most astronomers have a familiarity with this set), and the largest is a thesaurus developed by the International Astronomical Union (IAU) (with the IVOA starting work on an update, the IVOAT). Newer than both are the AVM vocabulary – a recent effort intended for use when tagging astronomy outreach images – and the UCD list, in increasingly wide use as a set of standardised database column headings[4]. For further discussion see [1].

[4] `http://www.ivoa.net/Documents/latest/Vocabularies.html`

Vocabulary	Original Publisher	Purpose	Number of Concepts
Journal Keywords	Journal publishers	Tagging articles to aid retrieval	311
Astronomy Visualization Metadata (AVM)	various	Tagging images for dissemination	208
The IAU Thesaurus (IAUT)	IAU	Library cataloguing	2551
The IVOA Thesaurus (IVOAT)	IVOA	Update of the IAU Thesaurus	2890
Universal Content Descriptors (UCD)	IVOA	Labelling data repository column headings	473

Table 1. Astronomy vocabularies

While the IVOA vocabularies have provided a basis for standardisation of experimental terminology, there remain a few problems:

- There are no standardised tools or methodology for creating custom experimental descriptions based on these vocabularies.
- Users may be familiar with specific IVOA vocabularies relating to their subdiscipline, but not others, meaning that their description cannot describe their data as fully as a searching colleague might require.
- Searching of user-defined vocabularies and data is limited to terminology in the IVOA vocabular(ies) used to define them. For instance, a user vocabulary described using the IVOAT thesaurus has no relation to searches using keywords from the IAUT thesaurus.

Recent work in the Explicator project[5] has laid the foundations for a solution to these problems, by representing the main IVOA vocabularies in SKOS, and exploiting SKOS relationships to help domain experts articulate cross-vocabulary links [2].

2 Current Vocabulary Building Tools

The Explicator project has developed a number of tools for the creation and use of SKOS astronomy vocabularies. The main entry point for searching and exploring terminology is the Web Vocabulary Explorer[6], built upon the Terrier Information Retrieval Platform [3] and providing an AJAX frontend for searching and browsing the astronomy vocabularies by entering a simple search string to find matching concepts. Fig. 1 (left) shows the search results for "star". The use of Terrier is important, in order to provide useful ranking of results: this vocabulary contains a large number of labels with common strings, so a naive search for "star" produces more than 600 concepts which have that string somewhere in their label, with the key concept 'Star' appearing uselessly far down the list. Using Terrier's ranking support, however, the appropriate concepts from

[5] `http://explicator.dcs.gla.ac.uk`
[6] `http://explicator.dcs.gla.ac.uk/WebVocabularyExplorer`

the three searched vocabularies appear at the beginning of this list. The explorer allows users to expand results and view details of concepts, such as alternate labels, available definitions and semantic relationships. Related concepts, both within a vocabulary and across vocabularies, can be explored by following links to broader, narrower, related, and equivalent concepts. Searches can be configured by selecting sets of vocabularies and mappings. This service is also available via XML-RPC, so that it can be embedded within other applications.

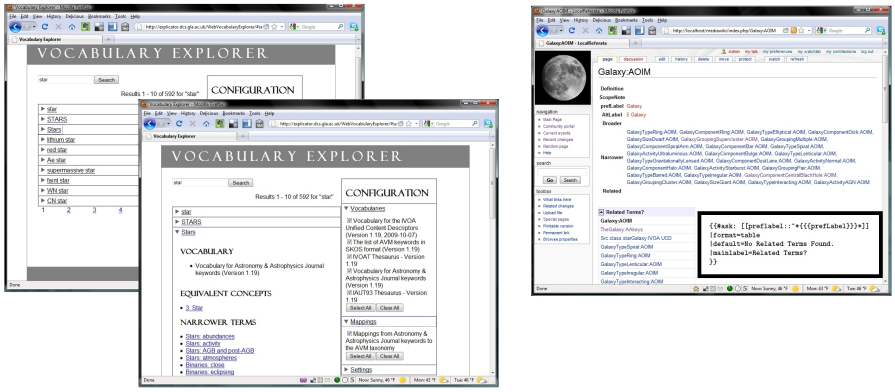


Fig. 1. The Web Vocabulary Explorer interface (left), and the inline search query and its use in the AOIM Galaxy definition (right)

To create links between the main vocabularies in Table 1 we have a Java mapping application providing a GUI interface to declare mappings between vocabularies that can then be integrated into the Web Vocabulary Explorer. The five vocabularies listed here were pre-existing ones, though not published as SKOS, and so were converted from their original formats as part of the process of developing [4]. The tool also allows the inclusion of automatically created RDF representations of databases, created using the D2RQ database to RDF mapping tool[7]. The other important source of ontology information within the VO is the IVOA's resource registry[8], which curates resource metadata using a standardised set of XML Schemas, which we have also converted to RDF Schemas using XSLT transformations.

Part of the point of the tool's search functionality is to help users find relevant concepts in multiple vocabularies, and to support them in articulating inter-vocabulary mappings. However we do not aim to do any automatic vocabulary alignment.

3 Semantic Mediawiki in the Vocabulary Lifecycle

While the astronomy community is in general technically adept, the immediate payoff from adopting the tools described in section 2 and converting to SKOS

[7] `http://www4.wiwiss.fu-berlin.de/bizer/d2rq/`
[8] `http://rofr.ivoa.net`

representations is not obvious (or apparent) enough to users to make this an attractive option (this is a general problem, also discussed in [5]). What is needed is a cohesive, familiar and easily understandable interface that integrates these tools in a way that allows the creation of SKOS-based experiment descriptions and vocabularies (based on and utilising current IVOA standard vocabularies) with minimal expenditure on learning the underlying semantic representations. To this end we have proposed a coherent vocabulary 'lifecycle' methodology (creation, collaborative editing, linking and searching/use) – see Fig 2. This uses SMW as a collaborative vocabulary building tool to create and edit vocabularies (1), *link* (1) these to existing IVOA vocabularies (2) and have them automatically *exported* to (3) and *imported* from (5) their corresponding SKOS representations (4) for use in the Web Vocabulary Explorer.

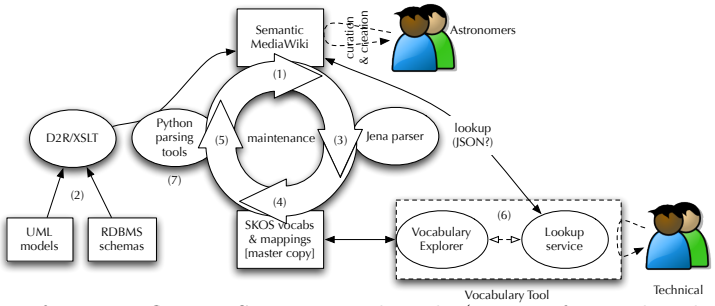


Fig. 2. Information flows in Semantic MediaWiki (see text for numbered notes)

To link SMW to our existing tools (6), we have developed a general set of python scripts (7), using pywikipediabot[9] and the rdflib[10] library to automate the uploading and parsing of our SKOS vocabularies into Wikipedia pages[11]. The SMW pages are based on a simple semantic form/template structure, parsed from the main SKOS vocabularies (4) and uploaded using the python bots. Similarly we use a Jena-based parser to parse the SMW OWL/RDF export (3) for a particular vocabulary and create the corresponding SKOS version for re-inclusion in the Web Vocabulary Explorer search.

This linking of the five main IVOA vocabularies into SMW pages means that we now have a base set of terms for users to begin using in their own experimental vocabularies. To help users find related terminology (e.g. for broader, narrower, or related matches in their SKOS terms) we use simple inline queries embedded in the main vocabulary term template to show (on each term's page) the possible related terminology. Fig. 1 (right) shows the inline query used in the main template of the vocabulary wiki pages and an example, the AOIM term 'Galaxy'. This shows the main definition (scopenote, prefLabel, altlabel, broader, narrower and related) and a table of the possible related terms (including TheGalaxy in the AAKeys vocabulary and the src.class.starGalaxy from the

[9] `http://meta.wikimedia.org/wiki/Pywikipedia`
[10] `http://www.rdflib.net/`
[11] We currently host this testbed at `http://vocabularies.referata.com`

UCD vocabulary) that may be linked to by the user as cross-vocabulary related terms.

4 Related and future work

There are other vocabulary development systems in existence, including the NeOn project's ontology editor[12], and its Cicero project, which is also based on SMW, and which supports an elaborate argumentation structure for collaborative ontology development (NeOn deliverable 2.3.1). On a similar theme is LexWiki[13], which is a platform for developing a biomedical vocabulary. The problem we are addressing, however, is *not* that of collaboratively creating a large ontology from scratch, but supporting the collaborative inter-relation of multiple existing vocabularies from various sources, with a community which is made more rather than less comfortable by having some of the underlying technology visible, and repurposable from user-written applications.

At present we are working on a mediawiki extension that will allow us to use the XML-RPC search from the Web Vocabulary Explorer to find related terms. This will use the Terrier search described above, to provide more accurate ranked searches for related terms, than is possible with the existing inline searches.

A key advantage, for us, of using a wiki-based solution is that it provides a good match to the expectations of the domain experts – they feel comfortable and in control when using it. Both the wiki and its embedded functionality must therefore evolve in tune with the user base, and an important strand of our future work on this project is to evaluate the provided functionality in use.

References

1. Gray, A., Gray, N., Ounis, I.: Vocabularies in the VO. In Bohlender, D., et al., eds.: Proc. Astronomical Data Analysis and Software Systems Conference (ADASS XVIII). Volume 411., Astronomical Society of the Pacific (2009) 179–182
2. Gray, A.J.G., Gray, N., Hall, C.W., Ounis, I.: Finding the right term: Retrieving and exploring semantic concepts in astronomical vocabularies. Information Processing and Management (2009). In press.
3. Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., Lioma, C.: Terrier: A high performance and scalable information retrieval platform. In: Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006), Seattle (Washington, USA), ACM (2006)
4. Gray, A.J.G., Gray, N., Hessman, F.V., Preite Martinez, A.: Vocabularies in the virtual observatory. IVOA Recommendation (2009) Available at: `http://www.ivoa.net/Documents/latest/Vocabularies.html`.
5. Gray, N., Linde, T., Andrews, K.: SKUA - retrofitting semantics. In Auer, S., et al., eds.: Proc. 5th Workshop on Scripting and Development for the Semantic Web at ESWC 2009, Heraklion, Greece. Volume 449 of CEUR Workshop Proceedings ISSN 1613-0073. (2009)

[12] `http://www.neon-project.org/`
[13] `http://informatics.mayo.edu/vkcdemo/lexwiki1/`

A Wiki-Oriented On-line Dictionary for Human and Social Sciences

Lydia Khelifa[1,2], Nadira Lammari[1], Hammou Fadili[3], Jacky Akoka[1]

[1] Conservatoire National des Arts et Métiers de Paris, 292 rue St Martin, 75141 Paris Cedex 03, France

[2] Ecole Nationale Supérieur d'Informatique d'Alger (ex INI), BP 68M Oued Smar, 16309, El Harrach, Alger, Algérie

[3] Fondation Maison des Sciences Humaine et Sociales de Paris, 54 boulevard Raspail, 75270 Paris Cedex 06, France

khelifalydia@gmail.com, {lammari, akoka}@cnam.fr, fadili@msh-paris.fr

Abstract. The aim of this paper is to contribute to the construction of a human and social sciences (HSS) on-line dictionary. The latter is Wiki-oriented. It takes into account the multicultural aspect of the HSS as well as the ISO 1951 international standard. This standard has been defined to harmonize the presentation of specialized/general and multilingual/monolingual dictionaries into a generic structure independent of the publishing media. The proposed Wiktionary will allow HSS researchers to exchange and to share their knowledge regardless of their geographical locations of work and/or of residence. After the conceptual description of this dictionary and the presentation of the mapping rules to Wiki semantic concepts, the paper will present an overview of the prototype that has been developed.

Keywords: Semantic Wiki, Human and Social Sciences, Multicultural Wiktionary.

1 Introduction

While social science studies human societies, human sciences deal with human groups and individuals, their history, their cultures, their accomplishments and their individual and social behaviors. Both social and human sciences (HSS) encompass heterogeneous disciplines like anthropology, sociology, economics, ethnology, geography, history, political science, archeology, linguistics science and religion science. They play a key role in understanding and interpreting the economic, cultural and social context of populations. The evolution of the research in this area inevitably involves knowledge exchange and sharing between researchers.

To promote exchanges between Maghrebi countries and France in the HSS area, the FMSH[1], with the collaboration of partners from France and Maghrebi countries[2], have defined a project aiming at the construction of a multicultural and multilingual content. This project will allow exchanges between Maghrebi and French researchers. It will also allow the sharing of knowledge related to the two cultures and to the two societies. In this project it has been decided to first construct an on-line dictionary for the HSS. This dictionary does not exist at the present time. It must respect the ISO 1951 standard [1], be extensible to many languages and exploit the Wiki technology. One of the reasons motivating the FMSH choice for the Wiki technology is the ease and the speed of defining, structuring and describing all types of data, according to different schema, using the WikiML (Wiki Markup Language). Moreover, the evolution management of this kind of application (dictionary application), generally difficult, is facilitated thanks to the Wiki platform, especially when the changes concern only the structure of the content.

The WikiMedia foundation supplies a Wiktionary. The latter is an open and universal dictionary. It is free for development and allows, authorized people to easily and rapidly edit, publish and maintain on-line content through collaborative processes that mutualize human skills. It also offers a complete versioning system and can alert anyone interested in particular themes when any content creation, modification or deletion, corresponding to his favorite themes, is performed. However, its current schema doesn't fulfill all the HSS dictionary functional requirements such as the search by context, hence, the idea to extend it.

The rest of the paper is organized as follows. Section 2 describes the peculiarities of the HSS on-line dictionary. Section 3 is dedicated to related works. Section 4 focuses on the conceptual modeling of this dictionary. The prototype is presented in Section 5. Section 6 concludes the paper and presents some perspectives.

2 The HSS dictionary description

To promote exchanges between of the two banks of the Mediterranean Sea in the HSS field, the development of a multilingual and multicultural e-dictionary has been initiated by the FMSH. This dictionary should, at first, contain the main HSS words used in France and in the Maghrebi countries, specify their use by both societies and supply their translation from one language to another one. This dictionary will be extended to all the languages of the Mediterranean countries later on.

The design of the on-line HSS dictionary must take into account the facts that:

• an entry A_k in a source language can have several meanings and therefore several translations B_1, …, B_m in the target language. Moreover, this same entry A_k can be defined with several components A_1, …, A_i of the dictionary schema (synonym, antonym, related nouns, pronunciation, etymology, etc). Each of these components could be an entry in the source language and could, therefore, have several meanings in the source language and several translations in the target language (Fig. 1). Let us

1 One of the acronym of the "La Fondation Maison des Sciences de l'Homme" (FMSH), http://www.fsp.maghreb-france.msh-paris.fr/

2 The partners are: FMSH, Cnam of Paris, ESI of Algiers.

note that any source language is also a target language. It depends on the required translation. Moreover, it may occur that an entry in the source language may not have a correspondent entry into a target language.

• the meaning assigned to a HSS dictionary entry depends on the context of the definition of this entry. The latter is described by a finite and known set of contextual parameters that vary from one discipline to another one. Among these parameters we can mention geographic and temporal parameters for sociology.

• the components used for the description of an entry are those of the ISO 1951 standard [1].

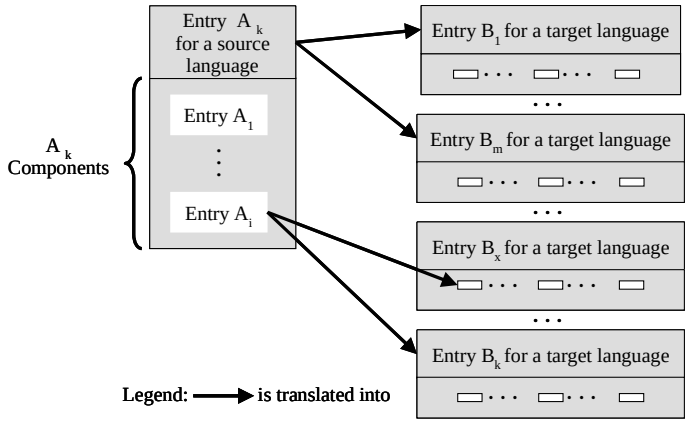


Fig. 1. The HSS dictionary schema extract.

Beside the constraints related to the description of the HSS one-line dictionary, its development, were in the specification document, conditioned by the exploitation of the Wiki technology for its advantages including the ease of construction and maintenance of collaborative contents by non expert users (users who are not specialists in computer science). Finally, it must allow the search by context.

3 Related Works

There are several projects for the construction of specialized on-line dictionaries. Among them, we can mention the PAPILLON project [2], the DHYDRO project [3], the JMdict/EDICT project [4] and the SAIKAM project [5]. In the PAPILLON project, the paradigm of the Linux collaborative building has been applied to the collaborative edition of definitions. It offers, among possible search criteria, the retrieving of a word according to its contextual reading. In the DHYDRO project a terminological and multilingual space specialized for the hydrographic domain has been built. JMdict/EDICT proposes a remote edition tool for a multilingual terminological database. SAIKAM is an on-line dictionary. It aims at the creation of new Thai words for Japanese ones.

Let us note also that the Semantic Web Deployment Working Group, part of the W3C Semantic Web Activity, recommended, since august 2009, the SKOS (Simple

Knowledge Organization System) model for the description of thesaurus, taxonomies or any other controlled vocabularies [11]. SKOS is based on the RDFS language.

However, none of the projects cited above uses the Wiki technology. This led us to explore the possibility to exploit the current Wiktionary project of WIKIMEDIA foundation. The latter proposes a Wiktionary per language. Some of them, like the Arabic Wiktionary, lacks structure. The other ones don't have similar structure (eg. English and French Wiktionary [13] [14]). For example, the French Wiktionary is organized into articles [13]. Each article is used to describe a word. It gathers:

- a main section for the description of the word in the language associated to the Wiktionary,
- zero or more other language sections, each for a language different from that of the Wiktionary,
- a categorization section that classifies the word into one or more categories from those listed
- and finally, a section that allows to establish links between the article and other ones in the others Wiktionaries. These links are oriented to articles having the same title. They don't concern their translations.

The main section proposes:

- a mandatory set of basic description elements: etymology, one or more sections for the type of word (i.e its spelling variants, its abbreviation, its derived words, its synonyms, its hyponyms, its translations, etc.)
- and a set of optional elements: pronunciations, anagrams, and a section «to see also» that gather the links related to the article and a reference section that gives the references used during the edition of the article.

The sections dedicated to languages are similar to the main section except that it doesn't contain some sections like the one needed for translation or for hyponymy.

The description possibilities supplied by the current Wiktionary project don't meet the HSS dictionary specificities. On one hand, it lacks an automatic management of correspondences that allows managing the complexity of referrals between the source language and the target language. It is possible to use the current Wiktionary to change an entry regardless of other entries to which it is linked. In other words, it is possible to add in a Wiktionary dedicated to one language A, a translation of a word into a language B without impacting the change in a Wiktionary dedicated to language B. Moreover, links between Wikis, in the Wiktionary, can be established only between articles having the same name. This means that we can not link two words, such that the first one is the translation of the second one, if the two words are not in the same Wiki. On the other hand, the current Wiktionary project does not allow contextual search of the meaning of words. This functionality is very important in HSS field and must be fulfilled by the HSS Wiktionary application.

Another version of a Wiktionary exists: OmegaWiki [12]. It is based on an extension of MediaWiki. OmegaWiki unlike the current Wiktionary project gathers in one space all the Wiktionaries. It overcomes the drawback of the current Wiktionary concerning the impact of changes from one Wiktionary to another. Finally, OmegaWiki, at the present time, can be used only for search and it does not supply a contextual search for the word meaning.

4 The HSS Wiktionary Design Approach

As mentioned in the previous section, a HSS on-line dictionary entry could have many descriptions. Each of these descriptions can be valid for a given context described by a set of contextual parameters like geographic and temporal parameters. Moreover, each description must respect the ISO 1951 standard. The design of the HSS on-line dictionary is, therefore, based on the correspondence between an entry and its contexts of definition in a source language and an entry and its contexts of definition in the target language. This correspondence is performed according to a schema that could contain the definition of the entry, the synonyms, the antonyms, the related words, the pronunciation, the spelling, etc.

The conceptual description of such dictionary could be represented using an UML class diagram. Figure 2 is an extract of this conceptual model. This model shows that the description of an HSS dictionary entry (word) in a given language is obtained by gathering the variants of this description. Each variant of a description corresponds to a context defined by the concerned discipline, the set of context elements which are context parameters values. Each discipline has its own context parameters. Each entry described with a given variant of a description could have a synonym related to this variant of description.

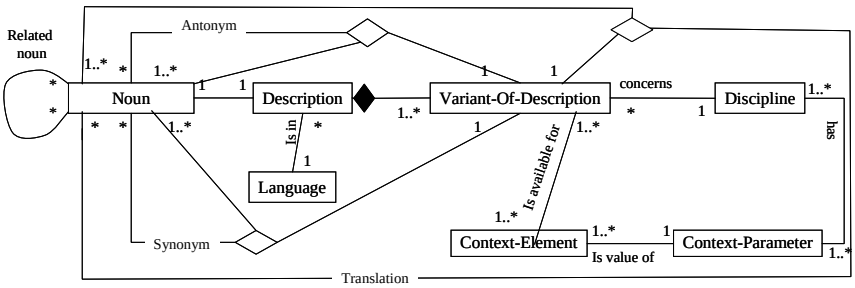


Fig. 2. An extract of the HSS dictionary conceptual model.

The use of the Wiki technology for HSS on-line dictionary constitutes, in the project, a technical constraint that we must comply to. To date, there are several Wikis. WikiNi, Wiclear, DokuWiki, MediaWiki and semantic Wikis are some examples of existing ones. Semantic Wikis such as KawaWiki [6], IkeWiki [7], SweetWiki [8], Kaukolu [9] and Semantic MediaWiki [10] are semantic web extensions of the Wikis. KawaWiki allows the creation of Wiki pages using RDF templates and their querying by means of SPARQL. IkeWiki is a tool for formalized and collaborative building of content. It offers the possibility to annotate the links and the possibility of reasoning. SweetWiki semantically annotates Wiki resources. It supports the social tagging, uses ontologies for the structures of the Wiki and offers a WYSIWYG editor. Kaukolu is a semantic Wiki based on JSPWiki. It allows the annotation, creation and display of pages. It also replaces Unified Resource Identifiers by alias to allow creation of new pages. Semantic MediaWiki is an extension of MediaWiki. It inherits the advantages of MediaWiki such as easiness to use, editing collaborative documents (minimum of technical prerequisite), and its evolution. It

also allows annotating Wiki pages, their content and the links between them. Moreover, for navigation purposes, the semantic Wikis, and in general the Wikis, allow the intensive use of hyperlinks. Therefore, a future user of the application can get a global view of a page and can then have a zoom (a detail) of the part of the content he (or she) is interested in.

Our study of the state of the art and its confrontation with HSS on-line dictionary peculiarities, allows us to retain, for its realization, the Semantic MediaWiki technology.

The concepts associated with a Semantic MediaWiki are represented in the metamodel of Figure 3. A Semantic MediaWiki, as shown in Figure 3, is a set of Wiki pages that can be annotated. A Wiki page can be related to another one through external hyperlinks. Hyperlinks can also be used within a page. They can also be annotated.

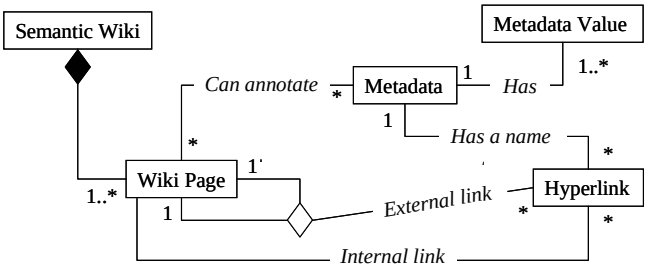


Fig. 3. The Semantic MediaWiki metamodel.

The mapping between the concepts of our on-line dictionary (Fig. 2) and the concepts of the Semantic MediaWiki (Fig. 3) is presented in Table 1.

Table 1. The mapping between HSS Wiktionary concepts and Semantic MediaWiki concepts.

HSS on-line dictionary concepts	Semantic MediaWiki concepts
Description/Variant of description	Wiki page
Context element	Metadata value of a context parameter
Language/ Discipline / Context parameter	Metadata
Antonym/Synonym/Related noun/Translation	Hyperlink

This table shows that the different descriptions of an entry (variant of a description) are mapped, in a Semantic MediaWiki, to Wiki pages. This is the same for the complete description of an entry. The concepts “Language”, “Discipline” and “Context parameter” are metadata. A context element of the HSS dictionary is mapped into a metadata value that can be taken by its corresponding context parameter. All the other concepts (Antonym, Related noun, Synonym and Translations) are translated into Wiki links.

Finally, to insure the extensibility of our Wiktionary to many languages (such as the Amazigh) and dialects of the Maghrebi countries, we propose to build a Wiki per language. The example of Figure 4 illustrates the structure of our HSS Wiktionary. This figure describes a French Wiki page for a variant of description of the word

"Entrepreneur" (one of its meanings in English is "contractor"). This page is annotated by the following metadata values:
• "Entrepreneur" is associated with the metadata "Word",
• "Sociologie" which corresponds to a value of the metadata "Discipline",
• "Français" which corresponds to the value of the metadata "Language",
• "13ième siècle" and "Maghreb" are respectively values of metadata temporal and geographic parameter. These two parameters represent the context elements of the context parameter "Discipline".

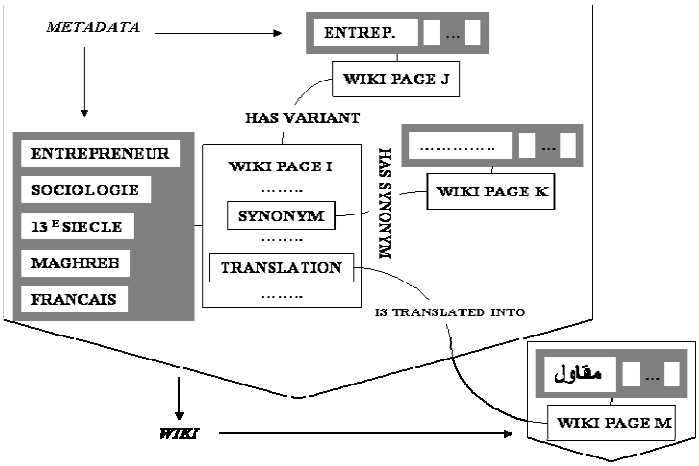


Fig. 4. An illustration of the HSS Wiktionary structure through an example.

The Wiki page associated to this variant of description of the word "Entrepreneur" is linked, in Figure 4, to other variants via the hyperlink "has variant". Moreover, this variant of description of the word "Entrepreneur" contains a hyperlink "is translated into" that links this Wiki page to the Wiki page representing the translation into Arabic of the word "Entrepreneur" for the same context of definition.

5 The Prototype

After translating the conceptual schema of our dictionary into a logical schema respecting the Semantic MediaWiki technology, we built a HSS Wiktionary prototype. Thus, we have chosen to build a Wiki by language and to establish links between them. Such a choice, allows us to construct a French-Arabic Wiktionary and then to extend it to other languages and dialects of the Mediterranean countries. Figure 5 is the welcome page of the Wiktionary. Through this page the user can enter a kind of Wiktionary (at the present time French and Arabic ones). He can also ask for the definition of a word (or its synonyms, or its close words) by giving all or some information (values of the context elements) about the context.

Fig . 5. The welcome page of our Wiktionary application.

The editor of the HSS Wiktionary (Fig. 6) includes, at the present time, a subset of the elements of ISO 1951 standard. Its extension to all elements of this standard or only to those useful for HSS field is possible. Using this editor, the user could annotate a Wiki page associated to an entry, by the metadata of its context of definition. He could also complete its description by using annotations associated with the elements of the schema issued from the ISO 1951 standard. Before entering a description (in a given language) of an entry (word) the user must first provide the context of the definition of this entry (i.e. the user must enter the discipline, the language concerned by the entry and the other context elements that validate and specialize its description). According to the context provided, the system will either propose to modify the last version of the description (if the entry already exists with the same context) or to create it. During the modification of an existing description (page) or its creation, the user has to use the proposed tags to add possible synonyms, antonyms, related nouns of the entry. Semantic MediaWiki translates these metadata into RDF.

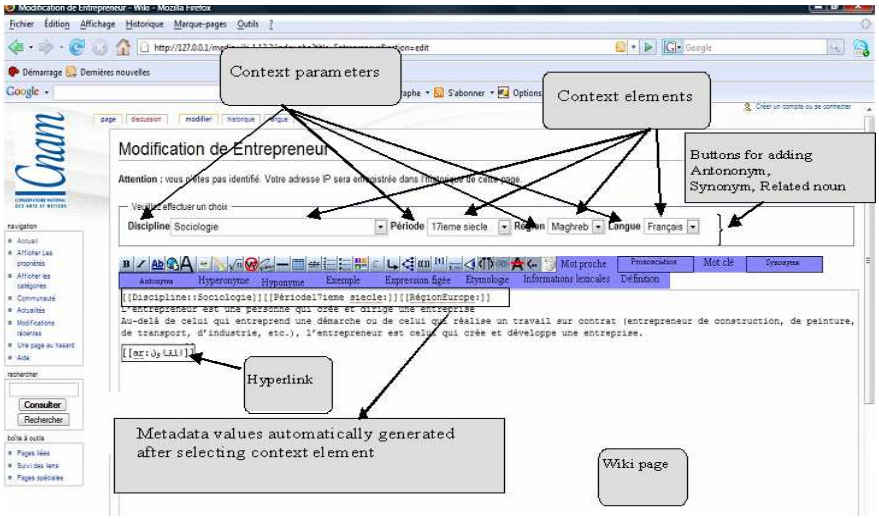


Fig. 6. Editor interface of the French Wiktionary.

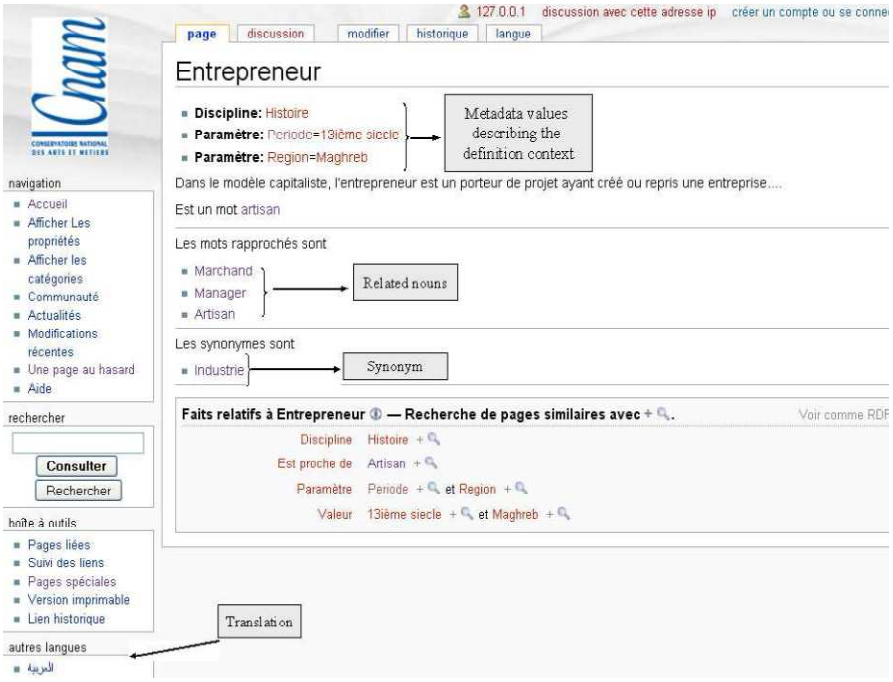


Fig. 7. An example of Wiki page consultation.

Note that due to the multicultural aspects of the HSS Wiktionary, an entry may not have a correspondent in a given target language. Let us note also that the global description of an entry could be obtained by gathering its variants in a Wiki page. The user may also wish to consult a description of an entry for a given context. The system, in this case, provides the description in which hyperlinks to synonyms, antonyms, related words and a correspondent translation appear. For example, the interface of Figure 7 is provided to a user who wants to obtain a description of the French word “Entrepreneur” for the context described by the metadata values.

6 Conclusion

We have described in this paper the HSS on-line dictionary. For this purpose, we have used, as required in the project specification document, the Wiki technology. The latter makes the content editable collaboratively and facilitates its exploitation.

After presenting the specificities of our on-line dictionary, we have synthesized them using an UML conceptual model. By taking into account the technical constraint associated to its implementation, we have proposed a first version of a prototype resulting from the mapping between the conceptual model of the dictionary and the Semantic MediaWiki metamodel. To evaluate the success of this first version, we asked experts, from different disciplines, to populate it with HSS words. While populating it, a cultural exchange of knowledge between researchers from the

Mediterranean countries will take place, allowing to share this knowledge between society members.
Future research will tackle the issue related to the separation between the presentation and the storage layer of the dictionary. As of today this separation was not possible for time reason. We intend to take advantage of SKOS (the W3C recommendation for the representation of thesaurus, taxonomies or any other controlled vocabularies) to perform such a separation. The latter will lead us to map the dictionary into multiple formats. In addition, we will take into account the access management aspect related to the security issues of the Wiktionary application. Finally, we will integrate the Amazigh language and its graphical symbols into the Wiktionary.

References

1. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=36609
2. Mathieu Mangeot: Papillon project: Retrospective and Perspectives. Proc. of Acquiring and Representing Multilingual, Specialized Lexicons: the Case of Biomedicine, Ed. Pierre Zweigenbaum. LREC workshop 2006, Genoa, Italy, 22 May 2006.
3. Sylviane Descotte, Jean Luc Husson, L. Romary, M. Van Campenhoudt and N. Viscogliosi: From specialised lexicography to conceptual databases: which format for a multilingual maritime dictionary. The 2d International Conference on Maritime Terminology. Turku, Finland. 12 May 1999.
4. Francis Bond, Jim Breen: Semi-automatic refinement of the JMdict/EDICT Japanese-English dictionary. 13th Annual Meeting of The Association for Natural Language Processing, pages 364-367, Kyoto, 2007.
5. Vuthichai Ampornaramveth, Akiko Aizawa, Saikam: Collaborative japanese-thai dictionary development on the internet. The Asian Association for Lexicography (ASIALEX) Biennial Conference, Korea, 2001.
6. Kensaku Kawamoto, Yasuhiko Kitamura and Yuri Tijerino, KawaWiki: A SemanticWiki Based on RDF Templates. Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT 2006 Workshops WI-IATW'06),2006.
7. Sebastian Schaffert, IkeWiki: A Semantic Wiki for Collaborative Knowledge Management. 15th IEEE International Workshops on STICA06, Manchester, UK, June 2006.
8. Michel Buffa, Gaël Crova, Fabien Gandon, Claire Lecompte, Jeremy Passeron, SweetWiki: A semantic Wiki. Special Issue of the Journal of Web Semantics on Semantic Web and Web 2.0, Volume 6, Issue 1, Pages 84-89, February 2008.
9. Malte Kiesel: Kaukolu: Hub of the Semantic Corporate Intranet. Workshop From Wiki to Semantics, ESWC 2006.
10. Markus Krötzsch, Denny Vrandecic, and Max Völkel: Semantic MediaWiki. The 5th International Semantic Web Conference, ISWC2006, 2006
11. SKOS. http://www.w3.org/2004/02/skos/
12. OmegaWiki. http://www.omegaWiki.org/Meta:Main_Page
13. Wiktionary: English Entry Layout http://en.wiktionary.org/wiki/Wiktionary:Entry_layout_explained
14. Wiktionary: French Entry Layout. http://fr.wiktionary.org/wiki/Wiktionnaire:Structure_des_articles

Automating Content Generation for Large-scale Virtual Learning Environments using Semantic Web Services

Ian Dunwell[1], Panagiotis Petridis[1], Aristos Protopsaltis[1], Sara de Freitas[1], David Panzoli[1] and Peter Samuels[2]

[1] Serious Games Institute, Coventry University, UK
{idunwell, ppetridis, aprotopsaltis, sdefreitas, dpanzoli}@coventry.ac.uk
[2] Faculty of Engineering and Computing, Coventry University, UK
{psamuels@coventry.ac.uk}

Abstract. The integration of semantic web services with three-dimensional virtual worlds offers many potential avenues for the creation of dynamic, content-rich environments which can be used to entertain, educate, and inform. One such avenue is the fusion of the large volumes of data from Wiki-based sources with virtual representations of historic locations, using semantics to filter and present data to users in effective and personalisable ways. This paper explores the potential for such integration, addressing challenges ranging from accurately transposing virtual world locales to semantically-linked real world data, to integrating diverse ranges of semantic information sources in a user-centric and seamless fashion. A demonstrated proof-of-concept, using the Rome Reborn model, a detailed 3D representation of Ancient Rome within the Aurelian Walls, shows several advantages that can be gained through the use of existing Wiki and semantic web services to rapidly and automatically annotate content, as well as demonstrating the increasing need for Wiki content to be represented in a semantically-rich form. Such an approach has applications in a range of different contexts, including education, training, and cultural heritage.

Keywords: semantic web applications, virtual learning environments, information systems applications

1 Introduction

Increasingly, web content is represented using semantic metadata formats which support the compilation and interlinking of information. One of the key advantages to such approaches is the ability to query and search this information using novel methods, such as relating 'geocoded' data to other web-based information repositories. Geocoding (for a comprehensive summary: see Goldberg, 2007), the process of transcribing named places to absolute geographic coordinate systems, has allowed

information to be queried in a host of different ways in various application areas; including public health (Rushton et al., 2006) and epidemiology (Krieger, 2003). Through the integration of these services with information sources such as Wikipedia, the potential exists to link both semantic and non-semantic Wiki content to real world locales.

In this paper, we explore an application of this combination of services to virtual learning environments. Whilst many learning environments currently rely on subject matter expertise for content generation and validation, such an approach is time-consuming, costly, and often involves the duplication by-hand of information already available from other sources to suit the format and context of the learning environment. Therefore, we consider a potential solution to be the use of geocoding data to identify an article held on a Wiki, and hence rapidly and autonomously annotate large environments, which can mirror real-world locales based in the past or present. This combination of a virtual world with a dynamic, editable, and peer-reviewable Wiki-based data source has immediate advantages in being able to support exploratory, peer-based learning models without requiring substantial input and guidance from subject matter experts. The source of the information driving the annotation in the proof-of-concept we describe in Section 4 is Wikipedia; However besides providing a demonstration of how semantic services can bridge into non-semantic data sources, this proof-of-concept highlights the long-term benefits that could be achieved by using fully semantic representations of information in these services.

Following an introduction to the state-of-the-art in Section 2, we go on to describe in Section 3 several concepts which underpin the implementation of systems using geocoding web services to provide content for learning environments which can be fed back to users in a range of novel and innovative ways. Section 4 details an implemented proof-of-concept using this approach, which uses the Rome Reborn (www.romereborn.virginia.edu) model alongside the GeoNames service (http://ws.geonames.org) to provide information to a user navigating the model in real-time. This proof-of-concept shows a simple approach to feeding information back to the user that can be expanded upon, and to this end we discuss the challenges faced in creating more sophisticated environments and learning experiences as well as the potential for future work in Sections 6 and 7.

2. Background

Many existing approaches towards creating virtual learning environments utilise the knowledge of subject matter experts to annotate content by hand. The integration of Rome Reborn with Google Earth (http://earth.google.com/rome/), for example, uses such an approach. Other applications in cultural heritage, such as the ARCO system (White et al., 2004), seek to allow curators or developers to create a dynamic virtual exhibition through the use of XML-based procedural languages, allowing dynamic modelling capabilities to be realised in a virtual scene. This technique enables the development of dynamic, database-driven Virtual Worlds, created by building

parameterised models of virtual scenes based on the model and the data retrieved from the database (White et al., 2009).

The MESMUSES project (Meli, 2003) highlights an interest amongst teaching institutions to provide learners with 'self-learning' environments, providing them with an opportunity to explore various knowledge spaces (i.e. digital information on museum artefacts) in a free-roaming virtual world. To this end, the MESMUSES project demonstrated a system that accesses cultural information through the novel concept of 'knowledge itineraries'. These itineraries represent a series of thematic paths that visitors can choose to follow, and when doing so various resources are offered to them including examples and explanations. Furthermore, the system, with the use of personalization methods, offers different knowledge domains to different categories of visitors. Similarly, the ART-E-FACT project (Marcos, 2005) proposes that the use of the semantic web can enable learning institutions to make cultural content available to researchers, curators or public in an increasingly meaningful and user-centric fashion. Marcos et al. suggest that the use of digital storytelling and mixed reality technologies can also create a new dimension for artistic expression. Within cultural heritage applications, therefore, there are multiple benefits that can be gained from using semantic technologies, such as the potential to gather data from across the web, filter this data using metacontent, and present it to the user in a dynamic and customisable fashion.

Outside of the specific domain of cultural heritage, attempts have also been made to annotate virtual environments to aid user navigation. For example, Van Dijk et al. (2003) demonstrate an approach using geometric and spatial aspects of the virtual worlds and its objects. Within a map of the environment, landmarks are added to identify various locales, supported by a personal agent with knowledge about the current position, visual orientation of the visitor, objects and their properties, geometrics relations between objects and locations, possible paths towards objects and locations, routes to the user and previous communications. By comparison, Pierce and Pausch (2004) present a technique for navigating large virtual worlds using place representation and visible landmarks that scale from town-sized to planet-sized worlds, whereas Kleinermann and colleagues (2008) propose methods in which the domain expert annotates the virtual world during creation, suggesting that since the world is being created using ontologies, the resultant semantic annotation will be richer. Navigation can then exploit these semantic annotations using a search engine - assuming, however, that the world has been created and annotated using this method.

Fundamentally, the approaches listed in this section primarily rely on direct intervention from either designer or subject matter expert to annotate an environment. Whilst it is undoubtedly the case that such an approach allows for certainty in the accuracy and validity of content, this approach also has drawbacks in requiring human resources for not only creation but also maintenance, since unless content is updated regularly, the experience is unable to retain users for long periods of time as content is gradually consumed. In work being undertaken at the UK Open University through the Luisa project (Mrissa et al., 2009, Dietze et al., 2008), advanced semantic web searching and acquisition techniques are being used to personalise and filter

information dynamically in real-time. This allows for complex reversioning and acquisition of data provided to the user, and can support more complex educational requirements for wider ranges of learner groups. Potentially, in the educational domain, this advance in intelligent querying is supported by service orientated architectures and may support advanced educational scenario developments that may be tailored to individual user requirements. For all these applications, extending the databases at the core of each system to include other web-services and sources of information could enhance both the volume and quality of content in virtual exhibitions and environments, easing user navigation and creating deeper, more compelling learning environments.

Wiki technology offers a basis for supporting such approaches. Since content can be simultaneously generated and peer-reviewed by a large base of users, large volumes of data may be generated with less expense or increased speed when compared to individual subject matter experts. Although validity can pose a concern, semantic representation simplifies identification and comparison between different data sources and can therefore aid designers in addressing this concern, and throughout the background literature an increasing motivation to create virtual worlds which are annotated in increasingly user-oriented fashions can be observed. In the remainder of this paper, we describe an approach to extending and automating annotation for large virtual environments which are based on real-world locales. This approach focuses on the use of geographic information services together with semantic web and Wiki data, to obtain data on points within the world which can form the basis of more complex filtering and mining techniques such as those proposed by the Luisa project (Mrissa et al., 2009). In the next sections, through a demonstrated proof-of-concept, we show the potential of such an approach to quickly and automatically annotate a virtual world with a large volume of information from a Wiki.

3. Automating Content Acquisition via Geocoding

In this Section, we describe in general terms the concepts behind the integration of web services such as Wikis with virtual worlds using geocoding. Whilst the proof-of-concept described in Section 4 focuses on the combination of Wikipedia with historical environment via the GeoNames service, this reflects a more general approach consisting of three fundamental steps. Firstly, coordinates in virtual space must be converted to a form suitable for input into the wide range of web-based GIS systems and databases. Secondly, the information obtained must be filtered to ensure relevance to the locale, time period, and usage scenario. Finally, this information must be presented to the user in an appropriate and coherent fashion. This section discusses these three issues in some detail; although the large number of web services, coupled with their diversity, makes generalisation a challenge, an attempt is made to describe the solution in as general terms as possible.

3.1 Coordinate Conversion

GIS systems commonly take coordinates as longitudes and latitudes. By comparison, virtual worlds contain arbitrary, typically Cartesian, coordinate systems. The conversion of these virtual coordinates to a real-world location can be simplified for relatively small areas with little variation in elevation (e.g. cities) by approximating the longitude and latitude as a Cartesian system. In this case the translation between a point x_0, y_0 on a virtual plane and real-world geographical location at longitude and latitude x_t, y_t can be expanded into simultaneous equations of the form:

$$x_t = \alpha\cos(\theta)x_0 - \alpha\sin(\theta)y_0 + t_x .$$

$$y_t = \alpha\cos(\theta)x_0 - \alpha\sin(\theta)y_0 + t_y . \quad (\mathbf{1})$$

Where α, θ, t_x and t_y describe the rotation, scaling, and translation between the two coordinate systems. This assumes both coordinate systems are aligned along the z-axis; if this is not the case then this can easily be accommodated by introducing the z coordinate in the above equation, although it should be noted the vast majority of 3D models have an immediately apparent vertical axis around which the virtual coordinate system can be defined. Solving these equations simply requires a set of virtual points and their real-world equivalents. The accuracy of the solution is, therefore, predominantly dependent on the accuracy of this point-set, and specifically, how accurately each real-world point identified is mirrored in virtual coordinates (whilst floating-point accuracy will also affect the solution, its impact is negligible in comparison). This is a challenge, since limitations in the fidelity of the virtual space can affect how accurately points can be identified and mapped to real-world points. Similarly, the resolution of GIS data sets limits the accuracy with which locations can be defined, as do difficulties in defining the centre of a building.

There are two potential approaches to increase accuracy. The first is to use GPS hardware to more precisely identify real-world points corresponding to virtual ones, although this is often impractical since it requires real-world presence. Hence, a more desirable solution may often be to increase the number of points sampled and average multiple solutions to Equation (1). It should be noted, though, that a 3-point sample proved adequate for the example given in Section 5, since the level of accuracy required is dependent on how tightly-distributed the information points in queried web services are, and this distribution (in the case of the systems used within the case study) has all points at least 20m apart. Given the capability to rapidly translate points in real space to virtual space and vice-versa, an immediate question is how to best identify the geographic point(s) which best represent the user's interest. It is possible to simply convert the position of the viewpoint in virtual space to a GIS location and provide data on that location, although this is only likely to represent the actual point of interest if the user is above the object looking down. Similarly, getting the user to directly intervene and click the point is a solution, but more seamless integration between information and virtual world could be achieved through saliency mapping and scene analysis, so as to base the selection of objects on the perceptual traits of the user. Studies of related problems in computer science, ranging from interest

management to selective rendering (Dunwell and Whelan, 2008, Sundstedt et al., 2005) have demonstrated that proximity alone is not an accurate measure of salience. A coarse solution is to generate a pick-ray (line in virtual space) from the centre of the viewpoint into the scene and select the first object it intersects, although more sophisticated approaches such as that of Yee et al., (2001) show the potential gains from more accurately modelling how users perceive, and interact with, three-dimensional scenes. Additionally users' historical behaviours can be used to detect motion of the viewpoint around objects and other traits that may indicate interest.

3.2 Filtration

Foremost amongst the advantages of semantic content representation in this context is the ability to filter information on a semantic level. Asides from the spatial filtering achieved through the use of a geocoding service, data can be filtered according to criteria such as date, particularly relevant in the case of Ancient Rome. In our case, we consider both the filtering of non-semantic data held on Wikipedia through a conventional keyword search, and also the use of DBpedia datasets to provide a semantic version of content. Due to the simplicity of the filtering task in this case, given the straightforward application of date and spatial filters, it is possible to use either semantic or non-semantic versions of Wikipedia content; however, in the longer term, more sophisticated applications and increased content volume will benefit from the advantages semantic search techniques bring.

3.3 Presentation

Finally, the filtered information must be presented to the user in a coherent form. This is of particular interest to developers of virtual learning environments, who often seek to present and represent information in new and innovative ways. Simple approaches can include the return of text and images within the application interface; the form in which these are presented to the user can be tailored by the designer to meet practical and pedagogic concerns. It is possible, for example, that this information could be used to create questions, (e.g. requiring the user to name a location then testing against available web data), images showing real-world equivalents of virtual locations, and other learning objects. This area has strong potential for future work: a more advanced method may be the use of virtual characters, for example as Vygotskyan learning partners (Rebolledo-Mendez et al., 2009). In this context, the information returned could form the knowledge base of the partner. The potential to autonomously provide data to an artificial intelligence from background web-services can enable these characters to behave more realistically and dynamically; conversational agents such as those of Daden (http://www.daden.co.uk/chatbots.html) have demonstrated this capability in second-life, and extending this technique to large-scale virtual recreations of real-world locales using the geocoding approach is an interesting avenue for future work.

4 Case Study

In this section, we present our working proof-of-concept which integrates the key concepts introduced in Section 3 into a working software platform. We present this firstly in terms of the high-level architecture, applicable to any real-world model, and then secondly with respect to a case study using a large-scale model of Ancient Rome developed as the principal output of the Rome Reborn project (Guidi et al., 2005).

4.1 System Architecture

Our prototype solution to automated annotation, using the principles described in Section 3, is described in this section. The key processes - coordinate selection, return filtering and conveying information to the user – are achieved through integration of the JME and LOBO APIs within a proprietary core engine. Input and output are handled again via an external API, in this case Java's Swing. On a more general level, this can be seen as a discretisation of the central tasks of rendering the model in real time, providing a web-interface, and providing a common user interface which integrates both the textual data retrieved from searches with the rendered three-dimensional scene. Therefore, these components may be interchanged to suit other hardware platforms (such as mobile devices) or languages as required. We therefore focus our discussion on the core engine, shown in Figure 1.

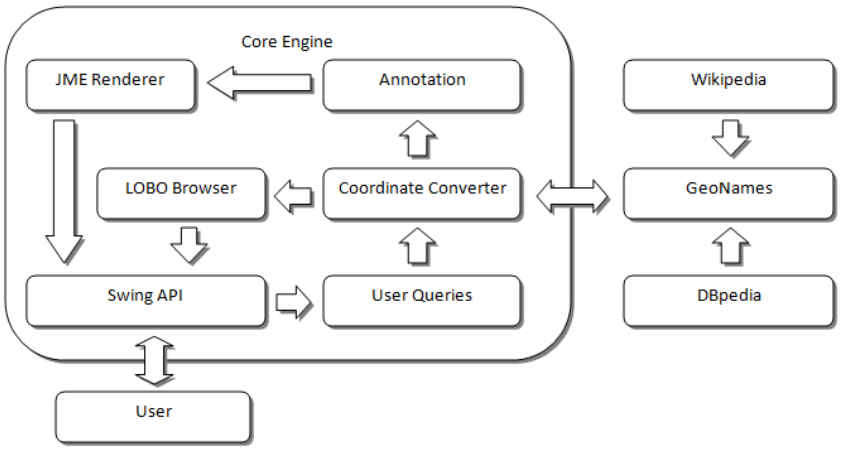


Fig. 1. Architecture using semantic web services to provide educational content and annotation for virtual representations of real-world locations both automatically and through user request

Automated annotation is achieved prior to run-time by automatically generating queries based on the specified real-world coordinates of the model. These allow a large volume of data to be captured and represented as information points within the virtual world, as discussed later in this section (see Figure 3). These points are passed to the rendering engine and hence used to create content, derived from processing of the raw XML data returned from web-queries. In this case, web-queries are directed

to the GeoNames integration with Wikipedia (http://www.geonames.org/) to obtain semantically-annotated data in the form of XML in response to input latitudes and longitudes. This is then processed and used as the basis for the construction of Aand minimises unnecessary web traffic generated by multiple queries with identical returns.

Fig. 2. Real-time rendering of the Rome Reborn model

Within the architecture illustrated in Figure 1, GeoNames is used as a bridge into both the non-semantic Wikipedia, and the semantic DBpedia service. Through the conversion of coordinates from virtual to real space (and vice versa), annotation is generated and passed to the rendering engine, and user queries are also handled using an integrated browser supported via the LOBO API. These allow the user to request data on their current location, which is spatially filtered via the viewport orientation, and temporally filtered by keyword and date comparison to the semantic data returned. This allows the returned XML to be filtered to only provide articles relevant to Ancient Rome since the more general data provided by the GeoNames service is not temporally restricted. *Query generation* both with regards to automated and user requested information is performed by constructing an HTTP request and passing the filters to the GeoNames service within the URL via CGI scripting.

The efficacy of the keyword-based filtering system is dependent on the richness of the semantic annotation of the returned data. In our proof-of-concept, this is restricted

to that data returned by the GeoNames service: the Wikipedia article title, synopsis, and date if provided are queried, although the date is often held non-semantically within the synopsis. The increasing drive towards creating semantic Wiki technology will have long-term benefits for this approach, enabling more accurate filtration as well as allowing information to be returned in more versatile forms. The prototype currently returns information to the user by filtering supplied XML via an XSL stylesheet: future work described in Section 6 will explore the use of this information to drive dialogic interactions with virtual characters. In the next section, we describe the application of this architecture to the model of Ancient Rome shown in Figure 2.

4.2 Rome Reborn

The proof-of-concept developed as part of this research integrates the GeoNames service, Wikipedia, and the Rome Reborn model (Guidi et al., 2005) using the architecture described in Section 4.1 to provide instant and automated semantic annotation of the 3D model with over 250 articles. A Java application was developed which allows the user to navigate through the model in real-time. Figure 2 shows the real-time Java/OpenGL render of the whole model, which includes prominent features such as the Colosseum, Basilica of Maxentius, Tiber River and Ludus Magna. The user is able to navigate through the model using keyboard and mouse in a standard first-person interaction paradigm, with their input affecting the position of the view point in virtual space. A selection of sliders allows cosmetic effects such as lighting and fog to be changed dynamically.

Performance is achieved through a discrete level of detail (LOD) approach, which segments the city into areas with three levels of detail, selected dependent on distance from the viewport. This allows for flexible management of performance by controlling the distance at which various levels of detail are selected. Further performance can be achieved by manipulating the far clip plane, ensuring ~30fps can be maintained. With respect to information retrieval and processing, the use of a local cache, coupled with the fact retrieval is either done prior to user interaction (in the case of annotation) or limited to the rate at which the user explicitly requests information (rarely more than once per second), results in the ability to respond to a user request immediately.

Virtual to real coordinate conversion was achieved using the technique described in Section 3, with three reference points taken at the most prominent structures and joined with the real-world equivalents as defined on Google Earth to form point sets. As the user moves the view point, requests to the Geocoding service are automatically generated by performing this translation on the view point location. Hence, this prototype uses purely distance-based measures of content relevance - the nearer the view point is to a location, the more likely it is to be returned by the Geocoding search as the closest point within the database, we refer to this as 'proximity searching'. To create the information service, a second pane is added (Figure 3) which uses the LOBO API to add a pure Java web browser within the application. Information points are loaded into the scene as simple geometric objects by querying

the Geocoding service for the 250 points nearest the centre, converting these coordinates to the virtual coordinate system, and adding them to the world. When a user clicks on one of these points in the 3D space, the Geocoding service (or cache) returns XML data centred on that point, which includes a title, summary, and link to the Wikipedia article nearest the queried latitude and longitude. This XML is, in this case, filtered through direct parsing hard-coded into the application, as well as a generic XSL style-sheet which formats the data to present it to the user. The link to the style-sheet is added by directly inserting a line to the XML during processing. If the user wants further information on a point, a link is provided to the Wikipedia article. It thus provides a simple example of the return filtering process. Local caching is used as shown in Figure 1 to minimise unnecessary web traffic. The solution demonstrates a simple proof-of-concept, showing all three components of the model described in Section 3 working to provide autonomous and dynamic information to the user as they explore the model.

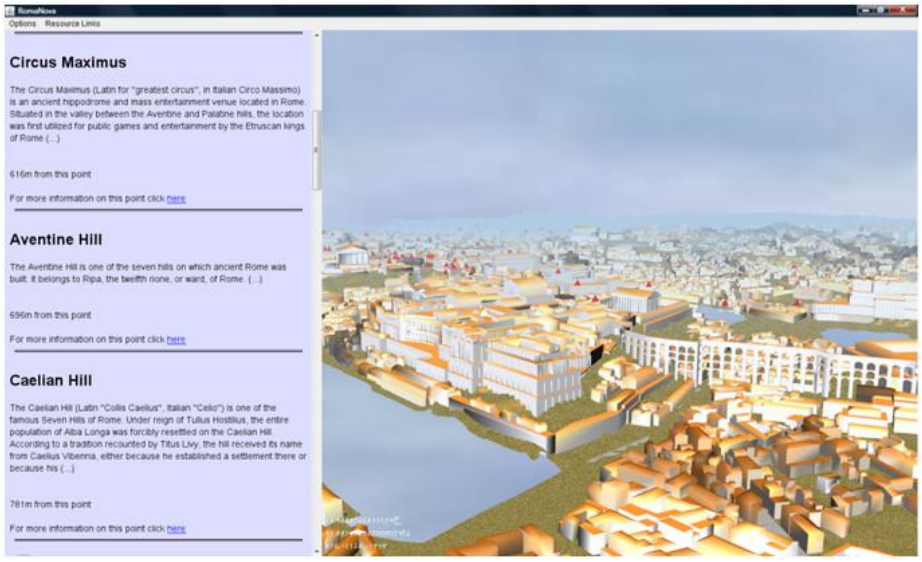


Fig. 3. The interactive application with information queried using GeoNames (left) rendered using XML/XSL

5. Discussion

The collaborative and dynamic nature of Wikis makes them an interesting area for pedagogic design. A central concept to Wikis is the notion of users as producers and evaluators, as well as consumers of content, and exploring this potential in a 3D virtual space within the proof-of-concept suggests several issues that need to be tackled. Firstly, the abstraction of the Wiki paradigm from the familiar interface may result in users failing to recognize it as such, and hence behave only as content consumers. Supporting the transition of the Wiki concept to different representational media requires that users continue to interact as content producers as well as

consumers, and this has repercussions for how user interaction is modeled and how interfaces are designed. A potential pedagogic advantage of an environment such as that developed is the facilitation of experiential (Kolb, 1984) or exploratory (de Freitas and Neumann, 2009) models of learning. The ability to immerse a learner within a detailed 3D environment, and utilize semantic services to provide detailed content and the ability to autonomously handle information requests and provide increasingly dynamic environments may have direct benefits to learning transfer. In the context of this paper, whilst preliminary qualitative work reinforces this hypothesis, significant challenges exist in defining how virtual learning environments can be accurately assessed. This is particularly the case where principal desired outcomes lie beyond the simple recollection of facts – a control study of virtual versus real scenarios may offer some insight in this respect, but would fail to reflect to typical role of virtual worlds as augmenting, rather than replacing, existing instructional techniques.

Furthermore, from an educator's perspective, one of the most prominent issues arising from the application of techniques such as those described within this paper is the transition of subject matter experts from content creators to content evaluators. As collaborative web-based knowledge bases expand, existing subject matter expertise is becoming increasingly available and accessible across disciplines. Furthermore, advances in how this information is represented (e.g. metadata formats) allow for versatility in how it is presented. Therefore, the role of educators and subject matter experts when designing learning environments increasingly becomes centred on the definition of information filters and presentation formats, so as to ensure that information is conveyed to learners in a valid and appropriate manner. Similarly, as virtual learning environments move towards experiential and situative pedagogies (Egenfeldt-Nielsen, 2007), and feature increasingly sophisticated intelligent tutors and characters, pedagogic design must support both learning within the environment itself as well as the integration of such environments across the curriculum as a whole. Whilst a key advantage of the technique described is that is capable of supporting exploratory learning, this infers that the usual cautions that should exist when creating exploratory learning experiences need to be considered. Foremost amongst these is the potential for the learner to deviate towards activities that fail to align with the desired learning outcomes. To overcome this, guiding the learner within the environment can be done in subtle ways using perceptual cues (Dixit and Youngblood, 2008), and integrating such models more fully with the methods described in this paper may be one avenue for creating experiences which guide the learner without constraining them in a perceivable way. The introduction of 'game elements' such as objectives, missions, or timed activities also has potential for increasing learning transfer when compared to open simulations (Mautone, 2008), and can also support more structured learning experiences within open, exploratory, environments.

Development of the working proof-of-concept identified a number of technical challenges. Firstly, although the visualization of the model itself is somewhat beyond the scope of this paper, rendering a large environment in real time is computationally intensive, and the overheads incurred by attempting work on such a scale can often

prove restrictive: for example, annotating the environment incurs additional performance overheads, and the level of annotation must be carefully balanced so as not to overload the user with information. Secondly, whilst GeoNames provides one potential link to semantic services, it is by no means the only such link which could be utilized. In our case study, we have demonstrated the case of using GeoNames to bridge into the non-semantic Wikipedia – however, careful consideration and selection of appropriate services is essential. In our case, bridging into Wikipedia was beneficial due to it containing the fullest collection of relevant information, though this is likely to change rapidly as services such as DBpedia offer increasing volumes of pure semantic content. In turn, this can be more fully utilized in different forms to add more depth to and variety to how information is represented and conveyed to the user.

One of the main issues regarding the use of automatically-generated educational content derived from the semantic web is the difficulty in ascertaining its accuracy and validity *in lieu* of a human expert. Doing so autonomously in a way which guarantees validity remains a substantial challenge, compounded by the need to also filter this data according to user needs. Furthermore, the dynamicism of web- based services and information, and the subsequent implications this has for instructional and educational programmes which are typically designed as repeatable courses (information will change over time as its web-based sources are edited, expanded, or removed), are an important consideration. Despite these drawbacks, the long- term advantages of evolving the techniques described within this paper are numerous: the large volume of freely available content allows for large volumes of relevant information to be rapidly integrated into the model, and at negligible cost compared to proprietary content development. For large, expansive areas, such as the city-scale model used in the case-study, these methods allow for more comprehensive and rapid annotation of content.

The more general challenges faced in developing and applying systems that integrate virtual worlds, web-based information, and intelligent tutoring for learning purposes must be addressed on both technological and pedagogic levels. This paper has presented several key technical issues, although the underlying pedagogy and, more fundamentally, purpose, of learning environments must always be a consideration in their long-term development and implementation. In the next section, we discuss some avenues for future work.

6. Conclusions and Future Work

This paper has further demonstrated the potential for the integration of information obtained from web-services into virtual learning environments. The solution is generically applicable: any real-world locale could be implemented using the approach described in the case study by simply changing the point set used when solving the equations presented in Section 3. The approach therefore has potential applicability to a wide range of learning environments, allowing developers to rapidly annotate content with information from a wide range of sources automatically and

dynamically. As mentioned in the previous section, developing pedagogies that realise the potential of this technology is a key area for future development. Open and exploratory environments may be capable of immersing and engaging learners, but if learning requirements are not met, they have limited use. Comparative evaluation of the various approaches that can be used to address this issue is a particularly relevant area for future study. The notion may also be introduced of using the results of queries to generate new queries autonomously, for example, Koolen (2009) demonstrate the potential use of Wikipedia pages, obtained as demonstrated in this paper though GIS coordinates, for book searches. Additionally, domain expertise can be modelled (White et al., 2009) to generate improved results.

On a more technical level, future work will focus around the latter two stages of the model, improving how information is sourced, filtered and conveyed to the user. This is a significant research challenge in many areas; in particular, using the information as a knowledge-base for intelligent tutoring systems driving virtual characters that behave and interact naturalistically requires advances in natural language processing, dialogue construction and pedagogy. Attempts to provide characters with a full, detailed knowledge-base must also consider the development of web-services as well as methods of content annotation to facilitate simpler integration. The integration and presentation of the information within the world in innovative ways is also an interesting area for future work, for example, weather patterns and air quality may be visualised in virtual spaces which provide information on real-world environments and systems.

A more sophisticated approach may be to embed semantic information into a virtual character as a knowledge base that the character can use to drive their own behaviour, and further enhance interactions with human users. This has the potential to be compatible with the hybrid architectures often used to control virtual humans (Conde and Thalmann, 2004, Donikian and Rutten, 1995, Sanchez et al., 2004), which are typically responsible for both low-level control of the agent such as navigation and obstacle avoidance, but also more complex interactions with the environment and other characters. Many challenges exist in realizing such techniques effectively: not only does it require a substantial amount of supplemental work, for example to animate and visualize the character, but increasing levels of realism and believability also imply increased challenges in creating characters able to adapt and behave dynamically. Consequently, the knowledge base may be limited to a specific context. Such techniques offer long-term potential for the application of semantic web-technology within virtual learning environments in a host of novel and interesting ways. However, the current state-of-the-art is often constrained by the large number of interrelated technical advances in many disciplines that are required to achieve these long term visions. In the next section, we describe a model which provides both a working, applicable approach for annotating worlds using existing technologies, whilst also accommodating and contributing towards these longer-term visions.

Finally, in this paper, we have focused on the user as a consumer rather than generator of content. Future potential exists for the use of virtual worlds to also allow users to create semantic Wiki content in innovative ways, by interlinking and

interacting with objects in virtual space. There is also an increasing demand for 3D content that is itself semantically-annotated (Spagnuolo and Falciendo, 2009). The methods described in this paper could provide a basis for allowing semantic annotation to be created for models such as Rome Reborn automatically by inverse geocoding. More significantly, as semantically represented 3D content becomes increasingly available, the ability to compose worlds autonomously using an integrated approach that adds content to the world based on its meaning and relevance to the learner, promises the potential to create sophisticated, adaptive learning environments.

References

1. Brusilovsky, P., Peylo, C.: Adaptive and intelligent web-based educational systems. International Journal of Artificial Intelligence. 13, 159--172 (2003)
2. Conde T., Thalmann, D.: An artificial life environment for autonomous virtual agents with multi-sensorial and multi-perceptive features. Computer Animation and Virtual Worlds 15, 311—318 (2004)
3. de Freitas, S., and Neumann, T.: The use of 'exploratory learning' for supporting immersive learning in virtual environments. Computers and Education, 52, 343—352 (2009)
4. Dietze S., Gugliotta, A., Domingue J.: Towards context-aware semantic web service discovery through conceptual situation spaces. In: CSSSIA '08: Proceedings of the 2008 international workshop on Context enabled source and service selection, integration and adaptation. 28, 1—8 New York, NY, USA. (2008)
5. Donikian, S., Rutten, E.: Reactivity, concurrency, data-flow and hierarchical preemption for behavioral animation. In: Fifth Eurographics Workshop on Programming Paradigms in Graphics. 137—153 Springer-Verlag (1995)
6. Dunwell I., Whelan J. C.: Spotlight interest management for distributed virtual environments. In: 14th Eurographics Symposium on Virtual Environments (EGVE08) 1, 56--64 (2008)
7. Dixit P. N., Youngblood, G. M.: Understanding information observation in interactive 3d environments. In: Sandbox '08: Proceedings of the 2008 ACM SIGGRAPH symposium on Video games. 163--170. New York, NY, USA (2008)
8. Egenfeldt-Nielsen S.: Beyond Edutainment: The Educational Potential of Computer Games. Continuum Press (2007)
9. Guidi G., Frischer B., De Simone M., Cioci, A., Spinetti, A., Carosso, L., Micoli, L., Russo, M., Grasso, T.: Virtualizing ancient rome: 3d acquisition and modeling of a large plaster-of-paris model of imperial rome. In: Proceedings SPIE International Society for Optical Engineering, 5665, 119--133 (2005)
10. Goldberg D. W., Wilson J. P., Knoblock C. A.: From text to geographic coordinates: The current state of geocoding. URISA Journal. 19, 33—46 (2007)
11. Kolb, D. A.: Experiential learning : experience as the source of learning and development. Englewood Cliffs, N.J: Prentice-Hall. (1984)
12. Koolen M., Kazai G., Craswell N.: Wikipedia pages as entry points for book search. In WSDM '09: Proceedings of the Second ACM International Conference on Web Search and Data Mining 44—53. New York, NY, USA (2009)
13. Kleinermann, F., Mansouri, H., Troyer, O. D., Pellens B., Ibanez-Martinez, J.: Designing and using semantic virtual environment over the web. 53--58.

14. Krieger N.: Place, space, and health: GIS and epidemiology. Epidemiology 14, 4 380—385 (2003)
15. Kallmann M., Thalmann D.: Modeling behaviors of interactive objects for real- time virtual environments. Journal of Visual Languages and Computing 13, 2, 177—195 (2002),
16. Mrissa M., Dietze S., Thiran P., Ghedira C., Benslimane D., Maamar Z.: Context-based Semantic Mediation in Web Service Communities. Springer, Berlin, (2009)
17. Meli M.: Knowledge management: a new challenge for science museums. In: Proceedings of Cultivate Interactive, 9 (2003)
18. Marcos G., Eskudero H., Lamsfus C., Linaza M. T.: Data retrieval from a cultural knowledge database. In: Workshop on Image Analysis for Multimedia Interactive Services Montreux,Switzerland, (2005)
19. Mautone T., Spiker A., Karp M.: Using serious game technology to improve aircrew training. In: The Interservice/Industry Training, Simulation and Education Conference (I/ITSEC) (2008)
20. Pierce J. S., Pausch R.: Colorplate: Navigation with place representations and visible landmarks. Virtual Reality Conference, IEEE, 288 (2004)
21. Rushton, G., Armstrong M., Gittler J., Greene B., Pavlik C., West M., Zimmerman D.: Geocoding in cancer research - a review. American Journal of Preventive Medicine 30, 2 516—524 (2006)
22. Rebolledo-Mendez, G., Dunwell, I., Martinez-Miron, E. A., Vargas-Cerdan, M. D., de Freitas, S., Liarokapis, F., Garcia-Gaona, A. R.: Assessing neurosky's usability to detect attention levels in an assessment exercise. In: Proceedings of the 13th International Conference on Human-Computer Interaction. Part I. Berlin, Heidelberg, 149—158 (2009)
23. Sundstedt, V., Debattista K., Longhurst P., Chalmers A., Troscianko T.: Visual attention for efficient high-fidelity graphics. In: SCCG '05: Proceedings of the 21st spring conference on Computer graphics, ACM, New York, NY, USA, 169—175 (2005)
24. Spagnuolo, M., Falcidieno B.: 3d media and the semantic web. IEEE Intelligent Systems 24, 2 90—96 (2009)
25. Sanchez, S., Luga H., Duthen Y., Balet O.: Bringing autonomy to virtual characters. In Fourth IEEE International Symposium and School on Advance Distributed Systems. Published in Lecture Notes in Computer Science, 3061, Springer (2004)
26. Troyer, O. D., Kleinermann F., Mansouri H., Pellens B., Bille W., Fomenko V.: Developing Semantic VR-Shops for E-Commerce. Springer, London, (2006)
27. Van Dijk E., Op Den Akker H. J. A., Nijholt A., Zwiers J.: Navigation assistance in virtual worlds. Informing Science, Special Series on Community Informatics, 6, 115—125 (2003)
28. White, R. W., Dumais S. T., Teevan J.: Characterizing the influence of domain expertise on web search behavior.In: WSDM '09: Proceedings of the Second ACM International Conference on Web Search and Data Mining. New York, NY, USA, 132--141 (2009)
29. White, M., Mourkoussis N., Darcy J., Petridis P., Liarokapis, F., Lister, P., Walczak, K., Wojciechowsky, R., Cellary, W., Chmielewski J., Stawniak M., Wiza W., Patel M., Stevenson J., Manley J., Giorgini F., Sayd P., Gaspard F.: Arco: an architecture for digitization, management and presentation of virtual exhibitions. In: CGI '04: Proceedings of the Computer Graphics International. IEEE Computer Society, Washington, DC, USA 622—625 (2004)
30. Yee, H., Pattanaik, S., Greenberg D. P.: Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments. ACM Trans. Graph. 20, 1, 39—65 (2001)

Lab Service Wiki: a wiki-based data management solution for laboratories production services

Antoni Hermoso, Michela Bertero, Silvia Speroni, Miriam Alloza, Guglielmo Roma
Centre for Genomic Regulation, Barcelona, Catalonia, Spain
email: {toni.hermoso, michela.bertero, silvia.speroni, miriam.alloza,
guglielmo.roma}@crg.cat

Abstract. *Lab Service Wiki* is a Semantic MediaWiki implementation for the management of a production laboratory. Here we describe its implementation on a protein service lab. Users of the service enter information about a sample and the desired analysis to be performed by using a semantic-enabled form built on top of a wiki page. After submitting, a workflow is created, and the manager of the service can assign different experimental tasks to the lab operators. The final output is the generation of a report for the requester. Users and operators, according to their profile and granted permissions, can track the state of the requests and the associated experiments at any time. People interested in this implementation can access it at: http://labservice.biocore.crg.cat

Keywords: wiki, semantics, protein, workflow, laboratory

1. Introduction

Establishing a new lab-based service requires the implementation of dedicated data management systems to track and store experimental information in a proper way.

Nevertheless, many small and medium sized laboratories and research facilities still handle and track users' requests, experiment results, and analysis reports in a very rudimentary way. These 'outdated' practices consist of using only traditional paper-based notebooks for annotating Standard Operating Procedures (SOPs) and experiment results, no rule-based traditional emailing, phone calling, or even mailing in order to establish a communication with the requesters and assign concrete tasks to a lab member. Furthermore, the constant evolution of new laboratory technologies and the growing amount of data generated represent nowadays a daunting challenge in the implementation of a proper data management system.

Because of the more complex panorama we are facing nowadays, the enhancement of laboratory workflows has become a 'must' for a lab-based service [1], even with very qualified technicians. A proper defined workflow is highly required not only because it facilitates the already mentioned massive data handling, but also because it can help to better accommodate those quality assurance requirements that are currently demanded by upper authorities in most present-day facilities to ensure highest quality of the service.

Specialized literature and scientific software vendors has traditionally drawn a line between LIMS (Laboratory Information Management System) and ELN (Electronic Laboratory Notebook) systems [2]. Whereas the former ones are used for labeling and tracking samples along a workflow, managing lab inventory (such as reagents and mediums), and monitoring instruments, the latter ones are used for annotating raw, intermediary, and final experimental data, results, and reports associated to the samples, as well as ensuring the sharing of guidelines and relevant information among co-workers.

These two hypothetical systems are ideally meant to coexist in a service and they should be connected or even reside in a unique or shared informatics infrastructure, so that the different user profiles should not mind about the logic behind and simply perform their specific tasks in a natural and easygoing fashion.

With the advent of the Internet and the growth of affordable and easy-to-setup local network installations, laboratory management and annotation systems could be extended beyond the very experimental workplace [3]. Data could then be centralized, exchanged and processed in an in-house or outsourced server, and users in front of thin terminals, or even devices and equipment themselves, could act as clients against the central server.

Although there are many client-server applications in the market, a very convenient approach is using web-based soutions. This way, any modern browser may suffice, without any need to install additional software .

1.1. MediaWiki as a convenient approach

By the early 2000s, wikis started to emerge and being adopted as centerpiece tools in collaborating and group learning environments both in the Internet and in private networks. The most notable example is the non-profit online encyclopedia Wikipedia, built over the PHP-written MediaWiki software [4].

MediaWiki, as a web based wiki collaborative application, provides consistent concurrency handling and data integrity, ensuring that a user edit cannot overwrite a

coincidental other user's addition, and a familiar interface so no extensive training is needed for learning how to input data.

There have already been different approaches taking advantage of Mediawiki possibilities in biological laboratory data management. One example is ArrayWiki [5], a global public repository of microarray data and meta-analyses that host many relevant images and their original experiments. Another one is OpenWetWare [6], an online open-science community of, mostly, 'wet-labs' where diverse information such as protocols or courses is shared. It also features a wiki-based electronic notebook.
By default, MediaWiki offers to these systems an open and well-known collaborative environment where trackability and authorship can be followed in a fine-grain basis.

Parallel to this, during the last few years there has been an increasing interest in applying semantic web principles, meaning and concepts rather than the style and content of common-day web, to MediaWiki installations. One notable approach is Dbpedia [7], an effort to structure Wikipedia information. Articles and relationships such as categories, and other tagged a posteriori, are exported as Resource Description Framework (RDF) files, and these can be used for building up complex searches using SPARQL query language.

Another project is Semantic MediaWiki [8], an extension to MediaWiki platform that can be quickly installed in and add semantic capabilities to plain wiki installations. As a complement of Semantic MediaWiki, a recommendable addon is Semantic Forms, an extension that allows to create forms that can conveniently edit wiki pages in a structured manner through web forms and link their fields to semantic properties.

One of the better known examples of Semantic Mediawiki applied to the biological area is SNPedia [9], a wiki-based database of Single Nucleotide Polymorphism (SNP). Semantic properties addition enable that potential users cannot only perform common full-text searches, but also field specific ones, such as chromosome locations or the technology —for instance, Microarray model— used to generate the data.

Taking advantage of these new web technologies, we started to develop *Lab Service Wiki* - a wiki-based laboratory management system,
This web application is concretely meant to handle relevant experimental information related to protein cloning, expression, and purification steps, thus providing wet-lab researchers with a proper tool to meliorate the lab workflow and keep control of the overall laboratory activity.

A test implementation is available at: http://labservice.biocore.crg.cat

2. Lab Service Wiki

Our target facility, Protein Service, consists of 1 head and 3 technicians. It works as an internal service of potentially around 100 users in a research center. There is an average of 4 requests per month, which can have from one up to one hundred or more associated experiments.
Before any wiki implementation, researchers used to submit their requests through PDF-based forms sent by email to the service. Once received, the responsible of the service could plan a meeting with the requesters to further discuss the project and gather additional information. After its outcome, the request could be accepted, modified or denied and one or more experiments run based on the given request. As a final result, the researcher could receive the service product (the purified protein itself) along with a report describing the most relevant experimental information.

The drawbacks of this approach were multiple: first, all information related to requests, samples, and experiments were not likely to be annotated in a standard way; second, all changes to original experimental data could not be tracked accordingly; third, data files generated during the analysis ended up being spread among different physical and virtual media, and if they were not gathered all together, they could get lost after report generation. This panorama represented a serious hurdle to any effective action to be performed by an evaluation third-party.

2.1. Implementation

As explained above, because of its simplicity of use and extensibility, MediaWiki posed as a firm candidate for hosting a system that fulfills the given requirements. Despite setting up a plain wiki system with a set of templates, customized extensions and cron-programmed or resident web robots was a feasible possibility, using Semantic MediaWiki, and other related extensions, greatly simplified the design. Pages could be “tagged” and linked semantically in multiple ways, so there was no need to use any other external application to process them first (e.g., parsing wiki syntax with regular expressions) in order to associate them to specific content of other pages (translated in Semantic MediaWiki as property values).

First of all, to grant the right access to the users in *Lab Service Wiki*, we created the following different user profiles: 1) the Administrator, responsible of the creation of new templates, users management and their training; 2) the Researcher, customer who can submit requests to the service using pre-defined templates, view the status of his/her requests at any time, and retrieve the study reports when the experiments are complete; 3) the Lab Manager, responsible of the service who can create, edit, delete new experiments, associated to submitted requests, using predefined templates; and, finally, the 4) Lab Members, expert technicians who can add, edit experimental data, but cannot create or delete experiments.

Once researchers obtain an account, which is assigned by default to a generic group, they can therefore log in and generate a request using a template form. Even though the latter seems to be equivalent to the original PDF-based version, it takes advantage of the Semantic Forms extension and therefore provides searchable fields and other additional functionalities.

The request form is the starting data seed for the upcoming workflow and the different fields are coupled to predefined semantic properties. In order to avoid any misuse, different restrictions were introduced at the logical and input level. At the logical level, we defined different data types associated to the properties, such as string, number or boolean, and which values can be allowed. At the input level, we could define the default input type, for instance text, checkboxes and the possible values, which could also be filtered by using regular expressions. This last option is especially useful for refusing incorrect alphabet characters in biological sequences (nucleotide or amino-acid ones).
On one hand, the form cannot be submitted if users fill non-allowed input values in a restricted field. On the other hand, in case there existed a page with a not-allowed value, Semantic MediaWiki would depict a warning icon next to the conflicting value. Therefore, this could be studied and addressed by the wiki administrator.
So, both logical and input restrictions should need to be kept compatible and in sync for ensuring data integrity and quality.

Using this form, researchers are required to input both sample and project information, and therefore can submit the new request. This action creates a new wiki page, that can be subsequently modified by the submitter at any time before the lab manager has accepted it.

Meanwhile, the lab manager receives a communication by email that a new request has been submitted. He/she can eventually modify some information (for instance, during a personal meeting or communication with the requester) and finally accept or reject the current request, selecting the value of the field ‘status’ (available options are Pending, Accepted, Discarded, Closed). It is important to say that only the lab manager can modify the field status and decide whether to accept or not the requests.

Thanks to the semantic annotation of the pages and different parser and user functions provided by several MediaWiki extensions, it is technically possible to avoid that the requesters can make any later modification after the status has been modified. The same mechanism is also used to prevent that other users apart from the original requester may access to any other request.

Once a request is accepted, the lab manager can generate and associate several experiments to it. Experiment wiki pages will reside in a different namespace restricted only to lab members by default. However, whenever desired, the lab

manager may choose to open the access of specific experiments to the requester so they can follow closely the development of the request.

The experiment page is also handled through web forms and, for convenience, split in different tab pages matching to the different stages and type of analysis (in our concrete case: cloning/subcloning, expression screening, scale-up purification and mutagenesis).
Provided request data is automatically passed from its original request to the experiment page. When suitable, thanks to Semantic Forms capabilities, some request information fields are also mapped to a corresponding field in experiment forms. This way, we can keep sample information intact and the lab members can modify mapped fields according to their expertise, overriding so user's initial suggestion.
Experiment stages will be conditioned by the request, so if the user did not want to perform any mutagenesis analysis (and it was not changed by a manager either), that tab will not be displayed in the experiment interface.

During the experiments, different types of data and media files can be produced. These can be also attached to the experiment pages. Semantic Forms provides an easier way to use interface for uploading files than MediaWiki's defaults. In case of a huge amount of data, such as large size files, or a file format that might not fit well inside wiki pages, linking URLs is always a suitable option.

2.2. Workflow, reporting and user permissions

The workflow of the experiment can be managed in more detail if necessary (*see Figure 1*), usually highly desirable in bigger laboratories with several workers, by selecting the lab members once they start to work in the experiment or when they become in charge of a certain stage. They could be notified by email when their username is invoked in a value field. The completeness of certain tasks can also be notified by the responsible, so the manager (or the same lab members group) can move to a next analysis, which can often depend in the completion of a previous one.
After all tasks are finished, the manager can choose to create more experiment pages from the request if the outcome is not as expected, enable open access to the experiment results to the researcher, or even generate a report page from the data of the very request and the results of the different associated experiments.

Thanks to conditional clauses introduced in the different templates of the wiki pages, the different statuses should remain coherent and synced along the interconnected pages: Request → Experiments.

That is, once all experiments are finished and the status of the request is marked as closed, no other experiment associated to that request can be generated. The very

manager would not be able to modify this, for instance, by creating another experiment page and generating a new report once a former one was considered definitive, unless that task is requested to be performed by the wiki administrator.

If tight group-associated permissions are followed and wiki administrator only intervenes according to well-defined guidelines, there is no easy way of forging or tampering the workflow. Pages, ideally only through web forms, can be edited either by plain users, lab members or lab managers depending on the permissions granted to a group for a certain namespace. MediaWiki permissions also permit differentiating between editing and page creating permissions. For instance, as explained above, only lab managers would be able to open a new experiment page, but lab members would be able to edit them in collaborative fashion despite they cannot create them themselves.
Moreover, the semantic logic behind the different page types is never intended to be writable by the mentioned groups. That means neither templates, nor forms specification nor semantic properties. Updating them should be under the sole responsibility of the wiki administrator. Since certain edits could break the consistency and interlinking of the semantic data, and consequently also the user-specific permissions and the workflow, these kinds of changes are supposed to be performed on a stage server using a sample subset of the existing data.

The traceability of the workflow is ensured with the default MediaWiki 'recent changes' option and also by checking individual pages history. These two options can be restricted for different roles and at the user level with conditional clauses using semantic queries. It makes sense to disallow access to 'recent changes' access to plain users.
Another application of using inline-searching feature of Semantic MediaWiki is getting detailed table-like reports about the status and the current stage of the experiments for the lab manager and the workload of the facility to the potential clients. Researchers can also track their own pending and pasts requests from their own user page.
Different blocks of information can be viewable by the different roles. Common users might only see the number of requests on queue so they cannot get impatient if theirs are not processed as fast as they might have desired, but lab members could need more details, such as the number of experiments associated to each request, and their creation date, so they can make up their own priorities.

Of course, apart from all the semantic linking possibilities, lab operators can still use the system as a lab notebook, not only by adding comments in experiments themselves, but also creating pages that might summarize the experience gained from the different experiments in order to improve existing SOPs.

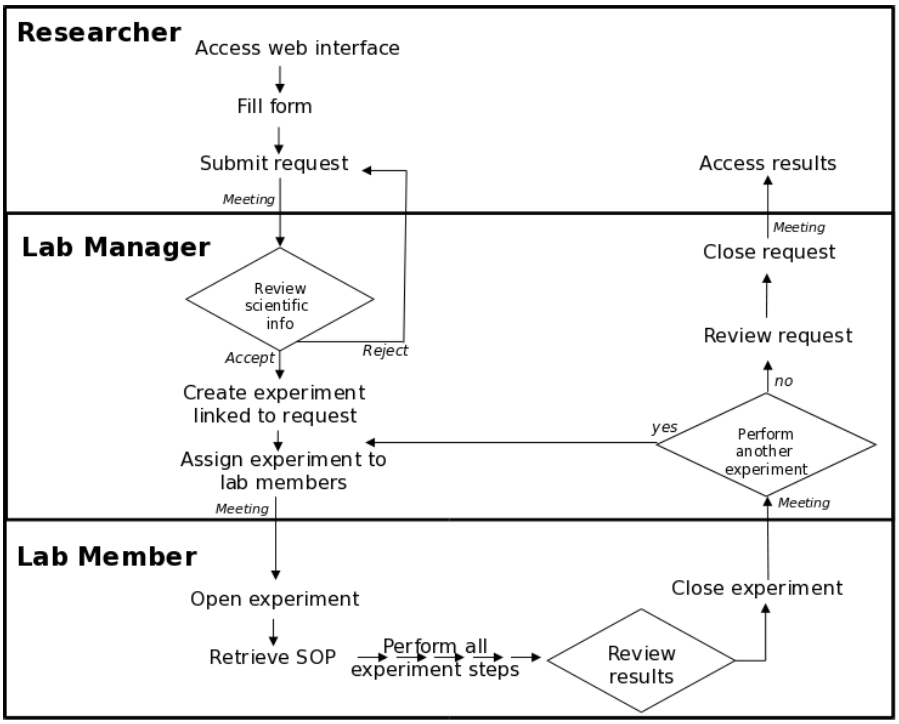


Fig 1. Simplified workflow of Lab Service Wiki.

3. Future advances

We could imagine about several features to improve the system. For instance, in the same facility, we could have access to an existing administration or catalog informatics system, which we could be interested in retrieving some data from. For this, we would need to use web robots against MediaWiki API, commonly written in Perl or Python scripting languages, which should mediate the connection to external databases and resources by querying them and updating accordingly the wiki. We could also trigger some applications to be run, for instance a sequence homology analysis. The result output could be linked externally within a wiki page, and also parsed in order to change a semantic field value.

Moreover, robots could also be used for automating some complex experiment workflows, so lab members do not need to generate hundred of pages from the same request if they expect to perform repetitive tasks.Since robots are to be put on move by triggers or in a periodical basis using system's cron, care must be taken to add the

necessary conditional logic requirements, in the wiki but also ideally in the very robot program, that may avoid any data breakage because of their failure. On the other, although they may have write rights, their editions should not be left unattended without validation by lab members.

As different experiments are performed, lab operators may notice that some data sets may repeat well enough to make them become a complex option value in a new simplified field that may encompass several previous ones. One solution for keeping backward compatibility with existing semantic definitions and, at the same time, trying to simplify the workflow (less fields to be filled), could be recurring to transclusion. In a few words, this means including the whole content of a page inside another one, so separate pages, as excerpts of information, can be kept apart for convenience and maintenance, and reused in the forms as many times as wanted.

We have centered the discussion upon a single research facility, but research institutions can also have many other hosted services not only in the same building, but spread in a campus, a city, a country or even all around the world. If different type of research analysis, using different equipment and in apart locations are to be performed upon the same sample, we might want sample information to be shared between the different experimental workflows. This is easier to be accomplished within the same MediaWiki installation, but it could also be worked out by using interwiki linking (as it is done between different languages versions of Wikipedia) and, more generally, thanks to well-designed web robots.

Semantic MediaWiki includes the feature to export existing relational information and semantic content as RDF files, which in turn could be analyzed by other software and used against other resources. And also, the other way around, external ontologies can potentially be imported into an existing Semantic Mediawiki installation. This may enhance the reporting we offer to the requestor by adding, for instance, functional genomics analyses by default thanks to Gene Ontology vocabulary [10]. Unexpected relationships may emerge if we datamine and process a bulk of experiments hosted in the system.

4. Conclusion

We described the implementation of *Lab Service Wiki* in our protein production service along with the proposed workflow to be used within the local research environment. Therefore, we consider that Semantic MediaWiki, as a concept empowered collaborative web system, is an excellent approach for designing a lab management and annotation system, which can be specifically adapted to the requirements of a modern day laboratory.

By using Semantic MediaWiki in contrast to a plain MediaWiki installation, we were able to link the content at a more detailed level that could be done by using only pages and categories. This way we assigned fine-grained permissions derived from semantic properties to active users and groups and, at the same time, both requesters and operators could benefit from specific searches and reports.

We also foresee many opportunities raised by the rational application of connecting different resources by web robots or by semantic content exchange.

Acknowledgments. We thank Oscar Gonzalez, Davido Castillo, and David Camargo for their technical support; as well as Luca Cozzuto, Francesco Mancuso, Ernesto Lowy, and our colleagues from other CRG core facilities for useful discussions.

References

1. Stephan, C., Kohl, M., Turewicz, M., Podwojski, K., Meyer, H.E., Eisenacher, M.: Using Laboratory Information Management Systems as central part of a proteomics data workflow. Proteomics, Jan 13 (2010)
2. Lims and ELN: 1 + 1 = 3. Scientific Computing. http://www.scientificcomputing.com
3. Ulma, W., Schlabach, D.M.: Technical Considerations in Remote LIMS Access via the World Wide Web. J. Autom. Methods Manag. Chem. 2005, 217--222 (2005)
4. MediaWiki. http://www.mediawiki.org
5. Stokes, T.H., Torrance, J.T., Li H., Wang, M.D.: ArrayWiki: an enabling technology for sharing public microarray data repositories and meta-analyses. BMC Bioinformatics 9, Suppl 6:S18 (2008)
6. Waldrop, MM.: Science 2.0. Sci Am. 298(5), 68--73 (2008)
7. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: 6. DBpedia – A Crystallization Point for the Web of Data. Journal of Web Semantics: Science, Services and Agents on the World Wide Web, Issue 7, 154--165 (2009)
8. Semantic MediaWiki. http://www.semantic-mediawiki.org
9. Cariaso, M., Lennon, G. SNPedia. http://www.snpedia.com
10. The Gene Ontology Consortium. Gene Ontology: tool for the unification of biology . Nature Genet. 25, 25--29 (2000)

OpenDrugWiki – Using a Semantic Wiki for Consolidating, Editing and Reviewing of Existing Heterogeneous Drug Data

Anton Köstlbacher[1], Jonas Maurus[1], Rainer Hammwöhner[1],
Alexander Haas[2], Ekkehard Haen[2], Christoph Hiemke[3]

[1] University of Regensburg, Information Science
Universitätsstr. 31, 93053 Regensburg, Germany
[2] University of Regensburg, Clinical Pharmacology, Department of Psychiatry
Universitätsstr. 31, 93053 Regensburg, Germany
[3] University of Mainz, Neurochemical Laboratory, Department of Psychiatry
Untere Zahlbacher Str. 8, 55131 Mainz, Germany;

anton.koestlbacher@sprachlit.uni-r.de; rainer.hammwoehner@sprachlit.uni-r.de;
jonas@maurus.net; ekkehard.haen@klinik.uni-regensburg.de;
hiemke@mail.uni-mainz.de

Abstract. The ongoing project which is described in this article pursues the integration and consolidation of drug data available in different Microsoft Office documents and existing information systems. An initial import of unstructured data out of five heterogeneous sources into a semantic wiki was performed using custom import scripts. Using Semantic MediaWiki and the Semantic Forms extension, we created a convenient wiki-based system for editing the merged data in one central application. Revised and reviewed data is exported back into production systems on a regular basis.

Keywords: drug database, medical information system, semantic wiki, data conversion

1 Introduction

PsiacOnline[1], a drug interaction database for psychiatry in German speaking countries, was released in 2006. As of 2010 it contains over 7000 drug interactions with comprehensive information on pharmacological mechanisms, effects and severity of each interaction. Strong emphasis lies on guidance how to handle interactions in practice. [1]

[1] *PsiacOnline* is an online service offered by *SpringerMedizin*: http://www.psiac.de

Built on top of the component-based and event-driven prado[2] framework PsiacOnline features an easy to use authoring tool for drug data. It also provides a simple XML interface for reusing data in other information systems, particularly Laboratory Information Systems (LIS).

After the system's introduction, we identified several additional data sources that are frequently used at the affiliated research institutes[3], providing content for PsiacOnline. These data sources consisted of different document types like Microsoft Excel sheets, Word documents, CSV data and relational databases. For example, biological pathway information for psychiatric drugs was kept in a Word document of which new versions were distributed to the lab staff via email. The staff also used relational databases with pharmacokinetic data that were part of a LIS system used for managing lab workflow. [2]

Other examples were manually edited Excel sheets with brand names and drug names or international non proprietary names (INN[4]) and ATC[5] code tables. They were distributed through an informal email based workflow in the lab.

Analysis of the various documents and their content showed that all content should be integrated into the existing dataset of PsiacOnline. This was the starting point of the OpenDrugWiki project, which now combines the converted and imported data sources with the existing PsiacOnline dataset. It offers an easy-to-use interface for collaborative editing of the unified data in one place.

The article shows a use case for semantic wikis in production environments in the field of professional pharmacological information in psychiatry. It describes why we chose a semantic wiki, how the import of the data is done, what the editing and review process looks like and how the data can be reused in existing and future software systems.

2 Why Use a Semantic Wiki?

Instead of trying to convert the data and thus expanding PsiacOnline and importing the data directly, we chose an approach that uses a semantic wiki as an intermediate system. This semantic wiki also provides a full replacement for PsiacOnline's authoring system. The decision for a semantic wiki was in fact not a single one, it was based on three decisions to the following questions: Why use a wiki? Why use semantic web technologies? And why use the combination of both?

---

[2] http://www.pradosoft.com

[3] University of Regensburg, Clinical Pharmacology, Department of Psychiatry; University of Mainz, Department of Psychiatry; University of Regensburg, Department for Information Science; Regional Hospital Kaufbeuren, Department of Psychiatry

[4] INN: International Non Proprietary Name: Generic name of a pharmaceutical ingredient issued by the World Health Organisation (WHO).

[5] ATC: Anatomical Therapeutic Chemical Classification System, used to classify drugs and other medical products, controlled by WHO Collaborating Centre for Drug Statistics Methodology (WHOCC).

2.1 Why Use a Wiki?

The main reason for favoring a wiki is that we will invite more institutes and authors to contribute to PsiacOnline, therefore supporting collaboration and versioning is of great importance. Wikis are well known to be of great use for distributed text editing and reviewing. This also applies to scientific communities. [3][4][5]

We wanted to replace the various inconvenient email based workflows by one structured storage and workflow system. We anticipate time savings in the participating organizations by centralizing and streamlining the editing and reviewing process. Time savings are already confirmed by users and are mostly achieved by discarding the inefficient email based workflows and by the possibility of editing all data in one place as well as the ability to instantly see changes made by other users.

Eventually, the affiliated research institutes do not only want to use the new system to publish information on drug interactions which is both, necessary and useful for psychiatrists or family doctors, but they also need a central platform for exchange of the underlying pharmacokinetic mechanisms which is important to motivate and execute further research.

2.2 Why Use Semantic Web Technologies?

Semantic Web technologies provide standards-based data exchange (RDF/XML) and storage methods (Triple Stores) and a powerful query language (SPARQL). This makes it easy to export parts of the semantic data back into production systems like PsiacOnline, LIS or other medical information systems (MIS) after they passed a review process.

The possible integration of data from other existing pharmaceutical or biomedical ontologies, for example the Open Biological and Biomedical Ontologies[6] was another reason to favor semantic technologies. Having the ability to provide data for services like Linked Life Data[7] or Linking Open Drug Data (LODD[8]) was additionally convincing.

2.3 Why Use a Combination of Both?

Based on the above arguments, the decision for a semantic wiki was identified as the best way to go forward. Both, wiki and semantic web technologies in combination, together with using a triple store connected to the wiki support each other in achieving the goals described above. Using a semantic wiki offers the possibility to extend the underlying data model at any time without trouble. This is important, as extensions will be needed when new data relevant for research becomes available.

[6] http://www.obofoundry.org
[7] http://linkedlifedata.com/
[8] http://esw.w3.org/topic/HCLSIG/LODD

3 Semantic Wiki Evaluation

We evaluated four of the mature semantic wiki engines: IkeWiki [6], Semantic MediaWiki [7] (with the Halo extension [8]), OntoWiki [9] and AceWiki [10]. Finally Semantic MediaWiki was chosen to be the product best-suited for our purposes, mainly because of its usability and the underlying MediaWiki [11] software, known to have a broad developer and user base and a big variety of available extensions. Further results of the evaluation, presented in a more general article, can be found in [12].

4 Implementation Details

OpenDrugWiki is based on Semantic MediaWiki and extensively uses templates, magic words[9], and various extensions. These include the Parser Functions extension[10], Semantic Results Format[11] and Semantic Forms[12] for convenient editing. Attached to the wiki we use the basic triple store for Semantic MediaWiki provided by Ontoprise [13]. It is based on Jena [14] and allows querying the semantic data that is stored in the wiki via a SPARQL [15] webservice. Having all semantic data available through a standards-based remote interface makes it easy to integrate new applications and gives us a way to bring it back into production systems.

4.1 Data Conversion and Importing

Converting and importing the various data sources proved not to be a trivial task. While reading the SQL databases and Excel sheets is a solved problem, the Microsoft Word documents are converted to Excel sheets using an add-in for Microsoft Word that was developed for this specific task.

After preprocessing the various data sources into Excel files and relational databases, the main import job is performed by a PHP CLI application. This application processes all data by matching known terms to semantic classes (brand names, drug interactions, INN etc) and merges duplicate entries coming from the different sources (see Fig. 3). It also computes redirections for sameAs-relations, based on the information provided in the legacy data sources. The import application then generates articles which are directly imported into MediaWiki using the its command line interface.

[9] http://www.mediawiki.org/wiki/Manual:Magic_words
[10] http://www.mediawiki.org/wiki/Extension:ParserFunctions
[11] http://www.mediawiki.org/wiki/Extension:Semantic_Result_Formats
[12] http://www.mediawiki.org/wiki/Extension:Semantic_Forms

For generating the articles we created boilerplates for each article class (brand names, pharmaceutical ingredients, drug interactions, others) consisting of various MediaWiki templates. Article boilerplates and templates were defined manually after analysis of the available data. A simple mapping between columns in the relational databases, the word and excel tables and the semantic properties is performed by the import tool.

Using this method, we generated and imported about 15,000 articles, consisting of about 3,000 brand name entries, 4,000 entries on pharmaceutical ingredients (INNs), 7,000 drug interactions and 1,000 other articles. All articles have data-type information and semantic properties which results in about 150k RDF triples. Reading and processing the data, generation of wiki articles and importing them into SMW take about one hour altogether.

4.2 Editing and Reviewing

After having successfully imported all data, the semantic wiki is used for editing and reviewing by the PsiacOnline authors as well as carefully selected associate authors. Since there is no Semantic MediaWiki extension available which supports a collaborative peer reviewed editing process, we are forced to manually track all changes made to the wiki articles. This is done by the core authors of PsiacOnline who immediately review all edits made by other authors. To control which data is exported back to production systems we store the user name and the revision date of each article as semantic properties. Only articles which were approved by one of the core authors are imported into production systems. This means, that a reviewer who checks edits of a normal author has to resave the edited articles, even if he himself made no changes. This process works perfectly for the moment, but cannot be seen as a long term solution.

4.3 Querying the Wiki

As a proof of concept and a useful application for the staff in the lab we created a simple Ajax powered web interface to retrieve data from the wiki. Given a list of drugs or brand names, it shows all drug interactions, the biological pathways involved and the citations on which the displayed information is based (see Fig. 2). This tool demonstrates the possibility to query the triple store attached to the wiki and can be used completely independent from the wiki itself. Being in private beta phase at the moment, it will be made publicly available, when the content is completely reviewed and double checked.[13]

The tool uses a PHP proxy script which queries the SPARQL endpoint, preprocesses returned data, and delivers it back to the Ajax application using JSON. Preprocessing consists primarily of dealing with the returned XML and character-set related quirks.

[13] Project Website: http://www.opendrugwiki.org/wq

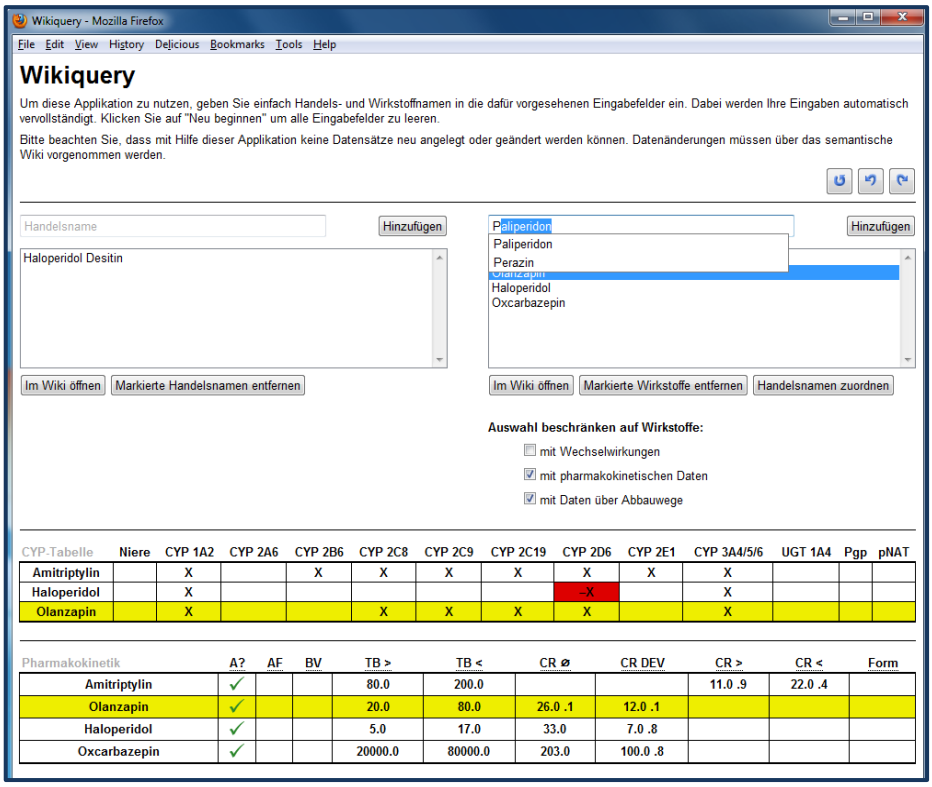


Fig. 1. Screenshot Wikiquery-tool (german)

4.4 Exporting Data

One of the crucial requirements for the project is the possibility to export data back into production systems like PsiacOnline and the labs' LIS software. Initial results were easily achieved by retrieving data from the wiki using ASK or SPARQL via JSON and XML interfaces. This results in structured data which is then synchronized with the data in production systems (see Fig. 3).

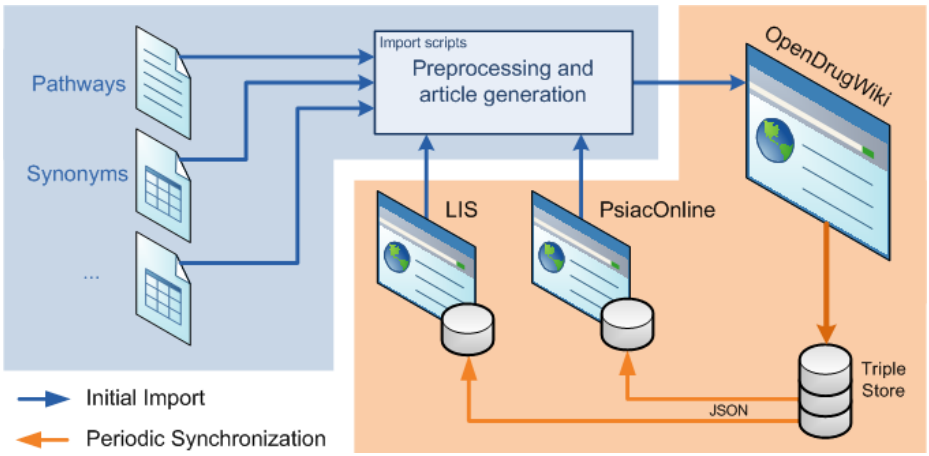


Fig. 2. Graphical overview of the import and export process

5 Conclusion and Prospects

With the approach presented in this article we wanted to show that a semantic wiki is an appropriate tool for consolidation of data with heterogeneous structure, sources and quality. The next step in this ongoing research project is the evaluation of the wiki's suitability to support continuous editing and reviewing processes in the different organizations, especially from a usability standpoint. We are anticipating good results, as Semantic Forms provides an easy-to-use interface for most purposes.

Since Semantic MediaWiki by default only provides semantic data for the latest revision of an article there is currently no easy way for integrating review processes. As we are preparing to open the wiki up to more and more research organizations for editing and contributing information on drugs and drug interactions, being able to have a reviewed and officially approved state of an article is currently the most important missing feature.

We would like to see the MediaWiki extension Flagged Revisions[14] integrated with Semantic MediaWiki, since this would help us to implement review processes and subsequently have reviewed and approved semantic data available in the triple store.

A benefit wikis and semantic web technologies offer is the possibility to create a multilingual information system by using interwiki links and semantic relations. A future task will be the creation of multiple wikis that will allow us to connect terms, drug names and drug interactions in different languages. Mapping classes and properties to standard drug and biomedical ontologies is therefore an important task either. In the near future we will integrate more data from new sources as they become available and begin connecting other production systems to the wiki to provide an efficient tool for researchers for editing drug data in one place.

[14] http://www.mediawiki.org/wiki/Extension:FlaggedRevs

6 References

1. Köstlbacher, A., Hiemke C., Haen E., Eckermann, G., Dobmaier, M., Hammwöhner R., PsiacOnline - Fachdatenbank für Arzneimittelwechselwirkungen in der psychiatrischen Pharmakotherapie. In: Osswald, A., Stempfhuber, M., Wolff, C. (Hrsg.). Open Innovation. Proc. 10 Internationales Symposium für Informationswissenschaft. Konstanz: UVK, 321-326. (2007)
2. Köstlbacher, A.: Information Management In A Neurochemical Laboratory, In: Kuhlen, R.r (Hrsg.) (2009). Information: Droge, Ware oder Commons? Proc. 11 Internationales Symposium für Informationswissenschaft. Konstanz: UVK, 567-570.
3. Leuf, B., Cunningham, W.: The Wiki Way: Quick Collaboration on the Web. Addison-Wesley, New York, 2001.
4. Hoffmann, R.: A wiki for the life sciences where authorship matters. Nature Genetics, Vol.: 40/9: 1047-1051 (2008)
5. Baumeister, J., Reutelshoefer, J., Nadrowski, K., Misok, A.: Using Knowledge Wikis to Support Scientific Communities. In: Proceedings of 1st Workshop on Scientifc Communities of Practice (SCOOP), Bremen, Germany (2007).
6. Schaffert, S.: IkeWiki: A semantic wiki for collaborative knowledge management. In: Tolksdorf, R., Paslaru Bontas, E., Schild, K., editors, 1st Int. Workshop on Semantic Technologies in Collaborative Applications (STICA'06) Manchester, UK, (2006)
7. Krötzsch, M., Vrandečić, D., Völkel, M.: Semantic MediaWiki. In: Proceedings of the 5th International Semantic Web Conference (ISWC'06). LNCS, vol. 4273, pp. 935-942. Springer, Heidelberg (2006)
8. Friedland, N.S., Allen, P.G., Matthews, G., Witbrock, M., Baxter, D., Curtis, J., Shepard, B., Miraglia, P., Angele, J., Staab, S., Moench, E. Oppermann, H., Wenke, D. Israel, D. Chaudhri, V., Porter, B., Barker, K., Fan, J. Chaw, S.Y., Yeh, P., Tecuci, D.: Project halo: towards a digital Aristotle, AI Magazine, Winter 2004, (2004)
9. Auer S., Dietzold S., Riechert, T.: OntoWiki – A tool for social, semantic collaboration. In Yolanda Gil, Enrico Motta, Richard V. Benjamins, and Mark Musen, editors, Proc. 5th Int. Semantic Web Conference (ISWC'05), number 4273 in LNCS, pages 736–749. Springer, (2006)
10. Kuhn, T.: AceWiki: A Natural and Expressive Semantic Wiki. In: Semantic Web User Interaction at CHI 2008: Exploring HCI Challenges (2008)
11. MediaWiki contributors: MediaWiki, The Free Wiki Engine, http://www.Mediawiki.org/w/index.php?title=Mediawiki&oldid=65192 (accessed March 3, 2010)
12. Köstlbacher, A., Maurus, J.: Semantische Wikis für das Wissensmanagement. Reif für den praktischen Einsatz? In: DGI e.V./M. Heckner, C. Wolff (Ed.). Information Wissenschaft und Praxis. Mai/Juni 2009, Dinges & Frick GmbH, Wiesbaden, pp. 225-231 (2009)
13. Ontoprise GmbH (Ed.): Basic Triplestore, http://smwforum.ontoprise.com/smwforum/index.php/Help:Basic_Triplestore (accessed March 3, 2010)
14. McBride, B.: Jena: A Semantic Web Toolkit. In: IEEE Internet Computing November/December 2002, pp. 55-59. (2002)
15. Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF. W3C Recommendation 15 January 2008. http://www.w3.org/TR/rdf-sparql-query/ (accessed March 3, 2010)

Enhancing MediaWiki Talk pages with Semantics for Better Coordination*

A Proposal

Jodi Schneider, Alexandre Passant, John G. Breslin**

Digital Enterprise Research Institute,
National University of Ireland, Galway
firstname.lastname@deri.org

Abstract. This paper presents a 15-item classification for MediaWiki Talk pages comments, associated with a new lightweight ontology that extends SIOC to represent these categories. We discuss how this ontology can enhance MediaWiki Talk pages, with RDFa, making content of such pages easier to parse and to understand.

Key words: MediaWiki, Wikipedia, Talk pages, RDFa, SIOC

1 Introduction

Wikis are often used for collaborative knowledge gathering and sharing, and coordination of this work may take place on and off the wiki (e.g. [8]). However, finding relevant conversations may become more difficult as their volume increases.

MediaWiki software[1], used by Wikipedia, Wikia[2], and other wikis, is one of the most popular systems, and we focus on it throughout the paper. Article-level coordination is common in MediaWiki; by default, MediaWiki installations provide a Talk namespace. Each article links to a Talk page (originally empty), which can be used to coordinate, discuss, and dispute the editing of that article. Figure 1 shows a sample Talk page. Talk pages are heavily used (as we discuss in Section 2.1), and some improvements to Talk pages have already been made available as MediaWiki plugins[3,4]. We believe that Talk pages could benefit from increased semantics.

As Talk pages grow, MediaWiki editors may benefit from tools to help identify relevant comments. We provide sample RDFa markup for MediaWiki Talk

* The work presented in this paper has been funded in part by Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Líon-2).

** John G. Breslin is also member of the School of Engineering and Informatics, NUI Galway

[1] http://www.mediawiki.org/

[2] http://www.wikia.com/

[3] http://www.mediawiki.org/wiki/Extension:LiquidThreads

[4] http://www.mediawiki.org/wiki/Category:Discussion_and_forum_extensions

Fig. 1. Talk page for the Semantic Web article in Wikipedia

pages, using a lightweight ontology for Talk page comments which extends SIOC [2]. This markup and ontology provide underlying metadata which could later be used to highlight and query for certain types of Talk page comments.

In the remainder of the paper, we first review related work, then describe 15 categories used to classify comments on MediaWiki Talk pages. Next we distill that classification system to a lightweight ontology for relevant Talk page comments, which we use to markup a Talk page segment in RDFa. Finally we outline work in progress on leveraging this ontology with RDFa markup and JavaScript- and SPARQL-based tools.

2 Related Work

2.1 Talk pages are heavily edited on Wikia and Wikipedia

Based on their studies of Wikia, Aniket & Kittur postulate that article talk scales linearly with the size of the wiki [5]. They compare coordination and Talk pages of Wikipedia and over 6000 Wikia wikis, finding differences which they attribute to differences in community size and type.

Wikipedia's Talk pages are heavily used, and in recent years, Talk pages have been added more quickly than articles, growing at a rate of 11x, compared to 9x for articles [11]. Over a 2.5 year period, edits to Wikipedia Talk pages nearly doubled, from 11% to 19% of all page edits, while article edits nearly halved

from 53% to 28% of all page edits [10]. Further, Wikipedia's users make a larger or smaller percentage of edits to Talk pages depending on their social roles [12].

2.2 Studies of Wikipedia Talk pages

While Wikipedia Talk pages have been studied from a content analysis, communications theory, and data mining perspective, further research is needed because the variance between Talk pages is significant. For instance, the most common type of discussion, coordination requests (described in Section 3 below), ranges widely, from 2% to 97% of the comments on a page, depending on the page [11]. Due to the variance, perhaps it is not surprising that researchers do not agree on the second most common type of discussion [3][11]. However, despite the evident variance, few categorical differences between Talk pages have been identified or systematically described. Furthermore, sample sizes for qualitative studies have been small (see [10] for a comparison of Featured and non-Featured articles with the largest sample size, 60 Talk pages). Other studies of Talk pages include [6], [4], [1], and [3].

Viégas [11] provides both a manual classification of 25 hand-selected Talk pages, and a quantitative analysis, which reveals that articles with Talk pages are more highly edited, and have more editors than articles without Talk pages. In particular,"94% of the pages with more than 100 edits have related Talk pages". The dimensions used in their manual classification are further discussed in Section 3, where they form the basis for our lightweight ontology.

3 Classifying comments in Wikipedia

Our classification began organically from the items in Talk pages we reviewed for our content analysis [9]. These coalesced into a set of classifications, which we then compared with the classification frameworks used in [11] and [10]. Since we planned to develop an ontology for editors to apply to their own comments, the directness of Viégas' classifications suited us, especially since these had already been used for at least two studies, and were very similar to our own classification. By contrast, since Stvilia classifies the possible information quality problems of an article, his classifications (such as cohesiveness and verifiability) require more abstraction, since they describe attributes of the article, not of the comment; further, some terms, (such as semantic consistency and security) might not be instantly accessible to the lay reader and wiki editor.

To update and extend Viégas' analysis [11], we undertook a manual content analysis [9] of Talk page comments, based on 100 Talk pages from five different types of Wikipedia Talk pages. Our content analysis used 15 non-mutually-exclusive classifications. First, we used the 11 classifications defined by Viégas [11]; Table 1 shows definitions of each term, with examples taken from Wikipedia Talk pages that we analyzed. To capture other features we were interested in, we added 4 new, non-mutually-exclusive classifications as shown in Table 2.

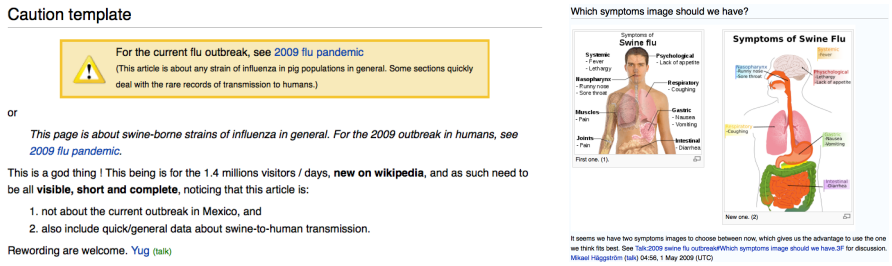
We added these types because:

Classification	Definition	Example
Requests/suggestions for editing coordination	Ideas, comments, or suggestions involving editing the article.	Currently some of the refs are YYYY-MM-DD format and some are Month DD, YYYY. Which format do we want to standardize to?
Requests for information	Questions asked by someone who doesn't intend to edit the page.	Where is Ligurian spoken in the Var ?
References to vandalism	Mentions of vandalism.	I've semi-protected the article for another week, the signal-to-noise ratio of the IP edits seemed too low.
References to wiki guidelines and policies	References to guidelines and/or policies of this wiki.	The section I removed had no sources / references - if you have sources they're no good being kept a secret ;) WP:VERIFY, WP:CITE. Thanks/
References to internal wiki resources	References to internal wiki resources such as diffs, Talk page discussions, old version of a page.	Would it be a good thing to re-add the links that were taken off in August? Somebody made them into a template that was subsequently deleted. The edit to recover the old links is here: [6]
Off-topic remarks	Remarks not relating to editing the article.	PLATO IS THE BEST MAN ALIVE! LONG LIVE PLATO
Polls	Formal proposals followed by statements such as Support and Oppose, with justifications.	A month should be deleted from the "Deaths in [CURRENT YEAR]" page ONE WEEK after the month ends...
Requests for peer review	Requests for peer review.	Users hoping to elevate articles to featured status may solicit a peer review.[11]
Information boxes	Special boxes with information, usually found at the top of a Talk page.	See Fig. 2(a), which proposes and discusses a new info box for the Swine influenza article.
Images	Images posted on the Talk page.	See Fig. 2(b)
Other	The sole exclusive category, describes items that don't fit elsewhere.	"This review is transcluded from Talk:Wiki/GA1. The edit link for this section can be used to add comments to the review."

Table 1. Viégas' 11 types of Talk pages comments [11]

Classification	Definition	Example
References to sources outside the wiki	References to sources, including print and deep web resources, outside this wiki.	Exclusive! Mighty Stef records football protest song"Hot Press. Not sure where to put it but I'll leave it here as somebody might find it useful...
References to reverts, removed material, or controversial edits	Discussions of reverts, removing material, or controversial edits.	I noticed some people edit the page into what it will be in 10 minutes but someone is reverting it...just let it be.
Reference to edits the discussant made	Applied when an editor discusses his/her own article edits on the Talk page.	Added the About.com review since the review was part of the reception section.
Requests for help with another article, portal, etc.	Solicitations for assistance elsewhere, or recruiting editorial help in the Talk page for another article.	This is just to invite attention to the page Facebook statistics just created; of all interested editors. I have just placed a mergeto tag in it. Thanks.

Table 2. Our 4 additional comment types for Talk pages



(a) (b)

Fig. 2. Comments from the Swine influenza Talk page containing: (a) a proposed infobox and, (b) images.

- Sources are heavily discussed in Talk pages, and some comments seem to be made soley to deposit a source. While many sources are on the open web (and can be detected as external links), print resources, inexact references, and deep web resources may also be provided.
- Disagreements about article content often take place in the context of reverts to the page. Discussions about removing content or editing controversial material may also take place on the Talk page before the article is edited.
- The Talk page may be used to notify other editors about a recent edit, perhaps to provide further description, anticipate questions, or clarify that a

suggestion has been implemented. Editors may also explain their own edits in discussions of reverts and edit wars.
– The Talk page is often seen as a site for communication with editors who have interest in or knowledge about a given topic. Requests for help, like Requests for information, draw on that perceived expertise.

4 A model for structuring wiki contributions

Based on the aforementioned 15 categories (11 from previous work plus the 4 that we introduced), we designed a lightweight vocabulary for annotating Talk pages. The main purpose of this model is to categorize each comment in the wiki page, so that, for example, one could immediately identify all the references to vandalism, all the pages requiring help, or all the sources recommended on the Talk page. This could be useful since editors may specialize, performing a certain type of task repeatedly [12]. Categorization could also facilitate automatically collating comments, for instance transcluding Requests for Information into a more appropriate spot, such as the Wikipedia Reference Desk[5] for that category. To that end, we provide a model (applied to a Talk page in Fig. 3):

– using existing ontologies, namely FOAF and SIOC, to model the users, the discussion topics (considered as SIOC threads), and the comments. Among others, we reuse the `sioct:WikiArticle` class from the SIOC Types module and the `sioc:has_discussion` property that was introduced by some of our previous work regarding modeling wiki structure using semantics [7].
– providing new classes to represent some of the classifications introduced in Section 3. We focused only on the requests and reference categories, for two reasons. First, these are the ones that people might indicate when they add new content (we will describe the process later). It is hard to imagine that someone would mark their own comment as off-topic; however, labeling it a "request for help" seems plausible. Second, these categories seem to be the most relevant for querying and retrieving information.

In addition, additional RDF properties could be used, e.g. from the Dublin Code vocabulary. For instance, when making a `ReferenceToEdit`, specifying a permalink to the edit could be done with `dcterms:requires`, or when making a `ReferenceToSources`, specifying the URI of a source with `dcterms:references`.

Our model, available at `http://rdfs.org/sioc/wikitalk`, then consists of:

– A class `WikiDiscussionItem`.
– Two classes, subclasses of the aforementioned one, named `ReferenceItem` and `RequestItem`, for references and requests, respectively, that have various subclasses as follows:
  • For the `ReferenceItem` class:
    ◦ `ReferenceToEdit`;
    ◦ `ReferenceToGuidelinesOrPolicies`;

[5] `http://en.wikipedia.org/wiki/Wikipedia:Reference_desk`

- `ReferenceToInternalResources`;
- `ReferenceToRevertsOrControversialOrRemovedMaterial`;
- `ReferenceToSources`;
- `ReferenceToVandalism`.

- For the `RequestItem` class:
  - `RequestEditingCoordination`;
  - `RequestHelpElsewhere`;
  - `RequestInfo`;
  - `RequestPeer-review`

Fig. 3. Annotated Talk page

5 Providing and using the annotations

5.1 RDFa Markup

Using this model, we then describe the type(s) of each comment, and the structural connections between these comments in MediaWiki Talk pages using RDFa markup. Here is an example before adding the markup (Listing 1.1), and after (Listing 1.2). The extracted RDF is also provided in Listing 1.3.

```
<h2>
<span class="editsection">[<a href="/w/index.php?title=Talk:Semantic_Web
    &amp;action=edit&amp;section=2" title="Edit section: Opening
    sentence">edit</a>]</span>
<span class="mw-headline" id="Opening_sentence">Opening sentence</span>
</h2>
<p>Could somebody please put examples of 'semantic web' immediately
    after the opening sentence? Otherwise it just sounds a bit waffly
    and, more importantly, the intelligent lay reader is lost. Thanks.
<a href="/wiki/Special:Contributions/86.42.96.251" title="Special:
    Contributions/86.42.96.251">86.42.96.251</a> (<a href="/wiki/
    User_talk:86.42.96.251" title="User talk:86.42.96.251">talk</a>)
    10:38, 30 March 2009 (UTC)
</p>
```

Listing 1.1. Example of a comment in a Talk page

```
<div xmlns:sioc="http://rdfs.org/sioc/ns#" xmlns:siocwt="http://rdfs.org
    /sioc/wikitalk#" xmlns:content="␣http://purl.org/rss/1.0/modules/
    content/" about="#Opening_sentence" typeof="sioc:Thread" rel="
    sioc:has_container" href="/w/index.php?title=Talk:Semantic_Web">
<h2>
<span class="editsection">[<a href="/w/index.php?title=Talk:Semantic_Web
    &amp;action=edit&amp;section=2" title="Edit␣section:␣Opening␣
    sentence">edit</a>]</span>
<span class="mw-headline" id="Opening_sentence">Opening sentence</span>
</h2>
<p about="#post_1" id="#post_1" typeof="
    siocwt:RequestEditingCoordination" rel="sioc:has_container" href="#
    Opening_sentence" property="content:encoded">Could somebody please
    put examples of 'semantic␣web' immediately after the opening
    sentence? Otherwise it just sounds a bit waffly and, more
    importantly, the intelligent lay reader is lost. Thanks.
<a href="/wiki/Special:Contributions/86.42.96.251" title="
    Special:Contributions/86.42.96.251">86.42.96.251</a> (<a href="/wiki
    /User_talk:86.42.96.251" title="User␣talk:86.42.96.251">talk</a>) 10
    :38, 30 March 2009 (UTC)
</p>
</div>
```

Listing 1.2. Example of a comment in a Talk page, with RDFa markup

```
<#post_1> a siocwt:RequestEditingCoordination ;
  content:encoded """Could somebody please put examples of 'semantic web
      ' immediately after the opening sentence? Otherwise it just sounds
       a bit waffly and, more importantly, the intelligent lay reader is
       lost. Thanks.
    <a href="/wiki/Special:Contributions/86.42.96.251" title="Special:
        Contributions/86.42.96.251">86.42.96.251</a> (<a href="/wiki/
        User_talk:86.42.96.251" title="User talk:86.42.96.251">talk</a>)
         10:38, 30 March 2009 (UTC)
"""^^rdf:XMLLiteral ;
  sioc:has_container <#Opening_sentence> .

<#Opening_sentence> a sioc:Thread ;
  sioc:has_container </w/index.php?title=Talk:Semantic_Web> .
```

Listing 1.3. Example of a comment in a Talk page, in Turtle (without prefixes)

5.2 Annotation and extraction tools

We are currently developing several services to provide and use the aforementioned annotations. First, we are creating two JavaScript plugins, an annotation plugin and a highlight plugin. Then, we will also investigate the use of SPARQL-based interfaces to query such annotations.

While editing the Talk page, an editor could use a JavaScript-based annotation plugin to specify which of the 10 classifications of our ontology apply. (Users do say that they are willing to choose the comment type.) The plugin would then generate the applicable RDFa markup. The annotation plugin could also get certain FOAF and SIOC attributes from the username or IP address. The annotation plugin will also facilitate user testing with the Wikipedia community, which may lead to further refinement of the Wikitalk module and its class labels, based on task-based evaluations with frequent wiki editors and other user testing of the annotation process.

So far we have created a plugin to use such annotations; relying on the RDFa markup, it uses a JavaScript RDFa parser[6] to parse a Talk page and to highlight relevant comments on a single Talk page, based on an ontology category to which they belong. We are currently evaluating this plugin and making improvements based on user feedback.

A third application, based on SPARQL, will allow querying to get "views" on the top of MediaWiki pages. For example, the user could "find all references to vandalism posted in the last 2 days" or "find all comments mentioning a source outside Wikipedia". SPARQL also opens up exciting possibilities, such as automatically collating comments, for instance transcluding Requests for Information into a more appropriate spot, such as (for Wikipedia) the Reference Desk for that topic, thus enabling new ways to automatically gather particular kind of comments, and facilitating the coordination process in MediaWiki instances.

6 Conclusion

Talk pages, as we have seen, are highly used, making it challenging to find relevant comments. To help fill this need, we used a 15-item classification for MediaWiki Talk page comments, extended from Viégas, and then developed a new lightweight ontology extending SIOC to represent the relevant categories. We then enhanced MediaWiki Talk pages with RDFa markup to indicate comment types and structural elements. That markup can in ongoing and future work be extracted with JavaScript and SPARQL, making the content of such pages easier to parse and to understand.

While the classifications in Tables 1 and 2 suit our immediate purpose, other alternatives are possible. Different classifications aiming towards a different ontology might focus more narrowly on the changes suggested (or indicated as made) by each comment (see, e.g. Table 3 in Stvilia [10]). Alternately, an ontology dedicated to a particular wiki could be based on information quality

[6] `http://www.w3.org/2001/sw/BestPractices/HTML/rdfa-bookmarklet/`

dimensions and editorial policies specific to that wiki. As our work progresses, we will be guided by user evaluations, to discover which such approaches might be beneficial for editors collaborating in wiki spaces.

References

1. Nicolas Bencherki and Jeanne d'Arc Uwatowenimana. Writing a Wikipedia article: Data mining and organizational communication to explain the practices by which contributors maintain the article's coherence. In *Annual Meeting of the International Communication Association*, Montreal, Quebec, May 2008.
2. John G. Breslin, Andreas Harth, Uldis Bojars, and Stefan Decker. Towards Semantically-Interlinked Online Communities. In *The Semantic Web: Research and Applications, Proceedings of the 2nd European Semantic Web Conference (ESWC '05)*, number 3532 in LNCS, pages 500–514. Heraklion, Greece, 2005.
3. Katherine Ehmann, Andrew Large, and Jamshid Beheshti. Collaboration in context: Comparing article evolution among subject disciplines in Wikipedia. *First Monday*, 13(10), October 2008.
4. Sean Hansen, Nicholas Berente, and Kalle Lyytinen. Wikipedia as rational discourse: An illustration of the emancipatory potential of information systems. In *40th Annual Hawaii International Conference on System Sciences*, 2007.
5. Aniket Kittur and Robert E. Kraut. Beyond Wikipedia: Coordination and conflict in online production groups. In *CSCW 2010*. ACM, February 2010.
6. Travis Kriplean, Ivan Beschastnikh, David W. McDonald, and Scott A. Golder. Community, consensus, coercion, control: cs*w or how policy mediates mass participation. In *Proceedings of the 2007 International ACM Conference on Supporting Group Work*, pages 167–176, Sanibel Island, Florida, 2007. ACM.
7. Fabrizio Orlandi and Alexandre Passant. Enabling cross-wikis integration by extending the SIOC ontology. In *Proceedings of the Fourth Semantic Wiki Workshop (SemWiki 2009), co-located with 6th European Semantic Web Conference (ESWC 2009)*, volume 464, Hersonissos, Heraklion, Crete, Greece, June 2009.
8. Christian Pentzold and Sebastian Seidenglanz. Foucault@Wiki first steps towards a conceptual framework for the analysis of wiki discourses. In *WikiSym '06: Proceedings of the 2006 International Symposium on Wikis*, 2006.
9. Jodi Schneider, Alexandre Passant, and John G. Breslin. A content analysis: How Wikipedia talk pages are used. In *WebScience 2010*, Raleigh, North Carolina, April 2010. http://websci10.org/.
10. Besiki Stvilia, Michael B. Twidale, Linda C. Smith, and Les Gasser. Information quality work organization in Wikipedia. *Journal of the American Society for Information Science and Technology*, 59(6):983–1001, 2008.
11. Fernanda B. Viégas, Martin Wattenberg, Jesse Kriss, and Frank van Ham. Talk before you type: Coordination in Wikipedia. In *40th Annual Hawaii International Conference on System Sciences*, pages 78–87, 2007.
12. Howard T. Welser, Dan Cosley, Gueorgi Kossinets, Austin Lin, Fedor Dokshin, Geri Gay, and Marc Smith. Finding social roles in Wikipedia. In *Proceedings of the American Sociological Association 2008*, Boston, MA, 2008.

Semantic Wiki Refactoring. A Strategy to Assist Semantic Wiki Evolution

Martin Rosenfeld, Alejandro Fernández and Alicia Díaz *

LIFIA, Facultad de Informática,
Universidad Nacional de La Plata, Argentina
{martin.rosenfeld,alejandro.fernandez,alicia.diaz}@lifia.info.unlp.edu.ar

Abstract. The content and structure of a wiki evolve as a result of the collaborative effort of the wiki users. In semantic wikis, this also results in the evolution of the ontology that is implicitly expressed through the semantic annotations. Without proper guidance, the semantic wiki can evolve in a chaotic manner resulting in quality problems in the underlying ontology, e.g. inconsistencies. As the wiki grows in size, the detection and solution of quality problems become more difficult. We propose an approach to detect quality problems in semantic wikis and assist users in the process of solving them. Our approach is inspired by the key principles of software refactoring, namely the cataloging and automated detection of quality problems (bad smells), and the application of quality improvement transformations (refactorings). In this paper we discuss the problem of evolving semantic wikis, present the core model of our approach, and introduce an extensible catalog of semantic wiki bad smells and an extensible toolkit of semantic wiki refactorings.

1 Introduction

Semantic Wikis [1] enhance the functionality of wikis with mechanisms to express semantic for the content in the wiki, in the form of *semantic annotations*. The synthesis of all semantic annotations in a semantic wiki defines its *ontology*.

Wikis evolve as a result of the collaborative effort of its users [2]. However, the uncoordinated edits of users may result in semantic wikis with poor quality regarding the underlying ontology. Thus, assistance needs to be provided to check that quality criteria are met and to minimize the effort of improvement.

In software engineering, refactoring techniques are applied to improve programs quality. The term refactoring [3] refers to the process of making persistent and incremental changes to a system's internal structure without changing its external behavior, yet improving the quality of its design. Refactoring is based on two key concepts: *bad smells*, that are an informal still useful characterization of patterns of bad source code, and *refactorings*, which are piecemeal transformations of source code that keep the semantics while removing (totally or partly) a bad smell.

* This work was partially funded by: the PAE 37279-PICT 02203 which is sponsored by the ANPCyT, Argentina.

A strategy for refactoring consists in the following components: a toolbox of refactorings, a catalog of bad smells, and detailed instructions on how to apply refactorings to remove each smell. Additionally, automated tools for the detection of bad smells and refactoring editors reduce the effort of refactoring and the chance of introducing errors. Environments in which refactoring is a mature practice (a culture), such as Smalltalk, usually offer these powerful tools.

Inspired by the principles of refactoring, this paper explores the use of such strategies to assist the evolution of semantic wikis and incrementally improve their quality. In this order, we give definitions for *Semantic Wiki Bad Smell* and *Semantic Wiki Refactoring*. Then, we adapt each of the components of a refactoring strategy to the context of semantic wikis. This includes a toolbox of semantic wiki bad smells, and a catalog of semantic wiki refactorings with their instructions on how to remove bad smells. Additionally, we discuss how tools can be used to automate the detection of bad smells and refactoring operations.

The structure of this paper is the following. We discuss in detail the problem of achieving quality in evolving semantic wikis in Section 2. Afterwards, in Section 3, we describe the state of the art in works to assist the evolution of semantic wikis. In Section 4, we present our approach that applies the ideas of software engineering refactoring in the context of semantic wikis. Finally, in section 5 we present the conclusions and future work.

2 The problem of evolving semantic wikis

The structure of a wiki, as well as its content, evolves as a result of the collaborative effort of the wiki users. Usually, wikis are created with very few or no structure in advance, so that users adjust it as a necessity to express knowledge, ideas, opinions, etc. Mader [4] suggests that the wiki creator should not try to guess the structure of a wiki in advance, but let it evolve into the optimal organization of information as people use it.

When the wiki is semantic, the wiki evolution also affects the formal representation of the wiki content, i.e. the underlying ontology. Thus, the ontology emerges with the wiki, as users add semantic annotations to the wiki content. Kousetti et. al. [5] use the term *Ontology Convergence* to mean the process of the implicit ontology evolving to a single model.

The problem of evolving wikis consists in assuring a good wiki quality through all the stages of the wiki evolution. If the evolution is not controlled, it is very possible that the wiki evolves in a chaotic manner. Many problems can arise as the result of multiple users collaboratively and incrementally editing a wiki. In traditional wikis, these problems are usually related to the following quality metrics: readability, structure, navegability, completeness, consistency.

Semantic wikis aim to improve the quality of traditional wikis by allowing to add semantic knowledge to the wiki content. However, they cannot assure that the wiki evolves in a correct manner and that a good ontology convergence is achieved. Kousetti et. al. [5] describe the following quality metrics for the semantic wiki's implicit ontology:

- Consistency: no sentence can be contradicted.
- Completeness: anything that needs to be in the ontology is explicitly defined or it can be inferred from other defined definitions and axioms.
- Conciseness: does not contain unnecessary definitions or redundancies.
- Expandability: you can easily add more knowledge without requiring to make major changes to the existing structure.
- Sensitiveness: the ontology is more sensitive if small changes can alter easily how well-defined a definition is.

Let's take a motivating example of a semantic wiki being employed to describe geographical places. In an early stage of evolution, a user creates a category "City". Later, another user creates a category "Capital city", attempting to form a more specific group for cities that are also capitals. However, this user does not realize that this category defines a subset of the resources of category "City" and, consequently, should be a subcategory of it. The lack of this semantic relation between the two categories results in many problems. First of all, it affects the conciseness of the semantic wiki. Every time a user creates a page for a city that is also a capital city, he will have to categorize it with both categories, so that no semantic knowledge is lost. Secondly, it affects the semantic wiki completeness. A user asking the wiki for a list of cities will not get as result the resources categorized with category "Capital City" but not with category "City".

The example above exposes that problems can arise while the evolution of semantic wikis takes place. Such problems are finally perceived as a lack of quality in the product, thus jeopardizing the success of the wiki. This results in the necessity of an extra work, that consists in periodically checking for problems in the wiki quality, and making the corresponding modifications. This work is commonly called "wiki gardening"[1]. However, as the wiki becomes longer, with many pages and semantic annotations, it becomes more difficult to detect quality problems and solve them appropiately.

In this context, where shared ownership and evolution of structure and content are a plus, assistance needs to be provided to check that quality criteria are met and to minimize the effort of improvement.

3 State of the art

Wikis are groupware. Peter and Trudy Johnson-Lenz [6] identify two prevailing approaches to groupware: *mechanism*, the use of explicit forms and procedures, and *context* or *open space*, to allow groups to self organize. Wikis are by nature closer to the later approach. There are semantic wiki tools, such as *Project Halo*[2], that support the definition of the ontology upfront (thus stressing the "mechanism"' approach). Those wikis are usually called *Semantic Data Wikis*. In contrast, our research focuses on wikis created with very few or no structure in advance, so that users incrementally create the ontology.

[1] http://www.wikisym.org/ws2008/index.php/What_are_the_tasks_of_a_wiki_gardener%3F
[2] http://www.mediawiki.org/wiki/Extension:Halo_Extension

In the field of ontologies, the problem of ontology evolution is one of the current topics of the research agenda. Djedidi et. al. [7] and Haase et. al. [8] focus on consistency achievement when a change in the ontology is performed. They employ resolution mechanisms to recover from four levels of inconsistencies: structural, logical, conceptual and domain dependent. Moreover, the former presents a list of metrics to evaluate the quality of an ontology. Baumeister and Seipel [9] describe a set of anomalies in the design of the ontology that may affect its mantainability, usability and understandability, and propose refactoring methods to eliminate them. They distinguish four categories of anomalies: redundancy, circularity, inconsistency and deficiency. Although these approaches make significant contributions to the field of ontologies, the domain of semantic wikis presents new challenges. Firstly, the collaborative way of editing a wiki increments the possibilities of generation of different kind of anomalies. Secondly, the combination of semantic annotations with wiki's tacit knowledge (i.e. text, images, etc.) has to be considered.

An approach to assist the evolution of semantic wikis is MOCA [5], which is a Semantic Mediawiki (SMW) [10] extension. It is a support system that assists wiki authoring and contribution to the background ontology. MOCA provides assistance in the edit page of the wiki, giving help with recommendations for types using the background ontology, and insertion of annotations without knowledge of the syntax. A similar approach is taken by the *Project Halo*, which is also an extension of SMW that provides intuitive graphical interfaces that facilitate the authoring, retrieval, navigation and organization of semantic data.

Although these two works assist the evolution of semantic wikis by helping users in the creation of content, they do not aim to neither find quality problems nor solve them. SMW+[3] is actually an approach to accomplish part of this. SMW+ is a semantic wiki built on top of SMW, aimed at the enterprise market. In addition to assisting users in authoring using the Halo Extension, SMW+ offers gardening functionality. It comprises a set of programmable tasks (called bots) to automate or assist users in the process of improving the quality of the wiki content. For example, there are bots to find pages without annotations, or to find artifacts on schema or annotation level which indicate a bad modeling.

Although the contribution of the gardening functionality of SMW+ should be recognized, it is strongly biased towards the identification of quality problems. It does not provide support for the implementation of the changes required to remove them and, thus, to improve the quality of the semantic wiki.

4 Refactoring semantic wikis

In this section we will see how each of the components of a refactoring strategy, that were described in section 1, are applied to the context of semantic wikis. But first of all, in section 4.1, we will need to define the model of the artifact that will serve as the base to define each of them.

[3] http://wiki.ontoprise.de/smwforum/index.php/MainPage

4.1 Core model

In order to define refactorings and bad smells, we need a model of a semantic wiki that will serve as a base. We call it the *core model*, which is shown in figure 1. It is defined as an OWL ontology, and we use UML diagrams to visualize it.

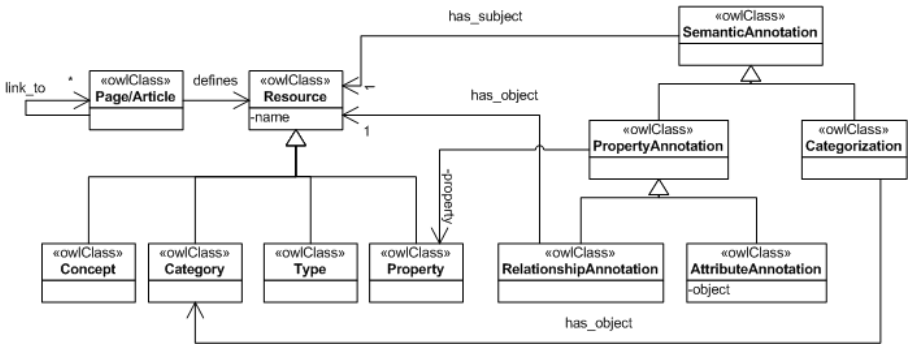


Fig. 1: Core Model: a conceptual model of a semantic wiki

The core model is a conceptual model of a semantic wiki and is inspired on the knowledge base of SMW. However, it can be customized and extended to support new features, so that new refactorings and smells could be defined in terms of them.

A *Resource* is anything that can be described in a wiki: *Concept*, that represents individuals in the wiki; *Category*, which allows to group resources; *Property*, that describes the semantic of relationships between resources; and *Type*, that is used to specify the object type of properties. Each of the resources can have a wiki *Page* or *Article* that describes it. Pages can be also directly related to other pages using untyped links.

Semantic Annotation is subclassified in Categorization and Property Annotation. A *Categorization* describes a relation of membership of a resource in a category. A *subcategory relationship* between categories is defined when the subject resource of a categorization is a category. In this case, the subject category is a *subcategory* of the object category.

A *Property Annotation* is composed by three elements: a *subject*, that can be any resource; a *property*, which is an individual of Property; and an *object*. *Relationship Annotations* are those property annotations whose object is another resource, while the object of *Attribute Annotations* are end values of a data type. Categorizations could be restated as relationship annotations in which the property describes the *belongs to* semantic, and the object is allways a category.

We assume the existence of a set of built-in properties with special meanings. One of them is the *Subproperty of* Property. When used as the property of a relationship annotation between two properties, it describes a subproperty rela-

tionship between the subject property and the object property. Another built-in property is the *Has type* Property that indicates the object Type of a Property.

4.2 A toolbox of semantic wiki refactorings

Semantic wiki refactorings are transformations in the semantic structure of a semantic wiki knowledge base, that allow to improve the implicit ontology quality. Refactorings describe an ordered way of appropiately performing such transformations, so that new inconsistencies, redundancies and other quality problems are avoided. Each refactoring is defined by: a *name*; a *description*, which summarizes the refactoring in informal language; and the *mechanics*, which are a series of mechanical transformations in the model of the wiki.

In the following we describe in detail the refactoring called *Create subcategory relationship* as an example. As it is not the objective of this paper to define a complete catalog of refactorings, we will later list other refactorings for which we will only mention their name and description.

Create subcategory relationship refactoring describes a transformation in which an existent category turns to be a subcategory of another one. Defining such relationship helps improve the quality of search results, the conciseness and completeness of the semantic wiki.

Following the example described in section 2, applying this refactoring would mean to make "Capital City" a subcategory of "City". The first step to appropiately accomplish this, must consist in actually making "Capital City" subcategory of "City". This is the basic step to create the subcategory relationship. Secondly, the categorizations of category "City" in each resource that also belongs to category "Capital City" have to be removed. This is because it is now implicit that every resource that belongs to "Capital City" also belongs to "City". Applying this step avoids redundant categorizations. The last step consists in eliminating redundancies in the categorization chain. Supose the existence of a category "Geographical place", that is supercategory of both categories. Then, the subcategory relationship that states that "Capital City" is subcategory of "Geographical place" is now redundant and, thus, has to be eliminated.

The following is the formalization of this refactoring. Applied to the example, the category "City" plays the role of *supercategory*, "Capital City" is the *subcategory*, and "Geographical place" is the *super-supercategory*:

Name Create subcategory relationship.
Description Make an existent category *subcategory* subcategory of another existent category *supercategory*.
Mechanics

1. Make *supercategory* supercategory of *subcategory*.
2. For each resource that belongs to *subcategory* remove its categorization of *supercategory*.
3. For each category *super-supercategory* that is supercategory of *subcategory*: if it is also a supercategory of *supercategory*, remove it as supercategory of *subcategory*.

Note that the steps defined in the mechanics of the refactoring are not bound to any semantic wiki implementation. The concrete actions that has to be done to actually perform the steps, depend on the way each semantic wiki implementation implements each of the semantic wiki features. In this way, refactorings are able to be reused in any semantic wiki implementation.

Table 1 presents a list of other possible semantic wiki refactorings. This list can be extended and customized. Furthermore, if the implementation of the semantic wiki supported new features, it could be extended to make use of them.

Name	**Description**
Move annotation	Change the subject of an annotation from one resource to another.
Unify categories	Unify two existent categories categoryA and categoryB. A new category categoryC is created and replace both of them. Existent categorizations of the two unified categories have to be replaced to the new one.
Split property	Split a property propertyA into two new properties propertyB and propertyC. The semantic annotations that have propertyA as property, have to appropiately change it to propertyB or propertyC.
Extract concept	A concept conceptA describes more than one real concept. A new concept conceptB is created. All the semantic annotations that have conceptA as subject or object, have to be placed appropiately.
Create subproperty relationship	Create a subproperty relationship between two existent properties.
Rename concept	Rename a concept and, consequently, the page that describes it. All the semantic annotations and untyped links that point to the concept, have to be updated.
Remove category	Remove an existent category. All its categorizations have to be also removed.

Table 1: List of semantic wiki refactorings

4.3 A catalog of semantic wiki bad smells

Refactorings describe *how* to appropiatelly perform transformations in the wiki. However, they say nothing about *when* they should be applied. Fowler [3] afirms that "deciding when to start refactoring, and when to stop, is just as important to refactoring as knowing how to operate the mechanics of a refactoring".

Defining bad smells may help determine when to apply each refactoring. A *semantic wiki bad smell* is a symptom in the semantic structure of a semantic wiki that possibly indicates a deeper problem, and suggests that a refactoring should be applied. It must be said that the detection of a bad smell does not necessarily implies a real problem. It must be analyzed in the current context and decide whether it should be applied or not a refactoring.

Each bad smell is defined with: a *name*; a *description*, which describes the bad smell in informal language; the *related refactorings* that should be applied

to remove the smell; and a *detection mechanism*. The detection mechanism may simply involve the execution of a query to the wiki knowledge base, or may apply more sophisticated methods such as data minning techniques.

As with refactorings, it is not the objective of this paper to define a complete catalog of bad smells. We will first describe as an example the *Twin categories* bad smell and then present a preliminar list of other possible ones.

The *Twin categories* bad smell describes the case when two categories appear together in categorizations very frequently. The first reason why this bad smell can be detected, is if two categories have the same semantic but different names. This case is intrinsic to the collaborative way of semantic wikis. This situation can arise because a user does not know a category already exists, or because of users belonging to different professional stuff or speaking different languages. The consequence of this problem is that a category is duplicated.

The second possible cause is that the twin categories define a subcategory relationship. The example presented in section 2 describes this situation. Because of the lack of the subcategory relationship between the categories "Capital City" and "City", every time a resource is categorized with category "Capital City", it should be also categorized with category "City". To remove the bad smell in this case, the "Create subcategory relationship" refactoring should be applied.

The following is the full definition of this bad smell:

Name Twin categories

Description Categorizations of two categories categoryA and categoryB appear together very frequently.

Related Refactorings

1. Unify categories
   Cause: The two categories describe the same category.
   Example: categoryA = "LIFIA", categoryB = "LIFIA Lab"
2. Create subcategory relationship
   Cause: The two categories describe a subcategory relationship.
   Example: categoryA = "Capital city", categoryB = "City"

Detection Mechanism In this case, we decided to use a semantic query as the detection mechanism. The semantic query is expressed in SPARQL query language, and is based on "SPARQL 1.1 Query"[4] specification.

```
SELECT DISTINCT ?categoryA ?categoryB
WHERE {  ?categorizationA has_subject ?subjectA
         ?categorizationB has_subject ?subjectA
         ?categorizationA has_object ?categoryA
         ?categorizationB has_object ?categoryB.
FILTER ((?categorizationA != ?categorizationB)
         && (?categoryA != ?categoryB))
      }
GROUP BY ?categoryA ?categoryB
HAVING (count(*) > PARAM)
```

Table 2 presents a list of other possible semantic wiki bad smells.

[4] http://www.w3.org/TR/2009/WD-sparql11-query-20091022/

Name	Description
Concept too categorized	A concept belongs to too many categories. This symptom may expose the fact that the concept describes more than one real concept.
Divergent property	Instances of a property diverge in range and domain. A property is used with many semantics.
Twin properties	Annotations of two properties appear together in the same resources very frequently.
Resource with no semantic annotations	A resource is not subject of any semantic annotation.
Large category	A category is object of too many categorizations.

Table 2: List of semantic wiki bad smells

4.4 Automating the detection of bad smells and refactoring operations

Automating the detection of smells is a necessity as the wiki grows in size and semantic knowledge. Defining a detection mechanism such as a semantic query for each smell, makes it possible to find all the occurrences of a bad smell automatically. Integrating a tool for the detection of bad smells in the semantic wiki would allow users to look for quality problems with less effort.

The mechanics of the refactorings describe an ordered way of applying transformations, consisting in a series of simple actions. However, it could become tedious and time-consuming to execute those actions by hand. Moreover, the chances of making mistakes and introducing new errors are increased. For wiki refactoring to be a productive strategy to assist semantic wiki evolution, effort and risk must be minimized.

Fortunately, many times it is possible to have a refactoring automation tool. Take for example the *Create subcategory relationship* refactoring. You have several changes to make and check for correctness if you do it by hand. With a refactoring automation tool, one simply selects both categories (subcategory and supercategory) and launches the refactoring. The refactoring tool applies all the steps of the mechanics as part of the same single transaction. The whole process takes seconds instead of several minutes. Moreover, the refactoring operation ensures that no action is lost and no bugs are introduced.

5 Conclusions and future work

Semantic wiki refactoring is an approach to detect quality problems in semantic wikis and assist users in the process of solving them. It is inspired by the key principles of software refactoring. We have discussed the problem of evolving semantic wikis, presented the core model of our approach, and introduced extensible catalogs of semantic wiki bad smells and refactorings.

As future work to complete and extend this investigation, it remains to complete the catalogs of semantic wiki bad smells and refactorings. A categorization

should be defined to achieve a better understanding. In other respects, the tools to automate the semantic wiki refactoring operations and semantic wiki detection of smells should be developed. It will allow to carry out experimental evaluations to check the effect of a refactoring strategy in the evolution of semantic wikis.

In other sense, as wikis have a strong social component, it can be investigated the social factor in a semantic wiki refactoring strategy. In this order, collaborative detection of smells and collaborative testing of refactorings should be studied. The first one arises as a necessity because some bad smells are difficult to be detected by automated detection mechanisms, e.g. a bad smell to detect inappropiate names for categories. We propose a social detection mechanism in which users could collaboratively agree the presence of a bad smell. The second takes the idea from software engineering agile methods. They advocate the use of automated unit testing to ensure that no bugs were introduced and the success of a refactoring. Adapting this idea, we propose a social testing in which wiki users could collaboratively state if a refactoring was successful or not.

Finally, future work will deal with refactoring of wiki's tacit knowledge. Tacit knowledge should be considered in the refactoring operations, and could also be the source of bad smells.

References

1. Schaffert, S., Bry, F., Baumeister, J., Kiesel, M.: Semantic wikis. IEEE Software **25**(4) (2008) 8–11
2. Leuf, B., Cunningham, W.: The Wiki way: quick collaboration on the Web. Addison-Wesley (2001)
3. Fowler, M.: Refactoring: Improving the Design of Existing Code. Addison-Wesley, Boston, MA, USA (1999)
4. Mader, S.: Wikipatterns. Wiley Publishing (2007)
5. Kousetti, C., Millard, D., Howard, Y.: A study of ontology convergence in a semantic wiki. In: WikiSym 2008. (2008)
6. Johnson-Lenz, P., Johnson-Lenz, T.: Post-mechanistic groupware primitives: rhythms, boundaries and containers. Int. J. Man-Mach. Stud. **34**(3) (1991) 395–417
7. Djedidi, R., Aufaure, M.A.: Onto-evoal an ontology evolution approach guided by pattern modeling and quality evaluation. In Link, S., Prade, H., eds.: FoIKS. Volume 5956 of Lecture Notes in Computer Science., Springer (2009) 286–305
8. Haase, P., Stojanovic, L.: Consistent evolution of owl ontologies. In: Proceedings of the Second European Semantic Web Conference, Heraklion, Greece. (2005)
9. Baumeister, J., Seipel, D.: Verification and refactoring of ontologies with rules. In: EKAW'06: Proceedings of the 15th International Conference on Knowledge Engineering and Knowledge Management. (2006) 82–95
10. Krötzsch, M., Vrandecic, D., Völkel, M., Haller, H., Studer, R.: Semantic wikipedia. Journal of Web Semantic **5**(4) (2007) 251–261

Using DSMW to build a Network of Semantic MediaWiki Servers

Hala Skaf-Molli, Gérôme Canals and Pascal Molli

Université de Lorraine, Nancy, LORIA – INRIA Nancy-Grand Est, France
{skaf, canals, molli}@loria.fr

Abstract. DSMW is an extension to Semantic Mediawiki (SMW), it allows to create a network of SMW servers that share common semantic wiki pages. DSMW users can create communication channels between servers and use a publish-subscribe approach to manage the change propagation. DSMW synchronizes concurrent updates of shared semantic pages to ensure their consistency. It offers new collaboration modes to semantic wiki users and supports dataflow-oriented processes.

1 Research Background: Collaborative Editing

Semantic wikis allow to create and edit collaboratively semantically annotated documents. However, compared with other collaborative systems, semantic wikis do not support offline work or multi-synchronous editing [1]. In existing semantic wikis, every change in a page is immediately visible for both end users and semantic engines. However, in many cases it is necessary to change multiple pages before making them visible. Existing semantic wikis cannot prevent users to access, navigate or query inconsistent pages. Moreover, the lack of multi-synchronous support prevents users to work isolated [2] and also prevents semantic wikis to support dataflow oriented processes.

To overcome these limitations, we propose a distributed approach for semantic wikis. In this approach, semantic wiki pages are replicated over a network of interconnected semantic wiki servers. Changes issued on one server are local but can be published to other servers. Remote servers can subscribe to these changes, pull them and integrate them to their local pages. Changes propagation remains under the control of the users. Concurrent changes on a page issued by different servers are handled by a merge procedure.

Using this approach, users can alternate between isolated periods of work and synchronization sequences with remote servers. They can introduce changes to multiple pages before to atomically render these changes public. They can choose when to incorporate, or not, remote changes. In addition, the approach can be the basis for implementing processes in which flows of semantic wiki pages can traverse a network of semantic wiki servers. Each wiki server can implement one or several steps of a particular process for the creation and maintenance of semantic pages.

2 DSMW approach

DSMW is an extension to Semantic MediaWiki (SMW) [3]. It allows to create a network of SMW servers that share common semantic wiki pages. DSMW manages the propagation and the integration of changes issued on one SMW server to remote servers on the network. The system ensures the consistency of the whole set of replicated pages.

DSMW users can create and edit semantically annotated wiki pages as with a regular SMW server. Then she/he can manage pages changes as a software developer does with her/his source code using a distributed version control system: she/he can work in isolation while editing pages and semantic annotation on a single server, then she/he can publish part or all of her own changes by pushing them to DSMW public feeds, and she/he can subscribe to any remote public DSMW feeds, pull changes from remote servers and integrate them to the local pages.

The DSMW extension adds two main features to SMW: an optimistic replication algorithm, and an ontology to manage changes, publication and integration of changes sets.

Page replication in DSMW is handled by a dedicated replication procedure. Since semantic annotations are embedded in page content in SMW, DSMW replicates only page contents and there is no need to deal with annotations. DSMW uses the Logoot algorithm to synchronize concurrent changes[4]. Logoot guarantees the consistency of the shared pages based on the CCI model (Causality, Convergence, Intentions [5], the model used also by Google Wave). The propagation technique is publish-subscribe: changes issued on one server can be published by pushing them to one or several *pushFeeds*. Remote servers can subscribe to these feeds by connecting *pullFeeds* to existing remote pushFeeds. Then, they can pull changes and integrate them to the local pages. Concurrent changes are merged by the Logoot algorithm. Hereafter a brief description of the operations related to replication.

Save: the SMW save operation, called when a user saves edit modifications on a page, has been extended to build and log patches. Patches represent changes to a page as a sequence of elementary Insert and Delete operations. Patches are computed by the Logoot algorithm as a diff between the current and the new version of the page being saved. Logoot uses a special index to determine absolute insertion and deletion positions of the elements in a page. Once computed, a patch is applied to the local page and logged.

CreatePushFeed: creates a named communication channel to publish local changes. The content of the feed is specified by a semantic query. All pages in the query result belongs to that channel, meaning that changes on these pages will be published through that feed. Note that a page can belong to several channels.

Push: the push operation computes the set of patches for a given pushFeed to form a ChangeSet. A changeSet is the ordered set of all patches logged for all the pages belonging to that pushFeed. Once computed, the changeSet is added to the feed and can then be pulled by remote servers.

CreatePullFeed: creates a named communication channel to pull remote changes. A pullFeed is connected to one single remote pushFeed, so a pull feed is defined by the URL of the remote server and the pushFeed name.

Pull: the push operation downloads all the pending changeSets published in the pushFeed connected to a given pullFeed. Patches extracted from the changeSets are then locally applied in the order they appear. To do so, Logoot uses the absolute positions computed during the patch creation to insert or delete page elements.

The DSMW ontology shown in the figure 1 represents all the concepts of DSMW: changes, change sets, push and pull feeds. This ontology makes possible the querying of the current state of the wiki and its complete history using SMW semantic queries. For instance, queries can extract *the list of unpublished changes* or *the list of published changes on a given channel*. This ontology is populated through the user interaction with the system: all the operations described in the previous paragraph create or delete instances of the DSMW ontology (see [6] for more details).

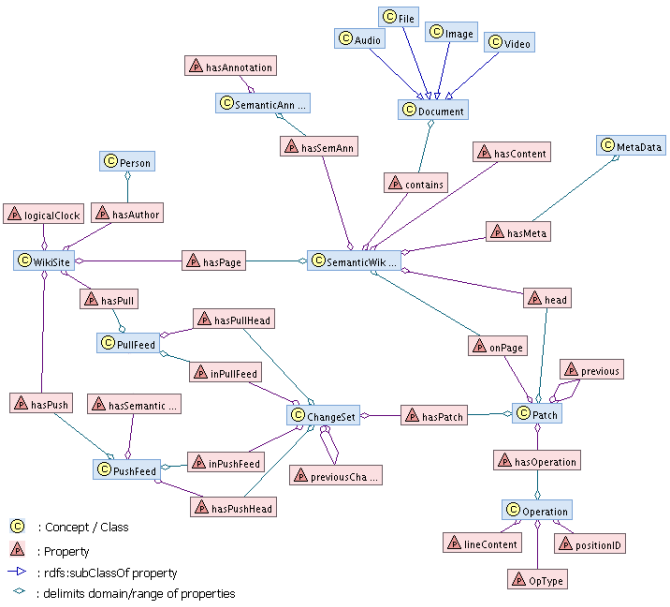


Fig. 1. Multi-synchronous ontology

3 Use cases and Applications

DSMW is used in ongoing French national projects: *WikiTaaable* and *CyWiki* *WikiTaaable* is a distributed collaborative knowledge building systems for cooking recipes [7]. It integrates a case-based reasoning engine. WikiTaaable uses SMW as a central module to manage all data and knowledge used in the system. DSMW supports the humans and machines collaboration by deploying several DSMW servers to implement continuous integration processes as those used during software development. WikiTaaable is accessible at `http://taaable.fr`.

The *CyWiki* project uses DSMW as an infrastructure for the collaborative and assisted transformation of textual content into formal and structured knowledge. The transformation process is a decentralized process in which both human agents and automatic agents (text-mining agents, classification agents) collaborate to build knowledge units (in the form of ontology elements). This knowledge can then be used to query and make reasoning about the content. The experimental and application domain of the project is education.

4 System Demonstration

The demonstration scenario will focus on three main use-cases:

The knowledge aggregation corresponds to the use of a DSMW server to aggregate and combine wiki pages and knowledge from multiple sources. This server subscribes to these sources by creating pull feeds connected to the public push feeds at each source. By combining these sources, the system can answer new queries that could not be evaluated on a single source. The demonstration example will be the following:

- a first DSMW server holds semantic wiki pages about hotels in a city. Hotels are described with various properties (rooms, prices ...) and their location in the city relatively to well-known places (e.g. the train station, the main square),
- a second DSMW server holds semantic wiki pages about touristic information in the city. It describes sites of interest and cultural events with various properties and their location in the city, relatively to well known places.
- a third DSMW server subscribes to the public push feeds of the two previous, and regularly pull them. It then holds semantic wiki pages on both hotels and touristic information and their location in the city, and maintain these pages consistent with the original sources. This server can answer queries that cannot be evaluated on the original sources, typically to find an hotel close to a particular site of interest.

The knowledge validation steps corresponds to the use of one or several DSMW servers to implement a validation process: prior to rendering public a set of semantic wiki pages, it can be desirable in some cases to validate their content by users or by running non-regression tests. The scenario will be based on the same hotel-tourist example. It consists in adding a fourth DSMW server

that will serve as a public front-end for querying the hotel-tourist knowledge base. The aggregation server will then serve to combine the original sources and validate the new knowledge base. This validation step is done by users verifying the semantic annotations and eventually modifying them and running tests by evaluating a fixed set of queries whose results are known and should not change. Once validated, changes are propagated to the fourth server and are thus accessible to end-users. This validation step ensures the consistency and the stability of the final knowledge base. Any change to the original sources is tested, verified and eventually fixed before to be queried by end-users.

The network construction use case corresponds to the construction of a network of interconnected DSMW server. The demonstration will show how the push and pull feeds are created on the different servers of the hotel-tourist example, and connected to create the network.

5 Conclusion

In this demonstration we have presented a new collaborative tool, called DSMW, to support multi-synchronous collaboration and dataflow processes over semantic wiki pages. DSMW is developed as an extension of SMW. The first public release release of DSMW was published in October 2009. A new release DSMW 0.5 is available at `http://dsmw.org`. We continue the development and the research on DSMW. Research concerns divergence awareness in DSMW and the analysis of the social networks built by the collaborative editing.

References

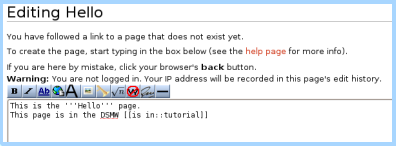
1. Dourish, P.: The parting of the ways: Divergence, data management and collaborative work. In: 4th European Conference on Computer Supported Cooperative Work. (1995)
2. Molli, P., Skaf-Molli, H., Bouthier, C.: State Treemap: an Awareness Widget for Multi-Synchronous Groupware. In: Seventh International Workshop on Groupware - CRIWG, IEEE Computer Society (2001)
3. Krötzsch, M., Vrandecic, D., Völkel, M., Haller, H., Studer, R.: Semantic wikipedia. Journal of Web Semantic **5**(4) (2007) 251–261
4. Weiss, S., Urso, P., Molli, P.: Logoot : a scalable optimistic replication algorithm for collaborative editing on p2p networks. In: International Conference on Distributed Computing Systems (ICDCS), IEEE (2009)
5. Sun, C., Jia, X., Zhang, Y., Yang, Y., Chen, D.: Achieving Convergence, Causality Preservation, and Intention Preservation in Real-Time Cooperative Editing Systems. ACM Transactions on Computer-Human Interaction **5**(1) (1998)
6. Rahhal, C., Skaf-Molli, H., Molli, P., Weiss, S.: Multi-synchronous collaborative semantic wikis. In: 10th International Conference on Web Information Systems Engineering - WISE '09. Volume 5802 of LNCS., Springer (2009) 115–129
7. Cordier, A., Lieber, J., Molli, P., Nauer, E., Skaf-Molli, H., Toussaint, Y.: Wikitaaable: A semantic wiki as a blackboard for a textual case-based reasoning system. In: SemWiki 2009 - 4rd Semantic Wiki Workshop at the 6th European Semantic Web Conference - ESWC 2009, Grce Heraklion (2009-05-16)

Distributed Semantic MediaWiki
http://dsmw.org

Hala Skaf-Molli, Gérôme Canals and Pascal Molli
Université de Lorraine, Nancy, LORIA
INRIA Nancy-Grand Est, France
{skaf, canals, molli}@loria.fr

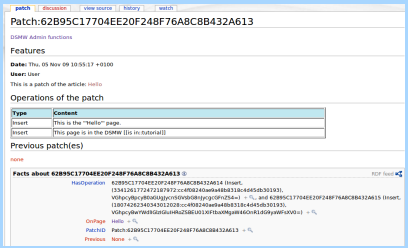
1 Abstract

DSMW is an extension to Semantic Mediawiki (SMW). It allows to create a network of SMW servers that share common semantic wiki pages. DSMW users can create communication channels between servers and use a publish-subscribe approach to manage changes propagation. DSMW synchronizes concurrent updates of shared semantic pages to ensure their consistency. It offers new collaboration modes to semantic wiki users and supports dataflow-oriented processes.



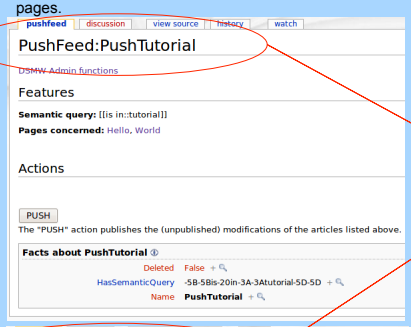
2 Saving pages generates patches

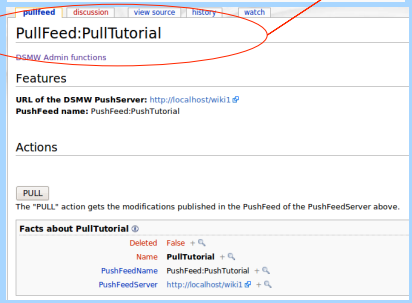
Saving pages generates patches represented as semantic wiki pages.



3 Wikis publish feeds of patches, Wikis subscribe to feeds of patches

Patches can be selected through semantic queries and published in "feeds". These feeds are also semantic wiki pages.

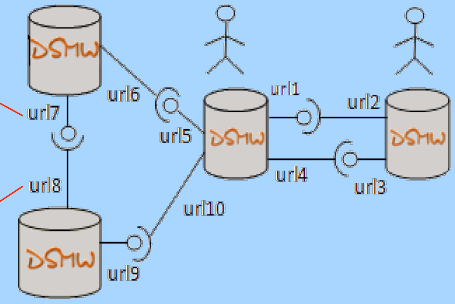




Wikis can subscribe to any number of feeds and start synchronizing. Subscription is also represented as a semantic wiki page.

4 Develop your networks, build editable mashups

If every DSMW nodes receives all operations, then every DSMW nodes have identical copies. DSMW ensure eventual consistency, causality and Intention preservation



6 Conclusions

DSMW allows to synchronize semantic wikis. Semantic wikis can publish feeds of patches and subscribe to feed of patches. DSMW ensures eventual consistency.

Multi-synchronous Collaborative Semantic Wikis . Charbel Rahhal, Hala Skaf-Molli, Pascal Molli and Stéphane Weiss. In Wise'09: International Conference on Web Information Systems, LNCS 5802, 2009.

http://www.dsmw.org

Annotation Component in KiWi

Marek Schmidt and Pavel Smrž

Faculty of Information Technology
Brno University of Technology
Božetěchova 2, 612 66 Brno, Czech Republic
E-mail: {ischmidt,smrz}@fit.vutbr.cz

Abstract. This paper deals with key functionalities of the KiWi annotation component and shows how it enables seamless combination of informal and formal knowledge and transformation of the former to the latter. It demonstrates how the advanced KiWi features, such as nested content items, reasoning and information extraction, can be used together to make rich semantic annotation easy and useful.

1 Introduction

The original wiki systems employ specific wiki languages to edit content. Such languages can easily be extended to allow semantic annotations, which is the approach taken by various semantic wiki systems, such as Semantic MediaWiki [2]. Other approaches to semantic data editing, as, e.g., in OntoWiki [1], provide a rich interface to edit RDF. However, these annotations are not integrated into wiki text content. In KiWi [4], we combine semantic annotations directly with the text content of the wiki pages and provide advanced user interfaces supporting the annotation process with the help of suggestions coming from information extraction.

2 Knowledge Representation in KiWi

KiWi data model is designed to integrate both formal and informal knowledge [4]. A core entity is a *content item* which may contain an XHTML text content or any other kind of multimedia. A title and a list of tags are associated with content items. Each content item corresponds to exactly one *resource*, which enables adding arbitrary RDF statements about the content item. The 1:1 relationship between a content item and a resource reflects a usual practice in semantic wikis. However, having only this linking mechanism is limiting as it is then not possible to represent formal statements about other entities than the current page. Some wikis, such as Semperwiki [3], allow defining an *about* entity, independent of the page, to describe other entities than the current page.

The KiWi nested content items and fragments allow for a more granular and more natural annotation. Fragments enable annotating arbitrary segments of text with arbitrary tags, comments and RDF metadata, which is akin to

annotating a paper with a marker, enhanced with semantics. Nested content items are used for annotating whole sections of text with arbitrary metadata. While no explicit *about* resource as in Semperwiki is supported in KiWi itself, such behaviour can be implemented in KiWi using the native KiWi reasoning support by creating rules. It is thus possible to define an 'about' rule, such that nested item would act as a proxy for a different resource, and any RDF triple assigned to the nested item could automatically be inferred on the referenced resource.

3 User Interface for Information Extraction

The information extraction service in the KiWi system uses natural language processing and machine learning algorithms to provide suggestions for annotations [5]. There are two ways users can interact with the information extraction services in KiWi.

3.1 ASIDE – Annotate Single Document Efficiently

Users can create and edit all kinds of annotations supported by the KiWi system mentioned in the previous section. The information extraction component supports the user by displaying suggestions.

Suggestions can be applied at various stages of the annotation process. Some suggestions can be shown directly in the text, so that the user can select the piece of text just by clicking on the suggestion.

When the user makes a selection of the piece of the text, all the suggestions relevant to that piece of text are displayed. This may include more suggestions than the previous step, as an additional information extraction step is taken at this time which employs apriori information on selecting the particular piece of text.

To support emerging knowledge, it is also crucial to support partially specified annotations (such as a link to an entity of which only type is known, but no entity to link to exists yet), or annotations that conflict with the current ontology (such as an object predicate linking to an entity of a wrong type). The user interface shows the partial annotations in yellow and erroneous annotations in red.

Some suggestions can be ambiguous, such as a suggestion for a link to a user page based on the user names. The annotation can be directly created from these kinds of suggestions, but it will be marked as partially specified, so the user sees that additional action is necessary to make this annotation into a 'green' correct one.

The suggestions can also be displayed in a list sorted by type. The suggestion list includes properties which are defined for the current content item type, but for which no suggestions have been found in the document. A user can thus see if there are some of the required annotations missing. Then, she can annotate just by dragging a selected piece of text and dropping it to a particular type box

to create an annotation of this type. List of types for the current content item is generated from the underlying ontology.

3.2 AMUSE – Annotate Multiple Documents Simultaneously (and Efficiently)

Especially when dealing with a new task, it is often the case that one needs to semantically enrich many documents of the same type, e.g., a bunch of minutes from a series of previous meetings. The use of the ASIDE tool introduced in the previous subsection on each individual wiki page would mean a tedious work. In these situations, it is preferable to focus on a specific type of annotations and process all the documents in one run.

AMUSE is a kind of discovery tool intended to identify all instances of the given type in all the documents available. This tool is also used to configure the information extraction services and to 'tune' it with respect to the particular type of annotation being extracted.

Machine learning algorithms are employed to classify potential instances. AMUSE takes advantage of existing annotations found in the initial training data and retrains the classifiers on the user feedback (accepting or rejecting suggestions).

The behaviour of the tool depends on the type of entity it is used on:

- *Types.* Identify all the pages of the given type, based on document classification. In addition to document features, contextual features derived from the links to pages of the given type are used for classification.
- *Tags.* Same as for types, but additionally also discover all the text fragments that should have this tag.
- *Datatype properties*, such as 'foaf:birthday'. Classify all the fragments of the particular type. A specific extractor can be assigned to each of these kinds of extractions (such as a date extractor for recognizing date information from text, money extractor to recognize amounts of money in a listed currencies, etc.)
- *Object properties*, such as 'foaf:currentProject'. Discovers links to entities and their roles. It works in combination with the type classifier to recognize roles of the potential links.
- *Other entities.* Discover links to this entity from other pages. This may involve disambiguating titles shared by several pages.

After the initialization of this tool for a specific entity (type, tag or property), AMUSE displays a ranked list of suggestions coming from various content items. Users can immediately accept or reject the suggestions, thus annotating the content items and improving the system by providing the training examples at the same time.

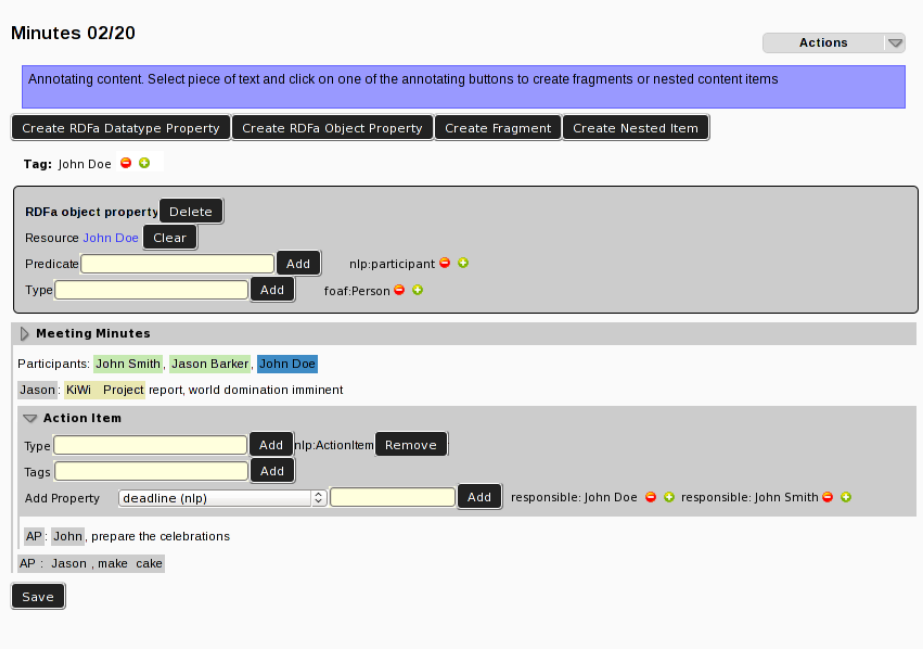


Fig. 1. The annotation tool, annotating a meeting minutes document. Currently editing a link to an entity named 'John Doe'.

4 Use-case Scenario

The scenario discussed in this subsection corresponds to an enterprise setting. A semantic wiki is used to facilitate the knowledge formalisation process in project management tasks. Various kinds of information need to be formally represented in the knowledge base, such as information about projects, customers, people, resources, meetings and tasks. This data can then appear in simple queries ('who attended the meetings where project Foo was discussed'), better task management (tasks can be formally defined directly in the meeting minutes document and automatically appear in the responsible person's 'todo' lists and calendars). This scenario assumes that an ontology describing the entities and their relations already exists in the system.

As demonstrated by Figure 1, meeting minutes are produced in the KiWi system. The annotation tool is opened. The system immediately offers suggestion regarding the type of the document. Selecting the proper type leads to more relevant suggestions. The information extraction component recognizes some of the names of people present and correctly offers the role 'participants'. One of the names could not be identified, because this user was not mentioned yet in the system. It is still recognized that the string corresponds probably to a name of a person, though, so a suggestion to create a new entity of the type 'foaf:Person' is

displayed. Accepting the suggestion creates a new entity in the knowledge base. It will be recognized in all further documents.

Some of the other recognized entities are irrelevant for the current context (such as matching general terms in the ontology), so the user rejects these suggestions. The provided feedback instructs the system not to offer these suggestions in similar contexts in future steps. The user accepts one other suggestion triggered by a label of one of the projects. The meeting is now formally associated with the project.

The user can create another annotation, such as selecting a piece of text around an action item and clicking the Nested Content Item button and selecting the type ActionItem. The ActionItem class specifies several properties, such as deadline and responsible persons. The user can fill the responsible person property just by dragging-and-dropping one of the person annotations created earlier. The task will automatically appear in the task list of the responsible person after the automatic application of the appropriate reasoning rule.

5 Conclusions and Future Directions

The annotation component introduced in this paper has become an integral part of the KiWi system. It enables formal semantic annotation of any kind of existing content. Information extraction supports the annotation by providing context-dependent suggestions which are naturally integrated into the user interface. The discussed use-case scenario shows advantages of the implemented tools in realistic conditions.

Our future work will focus on merging the annotation tool and the KiWi editor and on displaying the suggestions at real time while editing the content. We will also continue to collect real use data to quantify the actual improvement in the annotation process given by the suggestions.

Acknowledgement The research has received funding from the EC's Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 211932.

References

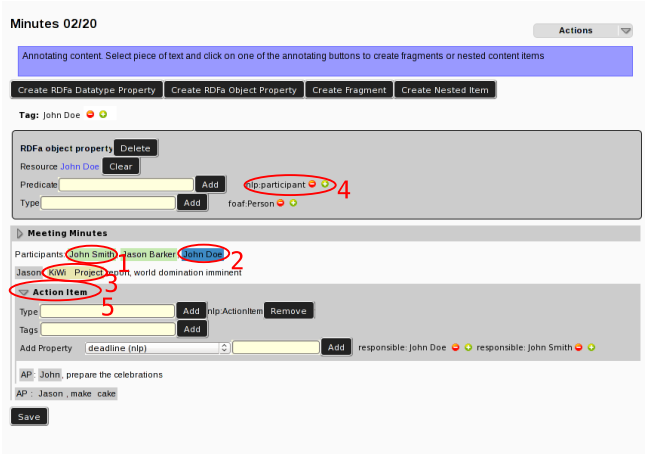
1. Auer, S., Dietzold, S., and Riechert, T. Ontowiki-A tool for social, semantic collaboration. *Lecture notes in computer science 4273* (2006), 736.
2. Krötzsch, M., Vrandecic, D., and Völkel, M. Semantic mediawiki. In *ISWC* (2006), vol. 6, Springer, pp. 935–942.
3. Renaud, E. O., Delbru, R., Möller, K., and Völkel, M. Annotation and navigation in semantic wikis. In *SemWiki* (2006), p. 29.
4. Schaffert, S., Eder, J., Grünwald, S., Kurz, T., Radulescu, M., Sint, R., and Stroka, S. KiWi–a platform for semantic social software. In *Proceedings of the 4th Workshop on Semantic Wikis, European Semantic Web Conference* (2009).
5. Smrz, P., and Schmidt, M. Information Extraction in Semantic Wikis. In *Proceedings of the 4th Workshop on Semantic Wikis, European Semantic Web Conference* (2009).

Annotation Component for a Semantic Wiki

Marek Schmidt, Pavel Smrž Brno University of Technology, Faculty of Information Technology

{ischmidt, smrz}@fit.vutbr.cz

ASIDE



1 Entity Suggestions match titles of existing content items. A user may choose the correct entity from a list of possibilities if the title is ambiguous.

2 Suggestions based on Named Entity Recognition recognize entities (person names and locations) that may not exist in the knowledge base yet.

A new entity with a correct type can be generated automatically.

3 Term suggestions, based on statistical automatic term recognition methods, suggest new entities.

Green annotations have a unique resource and a predicate.

Yellow annotations are incomplete or ambiguous.

Red annotations are inconsistent wrt. the ontology.

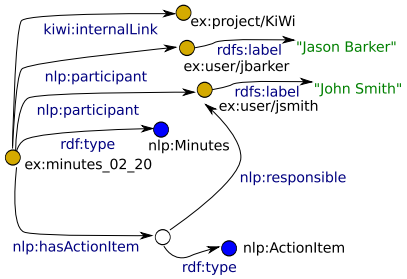
4 Type and predicate suggestions based on the ontology.

5 Nested Content Items have their own URI and so their own datatype and object properties.

Valid and complete RDFa annotation produces RDF data

RDFa fields in the content are updated if the RDF values change.

RDF



HTML page includes all relevant metadata as RDFa

HTML+RDFa

Minutes 02/20(watch)

Created on 26/05/2010 - Last update on 26/05/2010 by John Random
Tags: [Edit]
Rating:

Present: John Smith, Jason Barker, John Doe.

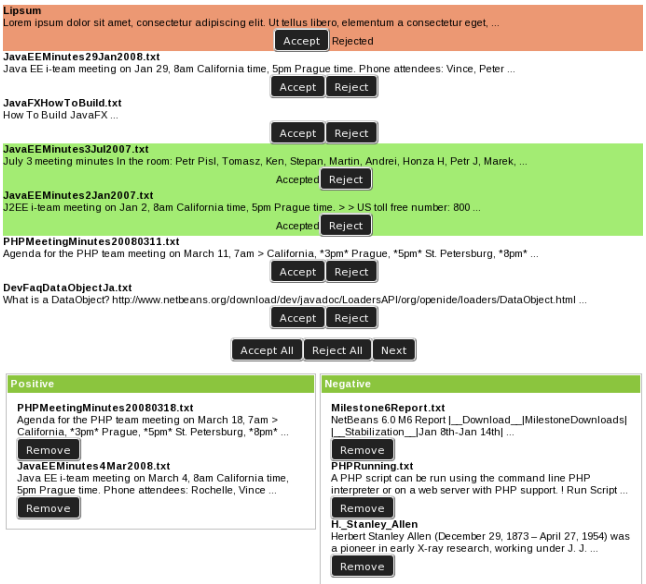
Jason: KiWi Project report, world domination imminent

AP: John Smith, prepare the celebrations.

AP: Jason Barker, make cake.

AMUSE

Suggestions for type "Minutes":



Using machine learning algorithm to annotate by example.

Suggesting Content Item tags and types based on document classification.

Suggesting RDFa link relations based on classification of contexts.

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 211932.

www.kiwi-project.eu

A Perfect Match for Reasoning, Explanation, and Reason Maintenance: OWL 2 RL and Semantic Wikis

Jakub Kotowski and François Bry

Institute for Informatics, University of Munich
http://pms.ifi.lmu.de

Abstract. Reasoning in wikis has focused so far mostly on expressiveness and tractability and neglected related issues of updates and explanation. In this demo, we show reasoning, explanation, and incremental updates in the KiWi wiki and argue that it is a perfect match for OWL 2 RL reasoning. Explanation nicely complements the "work-in-progress" focus of wikis by explaining how which information was derived and thus helps users to easily discover and remove sources of inconsistencies. Incremental updates are necessary to minimize reasoning times in a frequently changing wiki environment.

1 Introduction

One of the main goals of the semantic web [1] is to facilitate processing of information on the web for example by means of reasoning. Semantic wikis [2] are sometimes seen as semantic webs in small; they enhance traditional wikis with semantic annotations in order to make more information directly amenable to machine processing. On the web and even more in wikis, it is natural that inconsistencies arise during work in progress. Users should be supported by a system that not only tolerates inconsistencies but is also able to explain them. The focus of reasoning in KiWi[1] is therefore on a rule-based inconsistency tolerant reasoning that can be explained to users and that also allows for efficient knowledge base updates by the means of reason maintenance. This article describes the state of implementation as of KiWi version 0.8.

Current semantic web applications and frameworks such as Sesame [3], Semantic MediaWiki [4], or IkeWiki [5] implement either specialized RDF/S [2] reasoning or connect a specialized OWL-DL [3] reasoner such as Pellet [6] to provide more expressive reasoning. For example Sesame aims to be a general platform for semantic software based on RDF/S and therefore provides reasoning optimized for RDF/S data. For the Semantic MediaWiki, scalability is one of the top priorities which is why it limits its reasoning to the most efficient forms. In contrast,

[1] http://www.kiwi-project.eu/
[2] http://www.w3.org/TR/rdf-mt/
[3] http://www.w3.org/TR/owl2-profiles/

the Jena [7] framework, also provides custom rules and some support for dealing with inconsistencies in a dataset via so called validation rules. Both Sesame and Jena have only limited support for incremental updates. Jena employs a general purpose RETE-based forward-chaining reasoner which supports incremental additions but no incremental removals[4] [5]. Sesame itself has a limited support for custom rule reasoning and does not offer incremental removals in the general case. There is a reason-maintenance-inspired implementation of incremental updates for Sesame [8] but it is specific to RDF/S reasoning. For Sesame, there is also the OWLIM [6] reasoner which, however, also does not support incremental removals[7]. A contribution of the described implementation is a system capable of incremental processing of fact removals for a general monotonic rule-based reasoner. In addition, it can be easily extended for incremental rule updates.

2 Introduction to KiWi and sKWRL

KiWi is a social semantic platform that features four advanced enabling technologies: reasoning and reason maintenance, querying, information extraction, and personalization and has a wiki as its main application. See [9] for details about the KiWi conceptual model.

sKWRL is a simple KiWi rule language the syntax of which resembles the syntax of the N3 [10] language. It can express a subset[8] of OWL 2 RL – a partial axiomatization of the OWL 2 RDF-based semantics using rules[9] – and has been implemented to provide a starting point for reasoning, explanation, and reason maintenance and also as a step towards the full featured KWRL rule language. sKWRL is a triple pattern based rule language for RDF with two distinctive features: constraint rules and new resource creation in rule heads.

Triple pattern is a generalized RDF triple that can contain a variable in place of subject, property, and object. Bodies of a sKWRL rule consist of a conjunction of triple patterns. Head of a sKWRL rule contains either a conjunction of triple patterns or the "*inconsistency*" keyword. Rules with the "*inconsistency*" keyword in the head are called constraint rules, rules with a conjunction of triple patterns are called construction rules. All variables are implicitly universally quantified and there is no explicit quantification. Variables occurring in a rule head that do not occur in the body of a rule are allowed and construct a new URI reference for each variable binding of the rule body.

One or more sKWRL rules form together a sKWRL program. sKWRL programs can optionally use namespace definitions in a Turtle-like style. An example of a simple sKWRL program is a program deriving the RDF/S subclass and type hierarchy:

[4] http://tech.groups.yahoo.com/group/jena-dev/message/43618
[5] The focus of this paper the focus is on forward-chaining methods.
[6] http://www.ontotext.com/owlim/
[7] http://www.mail-archive.com/owlim-discussion@ontotext.com/msg00496.html
[8] Datatypes and OWL 2 RL rules that use a LIST[] expression are not supported yet.
[9] http://www.w3.org/TR/owl2-profiles/#OWL_2_RL

```
@prefix rdf : <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>

rdf-type: ($1 rdf:type $2), ($2 rdfs:subClassOf $3)
                                        -> ($1 rdf:type $3)
rdf-subclass: ($1 rdfs:subClassOf $2), ($2 rdfs:subClassOf $3)
                                        -> ($1 rdfs:subClassOf $3)
```

Following is an example of a constraint rule:

```
prp-irp: ($p rdf:type owl:IrreflexiveProperty), ($x $p $x)
                                     -> inconsistency
```

The *prp-irp* rule is one of the OWL 2 RL rules with "false" in the head. Currently, rules are loaded from an external file and cannot be modified from within the application.

Internally, the keyword "inconsistency" is replaced by a triple pattern conjunction constructing an annotation for every derived inconsistency. This is needed for explanation purposes for it allows to "track inconsistencies", see [11] for more about tracking. The inconsistency annotation is assigned to the default RDF graph and, in future, optionally to a graph specified by the user. Inconsistencies are displayed as "inconsistency" tags and are highlighted.

3 Implementation

sKWRL is implemented as a component of KiWi which is an enterprise Java application built using Seam[10] and the JBoss application server. The implemented reasoning strategy is semi-naive forward-chaining, also called materialization, which has already been argued to be feasible [8] for applications in the area of semantic web.

sKWRL reasoning is implemented by translating sKWRL rule bodies into JQL (a Java Peristence API version of SQL). The advantage of this approach is the database flexibility provided by JPA, the disadvantage is the inability to use a native database access, which hinders efficiency. The KiWi implementation should therefore be seen as a proof of concept not aiming for high efficiency.

The reasoner also stores derivations of each new derived triple in the form of a *justification* in Doyle's sense [12], i.e. a record of which triples and rules were used in the derivation of a triple. Justifications are then used by reason maintenance and explanation.

4 Reason maintenance

Reason maintenance is a technique originally devised by Jon Doyle [12] for use in problem solvers. A reason maintenance system works closely with a reasoner.

[10] http://seamframework.org/

The reasoner notifies reason maintenance about each derivation it makes and reason maintenance stores derivations in the form of a derivation graph. In the original systems, this graph was used as a kind of computation cache which helped to avoid the need of recomputing in case some base facts changed (i.e. were removed and later added again). Therefore, these systems never removed justifications. In contrast, KiWi uses justifications to determine what facts can possibly be affected by a fact removal thus avoids the inefficient, not incremental approach which is to remove all inferred facts and to do all reasoning anew.

Reason maintenance in KiWi is implemented using the Neo4j [11] NoSQL graph database which natively supports graph structured data.

5 Explanation

Explanation is important for supporting trust of users and it provides a way to determine the root cause of derived inconsistencies. Currently, explanation explains the origin of a derived triple simply by rendering its justification records. There are two renderings available: textual tooltips and an interactive JavaScript explanation tree.

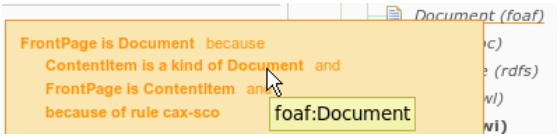


Fig. 1. Part of an explanation tooltip for the triple (localhost:FrontPage rdf:type foaf:Document).

Explanation tooltips present a simple textual explanation of a derived triple, see Fig. 1. The tooltip shows the last step of each possible derivation of the triple "*localhost:FrontPage rdf:type foaf:Document*". The implementation uses a minimal vocabulary to translate common properties into a more readable form.

The explanation tree, see Fig. 2, enables users to explore a graph of all possible derivations and to traverse them until explicit triples are reached. The explanation tree is complemented by a textual explanation, parts of which are highlighted by pointing to a tree node.

Acknowledgements. The research leading to these results is part of the project "KiWi - Knowledge in a Wiki" and has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 211932.

[11] http://neo4j.org/

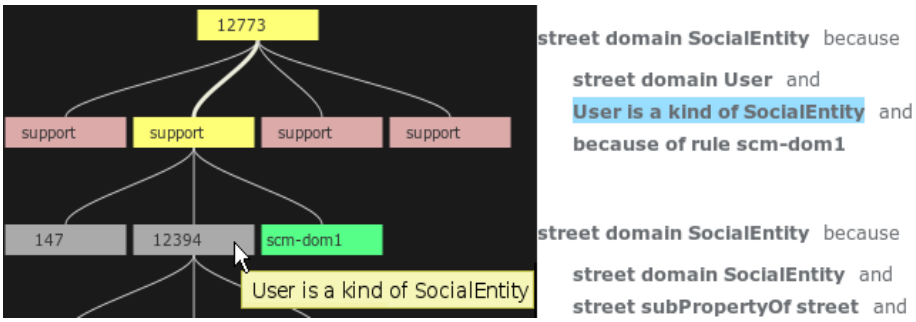


Fig. 2. An interactive explanation tree and a part of textual explanation of the triple *(kiwi:street rdfs:domain swap:SocialEntity)*. Numbers in graph nodes are triple ids of the corresponding KiWi triples. Green nodes represent rules, support nodes represent justifications and the currently selected derivation path is highlighted in yellow. Support nodes can be expanded and collapsed.

References

1. Berners-Lee, T., Hendler, J.: Scientific publishing on the semantic web. Nature **410** (2001) 1023–1024
2. Schaffert, S., Bry, F., Baumeister, J., Kiesel, M.: Semantic wikis. IEEE (2008)
3. Broekstra, J., Kampman, A., Van Harmelen, F.: Sesame: A generic architecture for storing and querying rdf and rdf schema. LNCS (2002)
4. Krötzsch, M., Vrandecic, D., Völkel, M.: Semantic mediawiki. In: ISWC. Volume 6., Springer (2006) 935–942
5. Schaffert, S.: IkeWiki: A semantic wiki for collaborative knowledge management. In: 1st International Workshop on Semantic Technologies in Collaborative Applications (STICA06), Manchester, UK, Citeseer (2006)
6. Sirin, E., Parsia, B., Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A practical owl-dl reasoner. Journal of Web Semantics **5**(2) (2007) 51–53
7. McBride, B.: Jena: A semantic web toolkit. IEEE Internet Computing (2002)
8. Broekstra, J., Kampman, A.: Inferencing and truth maintenance in rdf schema - exploring a naive practical approach. Workshop on Practical and Scalable Semantic Systems (PSSS) (2003)
9. Bry, F., Eckert, M., Kotowski, J., Weiand, K.: What the user interacts with: Reflections on conceptual models for sematic wikis. In: Proceedings of Semantic Wiki Workshop, ESWC, Greece, 2009. (2009)
10. Berners-Lee, T., Connolly, D., Kagal, L., Scharf, Y., Hendler, J.: N3Logic: A logical framework for the World Wide Web. Theory and Practice of Logic Programming **8**(03) (2008) 249–269
11. Bry, F., Kotowski, J.: A social vision of knowledge representation and reasoning. In: Proceedings of SOFSEM 2010: 36th International Conference on Current Trends in Theory and Practice of Computer Science, Špindlerův Mlýn, Czech Republic (23rd–29th January 2010). (2010)
12. Doyle, J.: Truth maintenance systems for problem solving. Technical Report AI-TR-419, Dep. of Electrical Engineering and Computer Science of MIT (1978)

Connecting *Semantic MediaWiki* to different Triple Stores Using RDF2Go

Manfred Schied[1], Anton Köstlbacher[1]. Christian Wolff[2]

University of Regensburg, [1]Information Science / [2]Media Computing
Universitätsstr. 31, 93053 Regensburg, Germany
manfred.schied@stud.uni-r.de; {anton.koestlbacher, christian.wolff}@sprachlit.uni-r.de

Abstract. This article describes a generic triple store connector for the popular *Semantic MediaWiki* software to be used with different triple stores like Jena or Sesame. Using RDF2Go as an abstraction layer it is possible to easily exchange triple stores. This ongoing work is part of the *opendrugwiki* project, a semantic wiki for distributed pharmaceutical research groups.

Keywords: triple store connector, semantic mediawiki, rdf2go

1 Introduction

Semantic MediaWiki (SMW) [1] is one of the most popular and mature semantic wiki engines currently available [2]. It is based on the MediaWiki software [3]. Queries within wiki articles allow reusing available semantic data. For extended query results or to query stored facts from outside the wiki it is necessary to connect SMW to a *triple store*.

We use Semantic MediaWiki for knowledge-based applications in the domain of pharmacy, focusing on psychiatric therapy. In the following, we briefly describe the motivation for a generic triple store connector (ch. 2), give some information on the application context (ch. 3) and explain our implementation approach (ch. 4).

2 An Abstraction Layer for Triple Stores

Triple stores are storage systems tailored for efficient storage of RDF data [4]. In addition, triple stores offer services and programming libraries for inferring new facts or for accessing data using a query language. In a distributed computing system a triple store is a rather independent component with specific features which can be utilized by an associated application programming interface (API). In addition, semantic wiki data can be used in other applications or served as linked data on the web [10].

At the moment, three different triple store products are available for use with SMW, each with a specific connector to SMW. Two of them, *RAP* [5] and *Ontoprise Basic Triplestore* [6] are based on open source software, the third one, *Ontobroker*, is

a commercial product [7]. As SMW (with *Halo Extension*, see [15]) doesn't follow W3C's recommended SPARQL/UL format exactly [8], but uses its own data format for communicating with triple stores, it is necessary to have a connector software between the two systems. To enable users of SMW to select the triple store most suitable for their needs, we have implemented a generic triple store connector using the RDF2Go library [9]. This setup abstracts from the underlying triple store and makes the storage layer easily exchangeable.

3 Application Context

The work described here has been carried out in the context of the recently started *opendrugwiki* project which itself evolved from the drug interaction database *PsiacOnline*[1]. In *PsiacOnline*, drug-interaction information in psychiatric treatment has been collected, uniformly structured, and evaluated by a team of experts in the field [11]. Transforming this approach in the direction of semantic social software appears as a logical next step: On the one hand, we expect a large community of interested experts working in psychiatry to be ready to contribute to this novel method of collecting interaction data. On the other hand, we assume that semantic wikis and the usage of structured knowledge representation standards are adequate for the given information and will allow for the answering of complex information needs.

4 Implementation Details

RDF2Go is a Java library developed at the *Forschungszentrum Informatik* (FZI) in Karlsruhe providing abstract data access methods to RDF triples stored in a triple store ("program now, decide on triple store later"[2]). It uses common adapter classes to access different triple stores. At the moment, RDF2Go delivers adapter classes for *Jena* [12] and *Sesame* [13] and can easily be extended to other triple stores. Communication between SMW and the triple store connector is done via SPARQL and the SPARUL extension [14]. Initial Loading of RDF data from SMW into the triple store is triggered with a SPARUL LOAD command on part of SMW. The connector handles this event by reading the semantic data directly from SMW's database tables due to performance reasons and a missing function for retrieving the wiki's semantic data as a whole via HTTP.

Figure 1 shows the overall architecture of our approach: The semantic media wiki accesses the triple store connector via SPARQL to retrieve query results and via SPARUL to trigger changes made in the wiki to the triple store. The triple store connector – the core component in our architecture – provides an adequate infrastructure for receiving commands and returning resulting triples using web service standards.

[1] *PsiacOnline* is an online service offered by *SpringerMedizin*: http://www.psiac.de
[2] Cf. http://semanticweb.org/wiki/RDF2Go and http://rdf2go.semweb4j.org/

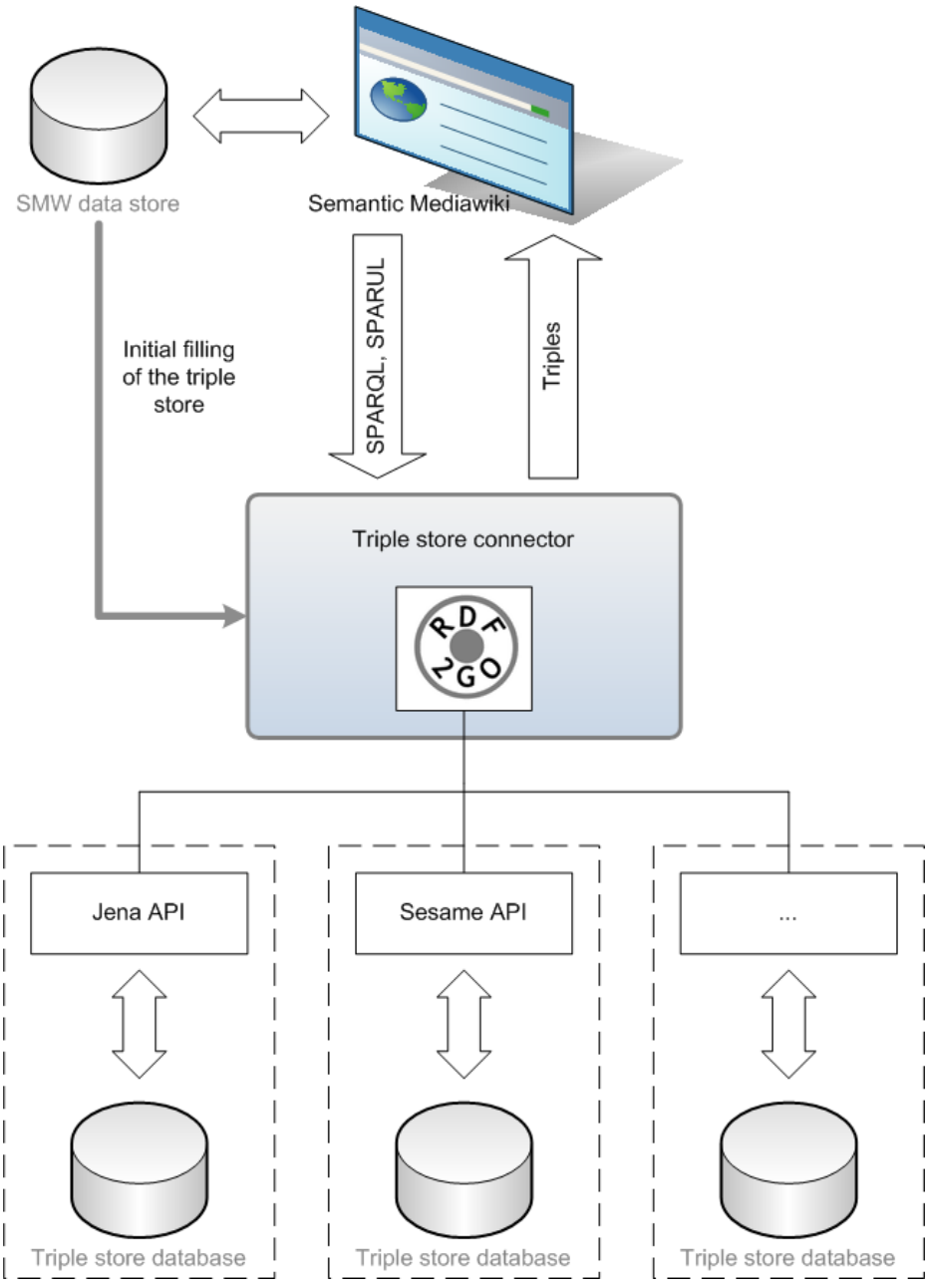


Fig. 1. SMW (with *Halo Extension* [15]) and triple store are connected via the triple store connector. RDF2Go helps to build triple store adapters for all supported triple stores.

5 Demonstration and Conclusion

For the demonstration of our approach, we will present typical usage scenarios taken from our application domain, i.e. drug interaction description and retrieval. Besides showing the feasibility of using an abstract triple store access layer, we also want to demonstrate how semantic wiki technology can facilitate search in complex structured medical data.

By making external semantic storage engines for *Semantic MediaWiki* exchangeable in an easy way, SMW can be conveniently integrated in sophisticated distributed systems. The wiki's semantic data can be re-used with other applications much better since it isn't limited to the triple store engines which have been implemented so far. This enables users of SMW to choose a triple store engine which fulfills their individual needs concerning inference and retrieval of semantic data.

6 References

1. Krötzsch, M., Vrandečić, D., Völkel, M.: Semantic MediaWiki. In: Proceedings of the 5th International Semantic Web Conference (ISWC'06). LNCS, vol. 4273, pp. 935-942. Heidelberg: Springer, (2006)
2. Köstlbacher, A., Maurus, J.: Semantische Wikis für das Wissensmanagement**.** In: DGI e.V./M. Heckner, C. Wolff (Ed.). Information Wissenschaft und Praxis. Mai/Juni 2009, Wiesbaden: Dinges & Frick GmbH, pp. 225-231 (2009)
3. MediaWiki contributors: Mediawiki, The Free Wiki Engine, http://www.Mediawiki.org/w/index.php?title=Mediawiki&oldid=65192 (accessed March 3, 2010)
4. Rusher, J.: Triple Store. Position Paper, Workshop on Semantic Web Storage and Retrieval (2003). http://www.w3.org/2001/sw/Europe/events/20031113-storage/positions/rusher.html
5. Oldakowski, R., Bizer, C., Westphal, D.: RAP: RDF API for PHP. In: Proceedings of the ESWC Workshop on Scripting for the Semantic Web (2005)
6. Ontoprise GmbH (Ed.): Basic Triplestore, http://smwforum.ontoprise.com/smwforum/index.php/Help:Basic_Triplestore (accessed March 3, 2010)
7. Ontoprise GmbH (Ed.): Ontobroker, http://www.ontoprise.de/en/home/products/ontobroker/ (accessed March 3, 2010)
8. Beckett, D., Broekstra, J.: SPARQL Query Results XML Format. W3C Recommendation 15 January 2008. http://www.w3.org/TR/rdf-sparql-XMLres/ (accessed April 21, 2010)
9. Völkel, M.: Writing the Semantic Web with Java. Technical report, DERI Galway (2005)
10. Bizer, C., Cyganiak, R., Heath, T.: How to Publish Linked Data on the Web. http://www4.wiwiss.fu-berlin.de/bizer/pub/LinkedDataTutorial/ (accessed April 22, 2010)
11. Köstlbacher, A., Hiemke C., Haen E., Eckermann, G., Dobmaier, M., Hammwöhner R., PsiacOnline – Fachdatenbank für Arzneimittelwechselwirkungen in der psychiatrischen Pharmakotherapie. In: Osswald, Achim; Stempfhuber, Maximilian; Wolff, Christian (eds.). Open Innovation. Proc. 10 Internationales Symposium für Informationswissenschaft ISI 2007. Constance: UVK, 321-326. (2007)
12. McBride, B.: Jena: A Semantic Web Toolkit. In: IEEE Internet Computing November/December 2002, pp. 55-59. (2002)
13. Broekstra, J., Kampman, A.: Sesame: A generic architecture for storing and querying RDF and RDF Schema. Technical report. Amersfoort: Aidministrator Nederland b. v., (2001).

14.Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF. W3C Recommendation 15 January 2008. http://www.w3.org/TR/rdf-sparql-query/ (accessed March 3, 2010)
15.Friedland, N.S., Allen, P.G., Matthews, G., Witbrock, M., Baxter, D., Curtis, J., Shepard, B., Miraglia, P., Angele, J., Staab, S., Moench, E. Oppermann, H., Wenke, D. Israel, D. Chaudhri, V., Porter, B., Barker, K., Fan, J. Chaw, S.Y., Yeh, P., Tecuci, D.: Project halo: towards a digital Aristotle, AI Magazine, Winter 2004, (2004), online: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.3.4804

Human-machine Collaboration for Enriching Semantic Wikis using Formal Concept Analysis

Alexandre Blansché, Hala Skaf-Molli, Pascal Molli, and Amedeo Napoli

LORIA
Nancy, France
{firstname.lastname}@loria.fr

Abstract. Semantic wikis are new generation of collaborative tools. They allow to embed semantic annotations in the wiki content. These annotations allow to better organize and structure the wiki contents. It is then possible for users to build knowledge understandable by humans and computers. By this way, machines are allowed to produce or update semantic wiki pages as humans can do. In this paper, we propose a new smart agent based on Formal Concept Analysis. This smart agent can compute automatically category trees based on defined semantic properties. In order to reduce human-machine collaboration problems, humans just validate changes proposed by the smart agent. A distributed version of wiki is used to ensure consistency of the content during the validation process.

Keywords. Formal Concept Analysis, Semantic Wiki, Human-Machine Collaboration

1 Introduction

Semantic wikis are new generation of collaborative tools [1,2,3,4]. They allow to embed semantic annotations in the wiki content. These annotations allow to better organize and structure the wiki contents. Semantic wikis allow mass collaboration for creating and emerging ontological resources. They guide the users from informal knowledge contained in documents to more formal structures.

Semantic wikis allow users to build knowledge understandable by humans and computers. By this way, they also allow machines to produce or update semantic wiki pages as humans can do. This opens the opportunity to consider machines as new member of communities to produce and maintain knowledge. Consequently, such "smart agents" can reduce significantly the overhead of communities in the process of continuously knowledge building and correct humans errors.

In [5], authors coupled a case-based reasoner with a semantic wiki. The case-based reasoner can enrich the wikis with new semantic pages and thus can be considered as a smart agent. As pointed out in [5], human-machine collaboration can lead to unstable system if not managed. For example, if humans change the category tree used by the case-based reasoner, the case-based reasoner can produce incorrect results from the point of view of humans users.

In this paper, we propose a new smart agent based on Formal Concept Analysis (FCA) [6]. This smart agent can compute automatically category trees based on defined semantic properties. By this way, the FCA smart agent leverages humans from these tasks. In order to reduce human-machine collaboration problems, humans just validate changes proposed by the FCA smart agent. This is achieved using the DSMW [7] semantic mediawiki extension.

The paper is organized as follows. Section 2 introduces the FCA framework. Section 3 shows how the FCA smart agent is used to enrich the wiki. Section 4 details the validation process. The last section concludes and points future works.

2 Formal Concept Analysis

In this paper, we present a smart agent that enrich a wiki based on a classification method. Actually, any classification methods might be used. We choose Formal Concept Analysis (FCA) because it extracts concepts organized into a lattice, which is interesting for the navigation into the wiki. In this section, we briefly introduce FCA.

Formal Concept Analysis [6] is a classification method allowing to build a concept lattice where concepts are composed of an intent, a maximal set of attributes, and an extent, a maximal set of objects sharing the attributes.

A context K relies on a set of objects G, a set of attributes M and a relation between objects of attributes $I \subseteq G \times M$. Considering an object $g \in G$ and an attribute $m \in M$, $(g, m) \in I$ means that g has the attribute m.

A context can be visualized as a binary table. Table 1 shows a (simple) example of context about animals. There are five attributes that describe animals. Animals may have hair, feather, wings. They might breath in air or water. Objects are animals: bat, bird, cat and fish. In the table, a cross in one cell indicate the animal has the corresponding attribute.

	Has hair	Has feather	Has wings	Breathe in air	Breathe in water
Bat	×		×	×	
Bird		×	×	×	
Cat	×			×	
Fish					×

Table 1. Example of context (animals)

FCA allows to build concepts organized into a lattice. A concept $C_1 = (A_1, B_1)$ is defined by an extent A_1 (a set of objects) and an intent B_2 (a set of attributes that define the concept). If $C_2 = (A_2, B_2)$ is a subconcept of C_1 (denoted by $C_2 \sqsubseteq C_1$), then $A_2 \subseteq A_1$ and $B_1 \subseteq B_2$. The top concept $\top$ contains all the objects and usually its intent is empty (unless an attribute is present in each object). The bottom concept $\bot$ is defined by all attributes but usually contains no objects (unless an object has all attributes).

On figure 1 is shown the concept lattice of the context of table 1. On the graph, every node is a concept. A link between two nodes indicates a subsumption relation (a concept is a subconcept of another concept). The intent of a concept is written on a gray background, the extent on white background.

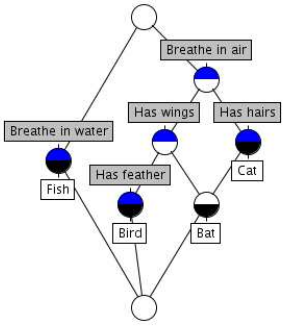


Fig. 1. Galois lattice based on the context from table 1

3 Wiki Enrichment

3.1 Principles

We developed a method that reorganizes the categories of the wiki according to the result of FCA. A new wiki will be created with the same pages and properties, but different categories, based on the lattice of concepts.

The new categories will be created based on the previous ones, and on semantic links between pages. Useful categories human users did not create might be discovered. It is even possible to start a wiki without creating any categories but only semantic links between pages, and then let the smart agent build the categories, based on the semantic links. The new categories facilitate the navigation in the wiki and provide an explicit and complete organization of the pages.

A mapping between original categories and lattice concepts is performed. Each category maps one (and only one) concept: the most general concept containing the category in its intent (the attribute concept). Each concept maps zero, one or several categories. If a concept maps a single category the category will be preserved. If a concept maps two categories or more, it means these categories are identical and should be merged (however this case is very unlikely). If a concept does not map any category, a new category will be created.

Currently, the enrichment is performed by a Java application that access the content of the wiki and create an enriched version of it.

3.2 Case study

The method presented in this paper will be illustrated by a wiki concerning academics. Here we present the initial content of the wiki. We have the following (user-defined) categories:

- `Category:Professor`;
- `Category:Topic`;
- `Category:Course`;
- `Category:Level` which contains two subcategories: `Category:Master 1 Level` and `Category:Master 2 Level`.

We also defined two properties:

- `Property:isTaughtBy`, the domain is a course, the range a professor;
- `Property:isAbout`, the domain is a course, the range a topic.

Finally, we added pages in the wiki:

- `Prof. Smith` and `Prof. Jones` in the `Professor` category;
- `Artificial Intelligence`, `Software Engineering` and `Networks` in the `Topic` category;
- `Knowledge Discovery`, in the `Course` and `Master 1 Level` categories, this page has two semantic links `isAbout:Artificial Intelligence` and `isTaughtBy:Prof. Smith`;
- `Semantic Wiki`, in the `Course` and `Master 2 Level` categories, this page has two semantic links `isAbout:Artificial Intelligence` and `isTaughtBy:Prof. Smith`;
- `Semantic Web`, in the `Course`, `Master 1 Level` and `Master 2 Level` categories, this page has two semantic links `isAbout:Artificial Intelligence` and `isTaughtBy:Prof. Smith`;
- `Design Patterns`, in the `Course` and `Master 1 Level` categories, this page has two semantic links `isAbout:Software Engineering` and `isTaughtBy:Prof. Jones`;
- `Network Administration`, in the `Course` and `Master 1 Level` categories, this page has two semantic links `isAbout:Networks` and `isTaughtBy:Prof. Jones`;
- `IPv6 Protocol`, in the `Course` and `Master 2 Level` categories, this page has two semantic links `isAbout:Networks` and `isTaughtBy:Prof. Jones`;

3.3 Formal concept analysis applied on the wiki

FCA can be applied on the content of the wiki. Objects to be classified by the FCA algorithm are the standard pages of the wiki.

The description of a page is composed of two parts: the categories it belongs to and the semantic properties it has (in our first prototype, we only considered wiki properties of type "Page"). Each of these two parts allow to build a context. We can combine these two context by apposition.

Based on the content of the wiki, as described above, we can create the context shown on table 2. When applied to this context, FCA returns the lattice shown on figure 2.

Table 2. Context based on the wiki

	Professor	Topic	Course	Level	Master 1 Level	Master 2 Level	isTaughtBy:Prof. Smith	isTaughtBy:Prof. Jones	isAbout:Artificial Intelligence	isAbout:Software Engineering	isAbout:Networks
Prof. Smith	×										
Prof. Jones	×										
Artificial Intelligence		×									
Networks		×									
Software Engineering		×									
Knowledge Discovery			×	×	×		×		×		
Semantic Web			×	×		×	×		×		
Semantic Wiki			×	×	×	×	×		×		
Design Patterns			×	×	×			×		×	
IPv6 Protocol			×	×		×		×			×
Network Administration			×	×	×			×			×

In the case study, as one can see on figure 2, four concepts match one category: `Professor`, `Topic`, `Master 1 Level`, and `Master 2 Level`. One concept matches two categories: `Course` and `Level`. All the other concepts do not match any category at all.

How to create the new categories depends on the number of categories matched by each concept. Depending on that number different methods are used. However, no categories are created for the two concepts $\top$ and $\bot$, as $\top$ always contains all pages and $\bot$ does not contain any page.

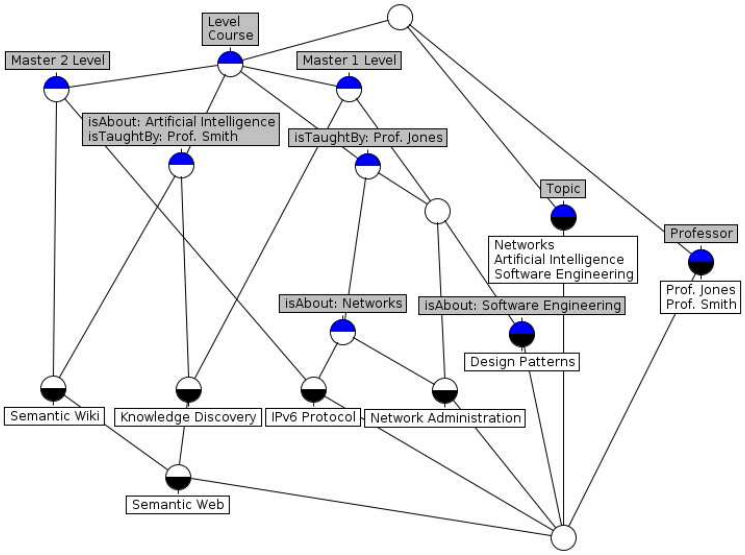


Fig. 2. Galois lattice based on the context from table 2

3.4 Preserving of an original category

If a concept matches one and only one category, this category will simply be preserved in the enriched wiki. This is the case of the category `Topic`, for instance.

Actually, in most cases, all the original categories are preserved.

3.5 Category merging

If a concept matches two categories or more, a new category is created. This new category will merge the content of the original matching categories: text of each pages are concatenated together. A default title is given to the category.

Category merging should be rare. It only happens if two or more categories always appear in the exact same pages. This would happen if several users use different terms for the same concept. Bit by bit, after a number of wiki edition, these different categories will appear in all the same pages and then will be merged by the FCA.

This is the case of the two categories `Course` and `Level`. Having these two categories is due to a naming problem. The enriched wiki has now only one category for this concept.

3.6 New categories

If a concept matches no category, a new one is created, with a default title.

This might happens in two (non-exclusive) cases:

- a page belongs to two categories or more;
- several pages having some identical properties.

A category about courses on software engineering has been created, based on the semantic relation in the page `Design Patterns`. Also, a category about courses available for both Master 1 and Master 2 students has been created, `Semantic Web` is a page of this category.

3.7 Category enrichment

Whatever the creation method of a category, all the new categories are enriched with new text content, based on properties. Sentences like "The pages belonging to this category seems to have relation T with the page P." would be appended in the page. This will help human users to understand the meaning of the category.

For instance, the category of courses about software engineering will contain the sentence "The pages belonging to this category seems to have relation `Property:isAbout` with the page `Software Engineering`.", as a description of the category.

4 Validation

4.1 Validation by human users

After the enrichment, new categories need to be validated by human users. Some merged categories might be spit, some new categories removed. Also, human users should edit all the categories: default titles should be changed into more relevant ones, text should be refined. We will present three examples of validation.

The first one concerns the two categories `Course` and `Level` that have been merged. Having this two categories was a mistake. Human users will acknowledge that and rename the merged category `Course`. They will also rename two of the subcategories `Master 1 Course` and `Master 2 Course` to make them more intelligible.

Another example concerns a new category that has been created based on the semantic relation in the page `Design Patterns` with a default name (`Category:New Category 42`, for instance). As explained in previously, the new category will contain a text describing some properties of the concept. A human user will understand that this category contains courses about software engineering and will rename it consequently. The same thing will be done for the category about courses taught by Prof. Jones.

The last example concerns a subcategory of `Master 1 Course` and `Prof. Jones' Course`. One might consider this category to be irrelevant, or at least not useful. A human user would decide to remove this category from the wiki and update the hierarchal links consequently.

4.2 Distributed wiki organization

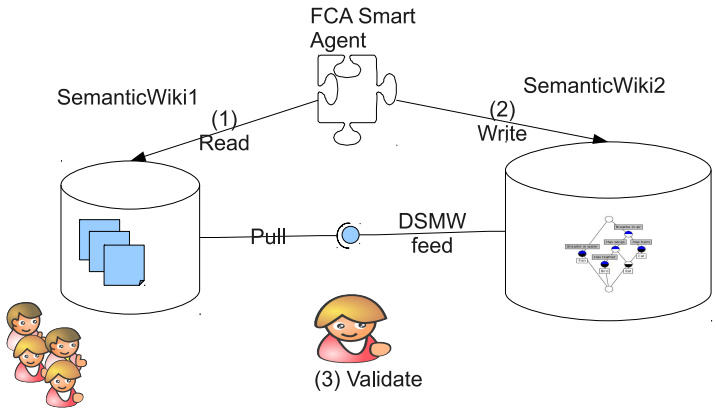


Fig. 3. Man-machine collaboration process

In order to ensure consistency of the data, we used a distributed wiki. Two semantic mediawiki sites are synchronized with the DSMW extension[1] [7] (see figure 3).

- The first one is the "SemanticWiki1" wiki. Humans access this wiki as usual.
- From this "SemanticWiki1", the FCA smart agent creates the lattice in the "SemanticWiki2" site.
- Human users will then check the content of this second wiki site, correct and refine the content.
- Next, they can push the content of "SemanticWiki2" on a push feed.
- Finally, administrator of "SemanticWiki1" can pull validated modifications from "SemanticWiki2" into "SemanticWiki1".

This scenario demonstrates how the DSMW extension can be used to implement processes. In this case, a simple process allows validation of changes produced by the FCA smart agent and avoids the problem of instability of human-machine collaboration.

4.3 Enriched wiki content

After validation, here is the content of the enriched wiki (SemanticWiki1 in figure 3) in the case study:

- `Category:Professor`, contains pages about Prof. Smith and Prof. Jones;

[1] `http://dsmw.org`

- `Category:Topic`, contains pages about Networks, Artificial Intelligence and Software Engineering;
- `Category:Course`;
- `Category:Master 1 Course`, a subcategory of `Category:Course`;
- `Category:Master 2 Course`, a subcategory of `Category:Course`;
- `Category:Artificial Intelligence Course`, a subcategory of `Category:Course`, the page indicates that Prof. Smith is teaching all the courses in this category;
- `Category:Prof. Jones' Course`, a subcategory of `Category:Course`;
- `Category:Master 1 Artificial Intelligence Course`, a subcategory of `Category:Master 1 Course` and `Category:Artificial Intelligence Course`, contains the page about Knowledge Discovery;
- `Category:Master 2 Artificial Intelligence Course`, a subcategory of `Category:Master 2 Course` and `Category:Artificial Intelligence Course`, contains the page about Semantic Wiki;
- `Category:Master 1 and 2 Artificial Intelligence Course`, a subcategory of `Category:Master 1 Artificial Intelligence Course` and `Category:Master 2 Artificial Intelligence Course`, contains the page about Semantic Web;
- `Category:Networks Course`, a subcategory of `Category:Prof. Jones' Course`;
- `Category:Software Engineering Course`, a subcategory of `Category:Prof. Jones' Course` and `Category:Master 1 Course`, contains the page about Design Patterns;
- `Category:Master 1 Networks Course`, a subcategory of `Category:Master 1 Course` and `Category:Networks Course`, contains the page about Network Administration;
- `Category:Master 2 Networks Course`, a subcategory of `Category:Master 2 Course` and `Category:Networks Course`, contains the page about IPv6 Protocol.

5 Conclusion and future work

Semantic wikis allow users to build knowledge understandable by humans and computers. By this way, they also allow machines to produce or update semantic wiki pages as humans can do. This opens the opportunity to consider machines as new member of communities to produce and maintain knowledge. Consequently, such "smart agents" can reduce significantly the overhead of communities in the process of continuously knowledge building and correct humans errors.

In this paper, we proposed a new smart agent based on Formal Concept Analysis. This smart agent allows to reorganize the wiki: new categories are computed and pages are placed into these new categories. This allows a better organization of the content and facilitate the navigation in the wiki.

The refactoring process needs to be validated by human users. Consistency of the wiki is ensured by the use of DSMW: a second wiki site is used to store

the result of the smart agent and is pulled back to the main wiki after human validation.

This paper presented an early work, and more research have to be done in the future. Clearly, if applied on a real wiki, a method such as FCA would produce a large amount of concepts, and it would by impossible for human users to validate any one of them. Some filtering methods should be used to prevent irrelevant categories to be added, based on the number of instances in a category or other criteria.

Using Relational Concept Analysis instead of FCA should provide interesting results. Other clustering methods will also be considered.

In the current version of our method, human users have a feedback from the smart agent, they will take into consideration the new categories that have been created. However, the smart agent does not have a feedback from the human users: if a category has been rejected during the validation process, the smart agent will create it again when the process will be reiterated. To avoid this problem, the smart agent has to be "history-aware" and use the information of the modification by human users during the validation process.

6 Acknowledgments

This research was part of the CyWiki project, funded by the Université Henri Poincaré of Nancy.

References

1. Völkel, M., Krötzsch, M., Vrandecic, D., Haller, H., Studer, R.: Semantic wikipedia. In: WWW '06: Proceedings of the 15th international conference on World Wide Web. (2006) 585–594
2. Schaffert, S.: IkeWiki: A semantic wiki for collaborative knowledge management. In: 1st International Workshop on Semantic Technologies in Collaborative Applications (STICA06), Manchester, UK. (2006)
3. Buffa, M., Ereteo, G., Faron-Zucker, C., Gandon, F., Sander, P.: SweetWiki: A semantic wiki. Journal of Web Semantics, special issue on Web 2.0 and the Semantic Web **6**(1) (2008)
4. Krötzsch, M., Vrandecic, D., Völkel, M., Haller, H., Studer, R.: Semantic wikipedia. Journal of Web Semantic **5**(4) (2007) 251–261
5. Cordier, A., Lieber, J., Molli, P., Nauer, E., Skaf-Molli, H., Toussaint, Y.: Wikitaaable: A semantic wiki as a blackboard for a textual case-based reasoning system. In: 4th Workshop on Semantic Wikis (SemWiki2009), held in the 6th European Semantic Web Conference. (2009) 18–32
6. Ganter, B., Wille, R.: Formal Concept Analysis, Mathematical Foundation. Springer (1999)
7. Rahhal, C., Skaf-Molli, H., Molli, P., Weiss, S.: Multi-synchronous collaborative semantic wikis. In: 10th International Conference on Web Information Systems-Wise 2009. Volume 5802 of Lecture Notes in Computer Science. (2009)

Human-machine Collaboration for Enriching Semantic Wikis using Formal Concept Analysis

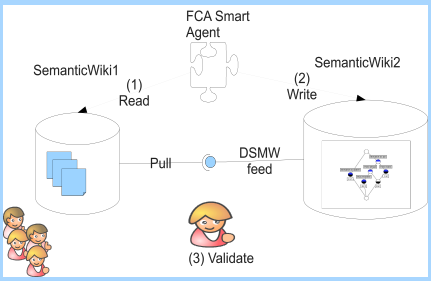
Alexandre Blansché, Hala Skaf-Molli, Pascal Molli and Amedeo Napoli
Université de Lorraine, Nancy, LORIA
INRIA Nancy-Grand Est, France

1 Abstract

Semantic wikis allow users to build knowledge understandable by humans and computers. They also allow machines to produce or update semantic wiki pages as humans can do. This opens the opportunity to consider machines as new member of communities to produce and maintain knowledge. ``Smart agents'' can reduce the overhead of communities in the process of continuously knowledge building and correct humans errors.
A smart agent can compute automatically category trees based on defined semantic properties. A FCA smart agent leverages humans from these tasks. In order to reduce human-machine collaboration problems, humans just validate changes proposed by the FCA smart agent.

2 Human-machine Collaboration

The FCA smart agent reads the semantic wiki pages and proposes a new categorization based on FCA in another semantic wiki. Users modify and when they agree, they synchronize the original wiki with the proposed classification thanks to DSMW extension.



3 FCA Processing

The FCA smart agent builds the table below by requesting the original semantic wiki. Next, it builds the FCA lattice and maps it on Semantic wikis categories.

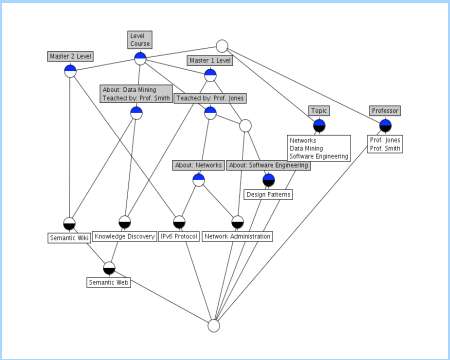
	Professor	Topic	Course	Level	Master 1 Level	Master 2 Level	Teached by:Prof. Smith	Teached by:Prof. Jones	About:Data Mining	About:Software Engineering	About:Networks
Prof. Smith	×										
Prof. Jones	×										
Data Mining		×									
Networks		×									
Software Engineering		×									
Knowledge Discovery			×	×	×		×		×		
Semantic Web			×	×		×	×		×		
Semantic Wiki			×	×	×	×	×		×		
Design Patterns			×	×	×			×		×	
IPv6 Protocol			×	×		×		×			×
Network Administration			×	×	×			×			×

Each category matches one (and only one) concept. Each concept matches zero, one or several categories.

- If a concept matches one and only one category, this category will simply be preserved in the enriched wiki.
- If a concept matches two categories or more, a new category is created.
- If a concept matches no category, a new one is created, with a default title.

4 Category Enrichment

The new categories are enriched with new text content, based on properties. Sentences like ``The pages belonging to this category seems to have relation T with the page P.'' would be appended in the page.



6 Conclusions

Semantic wikis allow users to build knowledge understandable by humans and computers. They also allow machines to produce or update semantic wiki pages as humans can do. This opens the opportunity to consider machines as new member of communities to produce and maintain knowledge. Consequently, such ``smart agents'' can reduce significantly the overhead of communities in the process of continuously knowledge building and correct humans errors.

http://www.dsmw.org

Prototyping a Browser for a Listed Buildings Database* with Semantic MediaWiki

Anca Dumitrache[1] and Christoph Lange[1] and Michael Kohlhase[1], and Nils Aschenbeck[2]

[1] Computer Science, Jacobs University Bremen, Germany
{a.dumitrache,ch.lange,m.kohlhase}@jacobs-university.de
[2] aschenbeck media UG, Bremen, info@aschenbeck.net

1 BauDenkMalNetz and its Intended Applications

Listed buildings, even if they are not top landmarks, are increasingly attracting visitors. People express interest in hidden gems in their neighborhood or along their travel itinerary, and in the history of the building they live in. All required data has been meticulously collected by the offices for historical monuments but is not flexibly *accessible*. In Bremen, the database of buildings (with location, map of the estate, construction history, architect, photos) is searchable and browsable online[3], but that only helps users who know how to use a rigidly structured database search form. Our beginning BauDenkMalNetz effort ("listed building web") aims at a wider purpose: modeling the semantic structure of these data, starting in Bremen but open for other data, and exposing them via a semantic web interface with enhanced querying and presentation capabilities. Requirements beyond interactive browsing comprise auto-generation of customized printed guides (e. g. "Bauhaus villas in my neighborhood"), on-demand presentation on mobile devices (e. g. "medieval churches along my travel itinerary"), and serving linked data for usage by other services.

2 Exploring Possibilities in Semantic MediaWiki

These requirements clearly demand semantic technologies. In this early phase, the *possibilities* of how to represent our knowledge, how to utilize it, and how to represent it to users are not yet entirely clear to us. Therefore, we have started building a *prototype* using Semantic MediaWiki (SMW [4]). Thanks to its stable MediaWiki foundation, its customizability and the wide availability of extensions, SMW is a preferred choice for building prototypes (see, e. g., [2]). In our case, another motivation is that it has already been used for conceptually similar applications. The Archiplanet [1] SMW site contains over 100,000 pages about buildings and architects, however, with a semantically rather weak ontology. We

* We thank the Landesdenkmalamt (state office for historical monuments) Bremen for their data, and Axel Polleres and the SMW community for technological advice.

[3] http://194.95.254.61/denkmalpflege/

are instead planning to follow the existing, stronger database schema of our data for incrementally developing an initial ontology, which is easy in SMW. We have started manually annotating a small, strongly interlinked dataset of listed buildings in one district of Bremen in that way[4] and will explore first possibilities for services on these data, drawing on the abundance of available SMW extensions: *Semantic Forms* is an extension for form/template-based user interfaces, providing forms for adding, editing and querying data, that allows for complex in-document annotations, like embedding forms into other forms by using form templates. This feature will allow for representing fine-granular entities, which nevertheless have some properties of interest, as annotated fragments of larger wiki pages, instead of having very small pages created for them. The custom templates created with *Semantic Forms*, which are adapted to the structure of the data, will also help us in a later step of the project, when we will automatically import a large amount of data entries from the existing relational database by a bot. *Semantic Drilldown* allows for interactively drilling down through different dimensions (= properties) of data. The whole range of values and the number of values is visible from the beginning. This extension enables filtering by semantic properties based on: property values (e. g. the address of a building), page categories (e. g. the district to which a building belongs), date ranges (e. g. the years between which the house was built). *Maps* and *Semantic Maps* are extensions for integrating Bing and Google maps; we will display the locations of buildings on maps. *Semantic Maps* supports compound queries, like which category an item belongs to, that also work with geo-coordinates. *Semantic Graph* is an extension for visually representing results of complex queries as graphs. That can serve to illustrate relations inside the ontology, like "part of" (one building being part of another building) or "time" (by creating a chronological alignment of buildings).

3 Conclusion and Outlook

Using SMW and its extensions, we are creating a functional prototype of the "listed building web", which will be expanded into a full-fledged web portal. We are planning to enhance our initial project-specific ontology by reusing CIDOC CRM, a standard ontology for cultural heritage [5], and GeoNames, a standard ontology for geospatial information, for which a number of web services exist [3].

References

[1] *Archiplanet*. URL: http://www.archiplanet.org.
[2] Jie Bao, Li Ding, Rui Huang, et al. "A Semantic Wiki based Light-Weight Web Application Model". In: *ASWC*. 2009.
[3] *GeoNames*. URL: http://www.geonames.org.
[4] *Semantic MediaWiki*. URL: http://semantic-mediawiki.org.
[5] *CIDOC Conceptual Reference Model*. URL: http://cidoc.ics.forth.gr.

[4] See http://mathweb.org/wiki/BauDenkMalNetz

BauDenkMalNetz

(Prototyping a Browser for a Listed Buildings Network with SMW)

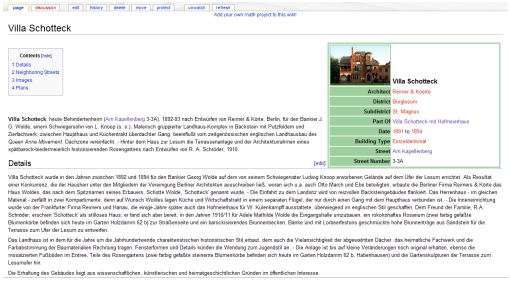
Anca Dumitrache[1], Christoph Lange[1], Michael Kohlhase[1], Nils Aschenbeck[2]

[1] {a.dumitrache, ch.lange, m.kohlhase}@jacobs-university.de, [2] info@aschenbeck.net

What BauDenkMalNetz is about

- In Bremen, the database of listed historical buildings is searchable and browsable online, but its rigid structure prevents from Implementing more complex queries on it.
- Model the semantic structure of these data, and expose them via a semantic web interface with enhanced querying and presentation capabilities.
- Prototype was built using Semantic MediaWiki (SMW).
- The ontology was inspired by the model provided by Archiplanet. Also, we plan to reuse and possibly enhance the CIDOC CRM and Geonames ontologies.
- Further enhacement will provide on-demand presentation on mobile devices (e.g. customized travel itineraries, based on queries like “Bauhaus buildings in my area”).

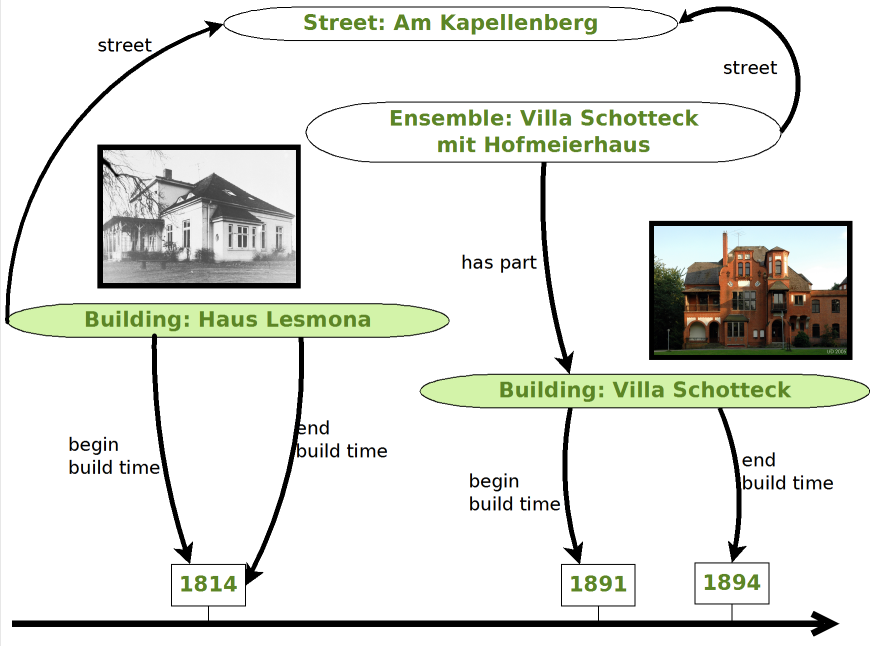
Current web interface of the Listed Buildings in Bremen database



BauDenkMalNetz prototype

SMW extensions

- **Semantic Forms**: represent fine-granular entities that have their own properties of interest as [editable] fragments; also useful for bots importing bulk data from an existing database;
- **Semantic Drilldown**: filter by property values – e.g. address of building, categories – e.g. type of building (house, church etc.), date range;
- **Maps** and **Semantic Maps**: display query results on a map;
- **Semantic Graph**: illustrate relations, e.g. “building – part-of – ensemble” or time (chronological alignment)



RDF graph of possible semantic relations between the data objects

http://mathweb.org/wiki/BauDenkMalNetz