Semantic Wikis for Personal Knowledge Management*

Eyal Oren[1], Max Völkel[2], John G. Breslin[1], and Stefan Decker[1]

[1] DERI Galway, Ireland
firstname.lastname@deri.org
[2] Forschungzentrum Informatik, Karlsruhe, Germany
voelkel@fzi.de

Abstract. Wikis are becoming popular knowledge management tools. Analysing knowledge management requirements, we observe that wikis do not fully support structured search and knowledge reuse. We show how Semantic wikis address the requirements and present a general architecture. We introduce our SemperWiki prototype which offers advanced information access and knowledge reuse.

1 Introduction

Wikis are collaborative hypertext environments, focused on open access, ease-of-use, and modification [8]. Wiki syntax is simple and allows creation of links and textual markup of lists and headings. Wikis commonly support binary data attachments, versioning and change management, change notification, full-text search, and access control.

Wikis are successful tools for collaborative information collection, as observed in the relatively high quality of the Wikipedia encyclopedia [4]. Lately, wikis are becoming popular for personal and organisational knowledge management as well. Knowledge workers use them individually, organisations deploy them internally, and project organisations collaborate through restricted-access wikis[1].

Since managing and enabling knowledge is "key to success in our economy and society" [16, p. 6], we analyse the requirements for knowledge management and how wikis support these requirements. Because knowledge is fundamentally created by individuals [9, p. 59], it is crucial to support these individuals in their personal knowledge management. Considering the knowledge creation spiral [9, p. 62–73], knowledge workers require support in:

1. **authoring:** codifying knowledge into information, enabling sharing
2. **finding and reminding:** finding and reminding of existing knowledge [17]
3. **knowledge reuse:** combining an existing body of knowledge [7]
4. **collaboration:** developing ideas through social interactions

* This material is based upon works supported by the Science Foundation Ireland under Grants No. SFI/02/CE1/I131 and SFI/04/BR/CS0694 and by the European Commission under the Nepomuk project FP6-027705.

[1] Our institutes use wikis for managing projects, clusters, and external collaborations; see http://twiki.org/cgi-bin/view/Main/TWikiStories for more anecdotes.

S. Bressan, J. Küng, and R. Wagner (Eds.): DEXA 2006, LNCS 4080, pp. 509–518, 2006.

1.1 Personal Knowledge Management Tools

Current tools for personal knowledge management have limitations: analog approaches are not automated and cannot be searched, traditional digital approaches are restrictive and do not support ad hoc structures.

Traditional tools such as todo lists or paper piles are very common [6] and are suitable for authoring, but they do not support finding, reminding, knowledge reuse, or collaboration. *Hierarchical filing* (of emails and files) allows browsing and (full-text) searching, but does not support authoring, knowledge reuse, reminding, and collaboration. *Personal information management* tools (e.g. MS Outlook) manage email, calendar, and tasks and support finding and reminding, but they do not support authoring, knowledge reuse, and collaboration.

1.2 Wikis for Knowledge Management

Wikis support authoring and collaboration to a high extent and are popular due to their simplicity and easy collaborative access [2]. On the other hand, wikis do not enable knowledge reuse and have only limited support for finding and reminding.

These limitations result from a lack of structure in the wiki content: almost all information is written in natural language, and has little machine-understandable semantics. For example, a page about the author John Grisham could contain a link to the page about his novel "The Pelican Brief". The English text would say that John Grisham wrote the Pelican Brief, but that information is not machine-understandable, and can therefore not be used for querying, navigating, translating, or aggregating any information.

More specifically, wikis do not allow structured access to data and do not facilitate consistent knowledge reuse:

Structured Access. A wiki does not offer structured access for browsing or searching information. One cannot currently **query** wiki systems, because the information is unstructured. For example, users looking for *"How old is John Grisham?"*, *"Who wrote the Pelican Brief?"*, or *"Which European authors have won the Nobel price for literature?"* cannot ask these questions directly. Instead, they have to navigate to the page that contains this information and read it themselves. For more complicated queries that require some background knowledge users need to manually combine the knowledge from several sources.

Another example of structured access to information can be found in page **navigation**: wikis allow users to easily make links from one page to other pages, and these links can then be used to navigate to related pages. But these explicit links are actually the only means of navigation[2]. If no explicit connection is made between two related pages, e.g. between two authors that have the same publishing company, then no navigation will be possible between those pages.

Knowledge Reuse. Reusing information through **reference and aggregation** is common in the real world. Consider for example that books are generally

[2] Except for back-references that appear on a page and show pages that reference it.

written by an author and published by the author's publisher. The books authored by John Grisham (on his page) should therefore also automatically appear as books published by Random House (on their page). But creating such a view is currently not possible in a wiki, and instead the information has to be copied and maintained manually.

In current wikis it is either assumed that people will speak a common language (usually English) or that **translations** to other languages will be provided. But manually translating pages is a maintenance burden, since the wiki system does not recognise the structured information inside the page text. For example, a page about John Grisham contains structured information such as his birth date, the books he authored, and his publisher. Updates to this information have to be migrated manually to the translated versions of this page.

2 Semantic Wikis

A Semantic wiki allows users to make formal descriptions of resources by annotating the pages that represent those resources. Where a regular wiki enables users to describe resources in natural language, a Semantic wiki enables users to additionally describe resources in a formal language. The authoring effort is relatively low: the semantic annotations are very similar to the layout or structural directives that are already in widespread use in ordinary wikis.

Using the formal annotations of resources, Semantic wikis offer additional features over regular wikis. Users can query the annotations directly ("show me all authors") or create views from such queries. Also users can navigate the wiki using the annotated relations ("go to other books by John Grisham"), and users can introduce background knowledge to the system ("all poets are authors; show me all authors").

In designing a Semantic wiki system several architectural decisions need to be taken. In this section, we explain the basic architecture and outline the design choices and their consequences.

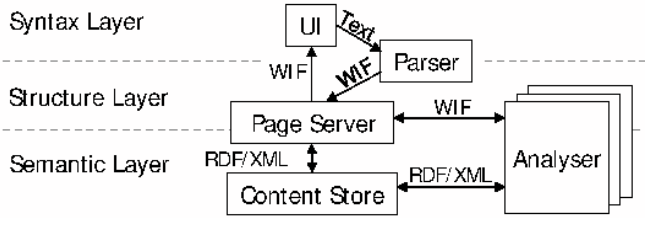


Fig. 1. Architecture of a Semantic wiki

2.1 Architecture Overview

A Semantic wiki consists (at least) of the following components: a user interface, a parser, a page server, a data analyser, and a data store, as shown in Fig. 1.

First we introduce each component, then we discuss the information access, the annotation language, and the ontological representation of the wiki.

overview: The *page server* encapsulates the business logic of the wiki and exposes its data in the neutral wiki interchange format WIF [15]. The *user interface* lets the user browse and query the wiki pages. When a page is edited, the WIF is converted to wiki syntax and the changed wiki syntax is *parsed* back to WIF. The *content store* stores all data as RDF, allowing querying with standard RDF query languages. The *analyser* interacts with the page server and the content store and augments pages and RDF with inferred relations; different types of analysers fit into the architecture, e. g. based on formal reasoning or on statistics.

user interface: responsible for all user interaction. If the wiki is web-based (the classical model), then the user interface is a web server. A desktop application can also act as the user interface component. In this case, collaboration is achieved by using a shared content store.

The user interface allows users to type text and annotations in a freely intermixed fashion. The user interface shows terms from shared *ontologies*, enabling users to browse for an appropriate term[3].

page server: includes standard wiki functionality such as version management, binary attachments, and access control.

parser: converts the text written by the user into objects: it parses the text for semantic annotations, layout directives, and links. This is transmitted in the wiki interchange format WIF.

content store: is responsible for storing and retrieving the semantic annotations, and for exchanging data with other information systems (such as other semantic wikis). An an off-the shelve RDF triple store can be used.

data analyser: is responsible for computing a set of related resources from a given page. In a regular wiki, this means finding all back-references, i.e. pages that link to the current one. In a Semantic wiki the relations between resources are much richer: the data analyser can use the annotations about the current and other pages to search for relevant relations in the content store (such as "other books by current author" or "other people with these parents").

2.2 Annotation Language

For the user of a Semantic wiki, the most visible change compared to conventional wikis is the modified annotation language. For Semantic wikis the annotation language is not only responsible for change in text style and for creating links, but also for the semantic annotation of wiki pages and for writing embedded queries in a page.

[3] Descriptions can be shared and understood if written in a common terminology, and browsing ontologies helps finding an appropriate common term.

Annotation Primitives. As in conventional wikis, internal links are written in CamelCase or by enclosing them in brackets; external links are written as full absolute URIs, or are abbreviated using namespace abbreviations.

Table 1. Annotation syntax

syntax	meaning
`rdf:type foaf:Person`	page has rdf:type foaf:Person
`dc:topic [http://google.com]`	page has dc:topic http://google.com
`dc:topic TodoItem`	page has dc:topic http://wikinamespace/TodoItem
`dc:topic ''todo''`	page has dc:topic "todo"
`?s dc:topic ?o`	embedded query for all pages and their topics
`?s dc:topic TodoItem`	embedded query for all todo items

The additional syntax for semantic annotations is shown in table 1: annotations are written on a separate line, and consist of a predicate followed by an object. Predicates can only be resources (identifiable things), objects can be either resources or literals. An example page is displayed in figure 2. It describes John Grisham, an author published by Random House.

JohnGrisham

```
John Grisham is an author and retired lawyer.

rdf:type foaf:Person
dc:publisher RandomHouse
```

Fig. 2. Example page

Subject of Annotations. Wiki pages often refer to real-world resources. Annotations can refer to a wiki page but also to the resource described on that page. For example, the triple "W3C created-on 2006-01-01" can refer to the creation date of the organisation or to the creation date of the wiki page about that organisation.

The question "what do URIs exactly identify" (of which the annotation subject is a subclass) is an intricate open issue on the Semantic Web[4]: a URI can for example identify an object, a concept, or a web-document.

Our approach is to explicitly distinguish the "document" and the "real-world concept" that it describes. Since we expect more annotations of the real-world concepts than annotations of the page itself, we attribute annotations by default to the real-world concept, and allow annotations about the page (such as its creation date, version, or author) to be made by prepending annotations with an exclamation mark.

For example, figure 3a shows a page that describes the World Wide Web consortium. The page explains the W3C and the annotations state that the

[4] See `http://www.w3.org/DesignIssues/HTTP-URI.html`.

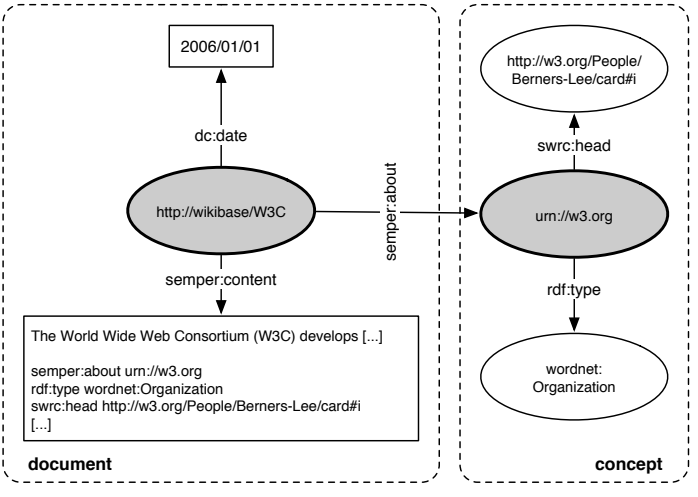
W3C

```
The World Wide Web Consortium (W3C) develops interoperable
technologies (specifications, guidelines, software, and
tools) to lead the Web to its full potential

semper:about urn://w3.org
rdf:type wordnet:Organization
swrc:head http://www.w3.org/People/Berners-Lee/card#i

Now we have an annotation about the page itself:
!dc:date "2006/01/01"
```

(a) example page



(b) RDF representation

Fig. 3. RDF representation of an example page

organisation is directed by Tim Berners-Lee. The last annotation, prepended with an exclamation mark, refers to the page (document) instead of to the W3C organisation: it states that the page was created on 2006-01-01. We use the "semper:about" predicate to relate the page to the concept that it describes.

Embedded Queries. Users can embed queries on any wiki page. These embedded queries are executed when a page is visited, and their results are included in the displayed page[5]. They could for example show aggregations (such as all the books written by John Grisham); embedding queries in page allows knowledge reuse by persistently combining pieces from different sources.

[5] Views resulting from embedded queries could be read-only or editable. Editable views cause some maintenance issues (should the change be recorded in the version history of the result page or of the page affected by the edit) similar to the view-update problem in databases.

As shown earlier in Table 1, embedded queries are written using triple patterns, sequences of subject, predicate, object, that can contain variables (names that start with a question mark). A triple pattern is interpreted as a query: triples matching the pattern are returned. Patterns can be combined to form joins. Fig. 4 shows the earlier example page about John Grisham, including an embedded query at the bottom of the page. The query returns all books written by JohnGrisham; it creates a view on the data that is displayed below the page text.

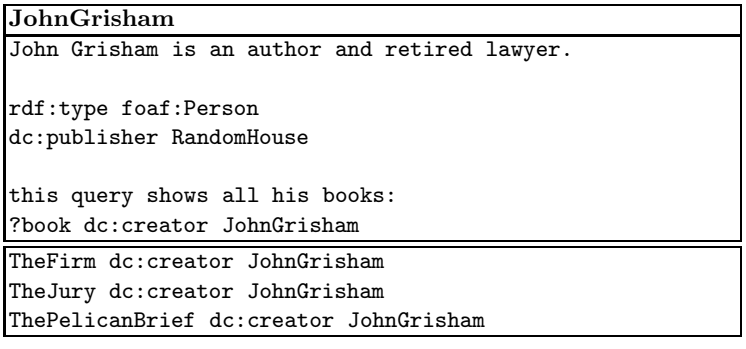


Fig. 4. Page showing embedded query

2.3 Information Access

Information can be accessed by structured navigation and querying facilities.

Navigation. Navigation in ordinary wikis is limited to explicit links entered by users; it is not possible to navigate the information based on structural relations. A Semantic wiki provides the metadata necessary to navigate the information in a structured way. For example, knowing that John Grisham is an author, we can show all other authors in the system, and offer navigation to them.

Our approach for structural navigation is based on faceted meta-data browsing [18]. In faceted browsing, the information space is partitioned using orthogonal conceptual dimensions (facets) of the data, which can be used to constrain the relevant elements in the information space. For example, a collection of art works can consists of facets such as type of work, time periods, artist names, geographical locations, etc.

Common faceted browsing approaches construct the facets manually for a specific data collection. But since in a Semantic wiki users are free to add arbitrary metadata, manual facet generation does not suffice; instead, we have developed a technique to automatically generate facets for arbitrary data [11].

Querying. We distinguish three kinds of querying functionality: *keyword search*, *queries*, and *views*:

1. A *keyword-based full-text* search is useful for simple information retrieval, and supported by all conventional wiki systems.

2. *Structured queries* use the annotations to allow more advanced information retrieval. The user can query the wiki for pages (or resources) that satisfy certain properties. To retrieve for example all authors one can query for "?x type author". Triple patterns can be combined to form database-like joins: "?x type author and ?x has-publisher ?y" retrieves all authors and their publishing companies.
3. As discussed earlier, users can create persistent searches by *embedding queries* in pages. A query included on a page is executed each time the page is visited and continuously shows up-to-date query results.

3 Implementation

SemperWiki[6] is our prototype implementation of a Semantic wiki. We give only a brief overview of the implementation, see [10] for details. Figure 5 shows a screenshot from the desktop version, displaying a page about Armin Haller. The page freely intermixes natural language and simple semantic annotations stating that he is a male person. On the right hand side related items are shown based on the semantic annotations. Users are offered more intelligent navigation based on the metadata, in addition to the explicit links between pages. On the bottom of the page we see an embedded query, that shows a continuously up-to-date view of all pages created by Eyal Oren.

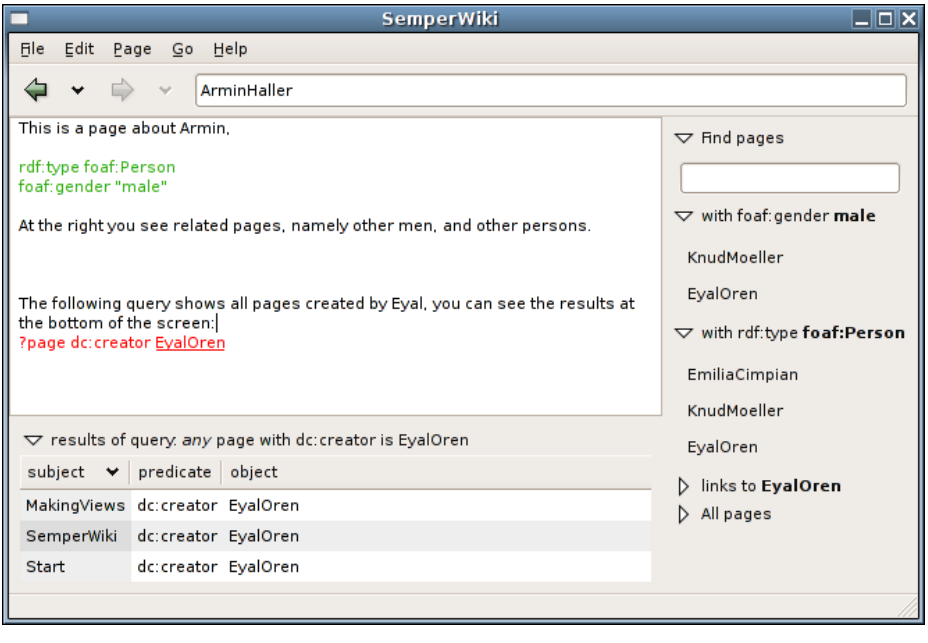


Fig. 5. Navigating and Information reuse

[6] `http://semperwiki.org/`

SemperWiki addresses the noted limitations of ordinary wikis. Concerning **structured access**, users can find related information through associative browsing: the wiki analyses the semantic relations in the data and provides navigational links to related information. Users can search for information using *structured queries*, in addition to simple full-text search.

Concerning **information reuse**, the semantic annotations allow better translation and maintenance; the annotations are language independent[7] and can be understood and reused without barriers. Users can also write *embedded queries*, creating saved searches (database views). These views can be revisited and reused, and provide a consistent picture of structured information. Furthermore all information is represented in RDF using standard Semantic Web terminologies which allows information exchange.

4 Related Work

Several efforts consider using Wikis as collaborative ontology editors, such as OntoWiki [5] or DynamOnt [3]. These efforts focus on ontology engineering rather than improving Wiki systems; they for instance do not follow the free-text editing model of Wikis.

Souzis [12] describes an architecture for Semantic wikis but focuses on annotating and representing page structure while we are concerned with page content, and discusses specific implementation decisions rather than generic architecture choices. Platypus [13] is a wiki with semantic annotations, but adding and using annotations requires significantly more effort than normal text. Both WikSAR [1] and Semantic Wikipedia [14] offer easy-to-use annotations, but neither allow reuse of existing Semantic Web terminologies, and both only allow simple annotations of the current page (thereby excluding blank nodes). Furthermore, none of the above consider the representation distinction between documents and pages.

5 Conclusion

Wikis are successful for information collection, but do not fully satisfy the requirements of personal knowledge management. We have shown how Semantic wikis augment ordinary wikis: using metadata annotations they offer improved information access (through structured navigation such as faceted browsing and structured queries) and improved knowledge reuse (through embedded queries and information exchange). We have implemented our architecture in a first prototype and plan to validate its usability in a future user study.

References

1. D. Aumueller and S. Aurer. Towards a semantic wiki experience - desktop integration and interactivity in WikSAR. In *Semantic Desktop (ISWC)*. 2005.
2. A. L. Burrow. Negotiating access within wiki: a system to construct and maintain a taxonomy of access rules. In *HyperText '04*, pp. 77–86. 2004.

[7] If ontologies contain translations of concept and property labels.

3. E. Gahleitner, W. Behrendt, J. Palkoska, and E. Weippl. On cooperatively creating dynamic ontologies. In *HyperText*, pp. 208–210. 2005.
4. J. Giles. Internet encyclopaedias go head to head. *Nature*, 438:900–901, 2005.
5. M. Hepp, D. Bachlechner, and K. Siorpaes. Ontowiki: Community-driven ontology engineering and ontology usage based on wiki. In *WikiSym*. 2005.
6. S. R. Jones and P. J. Thomas. Empirical assessment of individuals' 'personal information management systems'. *Beh. & Inf. Techn.*, 16(3):158–160, 1997.
7. A. Kidd. The marks are on the knowledge worker. In *CHI*, pp. 186–191. 1994.
8. B. Leuf and W. Cunningham. *The Wiki Way: Collaboration and Sharing on the Internet*. Addison-Wesley, 2001.
9. I. Nonaka and H. Takeuchi. *The Knowledge-Creating Company*. Oxford University Press, New York, 1995.
10. E. Oren. SemperWiki: a semantic personal Wiki. In *Semantic Desktop (ISWC)*. Nov. 2005.
11. E. Oren, *et al.* Annotation and navigation in semantic wikis. In *SemWiki (ESWC)*. Jun. 2006.
12. A. Souzis. Building a semantic wiki. *IEEE Intelligent Systems*, pp. 87–91, Sep. 2005.
13. R. Tazzoli, P. Castagna, and S. E. Campanini. Towards a semantic wiki wiki web. In *ISWC*. 2004.
14. M. Völkel, M. Krötzsch, D. Vrandecic, and H. Haller. Semantic wikipedia. In *WWW*. 2006.
15. M. Völkel and E. Oren. Towards a Wiki Interchange Format (WIF) – opening semantic wiki content and metadata. In *SemWiki (ESWC)*. Jun. 2006.
16. E. Wenger, R. McDermott, and W. M. Snyder. *Cultivating Communities of Practice*. Harvard Business School Press, 2002.
17. S. Whittaker and C. Sidner. Email overload: exploring personal information management of email. In *CHI*, pp. 276–283. 1996.
18. K.-P. Yee, K. Swearingen, K. Li, and M. Hearst. Faceted metadata for image search and browsing. In *CHI*, pp. 401–408. 2003.