Towards Semantic Desktop Wikis

Malte Kiesel, Leo Sauermann
DFKI GmbH - Knowledge Management Dept.
Erwin-Schrödinger-Straße, Bldg. 57
D-67663 Kaiserslautern, Germany
{malte.kiesel, leo.sauermann}@dfki.de

ABSTRACT

To manage information on a personal computer, tools are needed that allow easy entering of new knowledge and that can relate ideas to existing information. *Wikis* allow entering of information in a quick and easy way. They can be employed for both collaborative and personal information management. *Semantic Web* standards such as RDF(S) and OWL provide means to represent formalized knowledge. Using these standards to represent relations between individual desktop data sources, an integrated view of the user's information can be realized, known as the *Semantic Desktop*.

In this paper, we propose combining information represented using Semantic Web standards with the simple information management known from wikis to realize the integrated view on information—the Semantic Desktop. Wiki pages are used to author semantic content that acts as *semantic glue* relating information present in desktop applications and desktop data sources such as text documents. The result is a *Semantic Desktop Wiki*, which can form a melting pot for ideas and personal information management.

Keywords

Semantic Web, Semantic Desktop, Wiki, Semantic Wiki, Personal Information Management

1. INTRODUCTION

The Semantic Desktop [10] is about creating a network of associations between your sources of information—for example, text documents, web bookmarks, and calendar entries.

However, relating these resources semantically requires "semantic glue"—when connecting resources, you need a way to express *why* the resources are connected. Ideally, this information would be located *within* the resources. Unfortunately, often one cannot or does not want to tamper with the resources itselves.

In real life, this problem is not new. If you want to organize many files in a filesystem, you often resort *not* to use descriptive file and directory names exclusively but to create new files – *README* files come to mind.

In this example, file and directory names correspond to properties, files correspond to resources, and the READMEs correspond to resources that do not fit into the standard heavyweight resource schema but denote a lightweight information resource.

So, is there a kind of widely accepted standard for quickly writing down information that fits into the modern web–based information world? We argue *wikis* fit this decription. Nowadays, wikis are used for a wide range of applications, from the well–known Wikipedia [12], to corporate intranet applications, and personal wikis that are the equivalent of a personal notepad.

Therefore, it is natural to not only enable standard application data to be linked semantically, but to use wikis for supplying the *semantic glue* that is necessary for this network to function fully—providing powerful but at the same time simple ways of relating your data.

We will take a look at existing wiki implementations and the upcoming semantic wikis. Then we will show how data of the Semantic Desktop can be integrated to wiki pages, giving the opportunity to combine wiki text with structured information. Not only information from outside the wiki can be included— information authored inside the wiki can be used to augment information present in external resources. Finally, we will present our conclusion and further work.

2. WIKIS AND THEIR PROBLEMS

Current applications of wikis range from open encyclopedias and collaborative information spaces (most notably in open source software projects[1] but one can find wikis in almost every project that needs documentation to be created collaboratively) to personal notepads[2]. For some applications even specialized wiki implementations exist. For example, MediaWiki[3] is crafted for Wikipedia [12], focusing on the requirements of the encyclopedia use case.

Simple to the limits

One problem with most existing wiki implementations is that they take the *keep it simple* approach too far—while it is a good idea to make editing information as simple as possible, relying totally on basic wiki functionality for simple state-of-the-art structuring techniques is of doubtful benefit. For example, it may be an interesting thing to base a category system on backlinks[4] technically, but applying a generic technique (such as backlinks) to implement a specialized application (a category system) quickly becomes uncomfortable—nevertheless, this is a common practice in wiki implementations.

[1] In software project management software, wikis are already a major feature—see Trac (http://www.edgewall.com/trac/) as an example.

[2] WikidPad: http://www.jhorman.org/wikidPad/

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

[4] *Backlinks* are the set of pages pointing to the currently viewed page, which is in this case a page representing a page category—for example, if we create a page named *CategoryCompany* and put a link to it on every page that has a company as subject, we find all company pages by looking at the backlings of the *CategoryCompany* page

Copyright is held by the author/owner(s).
To appear in: Novática/UPGRADE special issue on "Semantic Web"

We argue that, while giving the user freedom when entering his information and thus, providing a low entrance barrier, a wiki should also provide more elaborate means to express information. This should be available as an *optional* feature in order to keep the entry barrier low.

One of these "more elaborate means of expressing information" is to introduce *semantics*.

Readable for whom?

To provide a qualitative idea of the term of *semantics* in the current context, one can say that it is about the differences of what words and statements mean to one person, to another person, and to computer programs. Only humans are able to read and understand the texts contained in a wiki—for machines, without sophisticated processing the wiki is a large number of text pages which link to each other, or, more technically spoken, a set of strings (page content) indexed by strings (page names), interlinked with each other.

This is a sad fact as this way, much information contained in wikis is either de facto unretrievable or needs vast efforts to get exploited. Take Wikipedia as an example: almost the complete common knowledge is in there. However, one cannot find the most important philosophers of a certain epoch, since the data about the individual philosopher's lifetime is present, but only in a human–, not machine–readable way. It has to be noted that there several ways of solving this problem, most notable the standard "wiki way", which is to simply set up a page for every epoch, describing its most important people. However, it is clear that this does neither scale nor satisfy more complex queries.

Metadata for wikis

This problem can be solved by supporting adding *metadata* to the wiki's contents, leading to a *semantic wiki*. This can come in various ways: A simple kind of metadata describing a wiki's page structure is, for example, using a concept hierarchy, and extending the wiki's contents to use that hierarchy. Put more formal, this means linking each wiki page to a concept using an *has-subject* property.

Obviously, categories are very coarse–grained metadata. However, one can think of metadata that is arbitrarily complex, ultimatively leading to a formalized representation of the complete wiki's contents. In practice, one will use something in the middle of the extremes. For example, the Wikipedia community tries to extend the wiki implementation used by Wikipedia with typed links [5], meaning that additionally to every link you write in the wiki's page text you can specify the link's type. E.g., when referencing *Germany* on the wiki page describing *Berlin*, the author is able to tag the page's link to Germany with the fact that the link uses the type *is-capital-of*.

Since the Semantic Desktop is also about managing your personal information sources such as files, a Semantic Desktop wiki should provide means to incorporate references to these information sources into the personal knowledge network.

Linking to external resources

A problem when trying to integrate the wiki idea into the Semantic Desktop scenario is that standard, web–based wikis only poorly support linking to desktop resources. The URL/hyperlink idea the WWW is based on simply does not support linking to local resources. While it is possible to use workarounds (such as `file://` URLs for linking to local files, or e-mail message ids for linking to e-mails), this is cumbersome and impractical on a larger scale (both workarounds have their problems: `file://` URLs only work on the host they are intended for but do not bear a link to that host—e-mail message ids identify a mail but do not supply a clue on how to retrieve it). The Semantic Desktop framework [10] provides means to identify and link resources by associating every resource present on the desktop with a URI, so a semantic wiki can make use of this and integrate with desktop resources.

3. SEMANTIC DESKTOP WIKIS

Taking the capabilities of semantic wikis, we can create a more productive work environment based on the simplicity of wikis, the semantical power of RDF and the vast data sources available on the Semantic Desktop. In this section we will show how these three approaches can be merged.

The *Semantic Desktop* is an approach to bring Semantic Web technologies to desktop computers. An overview is given in [9]. The social aspects and a possible roadmap for future developments can be found in Stefan Decker's work [3].

First, wikis are suited for *Personal Information Management* (PIM). For example, in the scenario of customer-relationship-management, the salesperson *Peter* of company **AcmeWear** may use a personal wiki to note information about his customers. It would be possible to generate wiki pages for products and for customers and for business meetings, where Peter meets customer *Company Freistein*. The product of choice would be *Security Glove MKL* that is needed by the customer to handle chemical probes.

Peter could now write a wiki note with the following text (bold words are links to other pages).

> Title: Customer Freistein. Text: **Kent Brockman** working at **Freistein** noted that he is interested in the **SecurityGloveMKL** for use in their chemical lab facility. Workers there have to handle glycol and AG342 and are unhappy with the existing gloves by our competitor **WorseWear**. I already had a good phone call and made an offer, see **OfferGU1234**.

In such a note, Peter is able to catch the current status of his relation with the Freistein company and Kent Brockman, the purchasing agent. We are aware that existing customer-relationship-management (CRM) solutions like Siebel-Sales or Update.com or SAP can handle this scenario, but do not provide the freedom and flexibility of a wiki system.

If AcmeWear considers to install a wiki system on Peter's laptop computer to support him in his CRM, a few questions will appear:

- How to integrate existing CRM information (telephone numbers, etc) into the wiki?
- How to integrate with the companies product catalogue and enterprise resource planning (ERP) system to get offers, prices, etc?
- Can Peter have reports on all emails and phone calls he does with the customer, right from the wiki?

Integrating external data sources

These are typical questions that are faced by AcmeWear's IT department to make the system more profitable for the company. Using conventional wikis, integration with outside data sources is nearly impossible. Wikis usually allow only links inside the wiki, to information that was already entered. In a company scenario, where not all information is kept in the wiki and instead is spread over the e-mail system, ERP and others, a wiki has to integrate. Also, if the wiki is personal and not a company-wide and shared information source, Peter will have less motivation to enter everything starting

with product codes and ending with customer's telephone numbers. Semantic wikis, as shown in the previous section, offer a solution to the problem of data integration. The straightforward approach is to build adapters that convert the existing data sources into the RDF representation and integrate these into the wiki. But this leaves us with the problem of adapters and integrating the adapters. A Semantic Desktop [8] allows the integration of various data sources. Using this approach, all available data sources would be first integrated to a Semantic Desktop data integration framework and then the semantic wiki would use the Semantic Desktop to access the information. Introducing the layer in-between allows developers and IT departments to concentrate on providing adapters that bring company information sources into the Semantic Desktop. Our own approach for data integration using heterogenous data sources is described and discussed in [11]. Data sources can be either treated as virtual RDF graphs or can be buffered completely in RDF databases, the first approach being a bigger effort. The work by Bizer and Seaborne about adapting SQL databases [2] gives light on how to integrate large SQL databases through virtual graphs and mapping definitions; for web data sources the SECO paper [4] gives indications. Ready built adapters can be downloaded from the sourceforge project *aperture*[5] or from collections like simile's *RDF-izers*[6]. So when these adapters and converters exist, they can be integrated into the Semantic Desktop framework. Serving as a data integration hub, it allows querying of all data sources using standardized protocols and query languages such as SPARQL[7]. SPARQL is the equivalent of SQL applied to RDF — making it the tool of choice for data integration tasks. On this basis, data from different external data sources can be integrated into wiki pages without having to adapt to all the different systems. Common wiki engines provide APIs and extension points, the integration of the Semantic Desktop features will make use of them. A typical extension to wikis is a special tag to link to pages on other wikis or external websites. The Semantic Desktop can be seen like an external website, each resource (identified by a URI) represents an external page. Data about telephone numbers, invoices, products, etc. can be queried using the query language SPARQL. Query results are then rendered into customized wiki pages.

Based on a Semantic Desktop framework, it is possible to answer the first three question of AcmeWear:

- Contact information (telephone numbers, etc) is integrated by converting the existing address book to RDF and providing it as a Semantic Desktop data source.
- AcmeWear's product catalogue and ERP system are also adapted, using dynamic adapters, that can translate questions posed in RDF to retrieve offers, prices, products, invoices, stock levels, etc. The wiki contacts the ERP system via the Semantic Desktop.
- Peter can have reports on emails, product offers, invoices, stock, etc. via automatic queries into the ERP system via the Semantic Desktop data integration hub.

This is the first advantage of the Semantic Desktop: any application (in our case, the semantic wiki) can access information from other data sources. The next innovation of a Semantic Desktop wiki is the way users can author the content. The creation of wiki pages requires that users know the titles of wiki pages (meaning, they usually have to know the exact spelling and structure of the wiki to create links). This manual authoring of wiki pages is a caveat in conventional wikis and remains a problem in semantic wikis.

We will take a look at existing (semantic) wikis and ways to improve them in the following section.

[5]http://aperture.sourceforge.net

[6]http://simile.mit.edu/RDFizers/

4. INTRODUCING SEMANTIC WIKIS

Several wiki implementations exist that implement the basic wiki features and also want to address the problems indicated above. In [6], an overview of semantic wikis and personal wikis is given, resulting in the description of *SemperWiki*, addressing the problems of Semantic Desktop wikis. Lets take a look at the ways metadata is implemented in different wiki implementation in the following.

In most traditional wikis, the idea of metadata typically only appears in a very technical way. For example, in *JSPWiki*[7], metadata is added directly into the wiki text using special tags, and mostly serves the purpose of implementing access control. In *SnipSnap*[8], metadata comes by ways of labels that can be attached to wiki pages which are a kind of categorization scheme.

The semantic wiki *Platypus*[9] adds RDF(S) and OWL metadata to wiki pages. Metadata has to be entered separately from wiki text and relates a wiki page to another resource; thus, metadata can be transformed into a list of *related pages* that can be shown along with the actual wiki page.

The *Semantic MediaWiki*[10] [5] is an extension of *MediaWiki*[11], the software used by Wikipedia. Again, metadata associated to a wiki page may point to other resources, but here, also data literals are allowed. Also, metadata is entered directly into the wiki text, and does not have to adhere to a schema.

Rhizome[12] builds on a framework that adapts techniques such as XSLT and XUpdate to RDF. In essence, RDF is used throughout the framework for almost everything, and RxSLT (an XSLT variant adapted for RDF) is used for transforming queries' results to HTML or other output formats. Page metadata has to be entered separately from the page. While the approach is very interesting from a technical point of view, the current implementation requires a lot practice with the underlying techniques.

So, current semantic wikis are lacking concerning extraction and usage of metadata—users have to enter metadata manually, and the only means of querying the metadata is either very simple queries built with a user interface or very complex queries entered manually as text in a query language.

Let us take a look at how better metadata handling and exploitation could be achieved.

Coupling semantics with the wiki's contents

In a standard wiki, certain words (written by the user in "CamelCase" or highlighted using special characters) indicate that these words get linked to the wiki page describing the corresponding topic.

In our semantic wiki prototype *Kaukolu*[13], we take the occurrence of keywords as evidence that semantic concepts and relations occur in the text.

For example, imagine we are editing a page named *MillersHomepage* containing the text *Mysoftware is written in Perl*. The wiki links *Mysoftware* to some RDF classes' instance, *written in* to some

[7]http://www.jspwiki.org/

[8]http://snipsnap.org/

[9]http://platypuswiki.sourceforge.net/

[10]http://semediawiki.sourceforge.net/

[11]http://mediawiki.sourceforge.net/

[12]http://rx4rdf.liminalzone.org/Rhizome

[13]http://kaukoluwiki.opendfki.de/

RDF property, and *Perl* again to an RDF instance, so the wiki concludes that these three RDF resources occur, and the user may build an RDF triple of the three resources recognized. This new triple is independent of the page and the user that created it.

The list of keywords is generated manually—these "semantic" keywords link to semantics similar to normal WikiWords/page titles linking to wiki pages[14].

Providing a formalization of a text in this way is quite an expensive process, as both vocabulary and resources must be created and looked up again when creating instance data (the formalized knowledge). However, partly this is almost the same problem that occurs when writing standard wiki pages: You have to either look up or remember the page's name that describes the concept you are currently talking about. Typically, one has to stop writing numerous times and start searching for the proper page name then.

This problem can be partly solved with several techniques, for example one can use features such as autocompletion (using for example ECMAScript) which should simplify the formalization process greatly from the user's point of view.

Creating RDF instances is only part of the problem. In order to build an ontology–enabled wiki, conformance of the RDF instances to an RDF Schema should be checked, possible properties should be proposed, and ultimatively one should be able to create new ontologies. Currently, Kaukolu supports none of these features truly—building RDF Schemas is possible only because RDF Schemas are formulated as RDF (which can be created by Kaukolu). However, no special support for building RDF Schemas is available. In the future, we plan to support the user when building RDF instances by listing properties defined in the RDFS class (this represents a kind of semantic *TODO* editing help). Checking instances against their schema and marking consistent versions of the wiki would be another step in the direction of an ontology–enabled wiki.

Building metadata queries

So now that we have metadata, we need ways to exploit it. Most semantic wikis support very simple queries ("List all resources *this* resource is related to") and hand–crafted arbitrarily complex advanced queries. A simple way to formulate queries of medium complexity would be desirable. One solution would be to assist the user by keeping track of the link types the user traversed when using the wiki, offering these types again when entering the query. Also, query refinement by ways of taking user feedback concerning query results into account could be implemented.

5. CONCLUSION AND FURTHER WORK

Semantic wikis will allow a combination of best breeds: the ease of authoring content known from wikis and the explicit semantic information of the semantic web. When semantic wikis are employed on the Semantic Desktop, they can be integrated into personal information management (PIM) scenarios.

First, the integration of diverse external data sources like ERP and other PIM systems allows the user to reuse existing information from systems like MS-Outlook in his personal semantic wiki. Then, complex queries can be formulated and their results displayed inside wiki pages—allowing the user to get an integrated view of information in the wiki. In the end we showed approaches that improve the user interface of such applications.

[14]Building the keyword list manually has its drawbacks. We intend to experiment with techniques known from natural language processing for automatic keyword extraction as well as incorporating linguistic ontology annotations [1] which also support multilinguality.

We plan to improve our semantic wiki prototypes and integrating them with our Semantic Desktop framework *gnowsis*. At the moment, our experiments have been conducted using three separate prototypes that emphasize different aspects. First, a wiki integrated with an early gnowsis version in 2003 contained an integrated web interface. Second, the current 2005 version of gnowsis shows a Java Swing GUI for the wiki that supports drag and drop, and semantic search capabilities. Finally, Kaukolu is our prototype for a completely RDF based semantic wiki, but offers no semantic desktop integration currently. Integrating these three projects will be the next challenge. A downloadable example application will be part of this.

Our aim is to create a *personal semantic wiki* that still provides all benefits known from wikis: ease of use, low entry barrier, free in form and semantics. Beyond the basic features, users can add explicit semantics to their wiki pages, annotating information inside the wiki as well as resources of their desktop data sources. These extended annotation possibilities and the extended querying and reporting functions create a new form of wiki: the personal semantic wiki.

6. REFERENCES

[1] BUITELAAR, P., SINTEK, M., AND KIESEL, M. Integrated representation of domain knowledge and multilingual, multimedia content features for cross-lingual, cross-media semantic web applications. In *Proceedings of the ISWC 2005 Workshop on Knowledge Markup and Semantic Annotation* (2005).

[2] C. BIZER, A. S. D2rq-treating non-rdf databases as virtual rdf graphs. In *Proceedings of the 3rd International Semantic Web Conference (ISWC2004)* (2004).

[3] DECKER, S., AND FRANK, M. The social semantic desktop. *WWW2004 Workshop Application Design, Development and Implementation Issues in the Semantic Web* (2004).

[4] HARTH, A. Seco: mediation services for semantic web data. *Intelligent Systems, IEEE Volume 19*, Issue 3 (May-June 2004), 66 – 71.

[5] KRÖTZSCH, M., VRANDECIC, D., AND VÖLKEL, M. Wikipedia and the semantic web — the missing links. In *Proceedings of Wikimania 2005* (JUL 2005), Wikimedia Foundation. http://www.aifb.uni-karlsruhe.de/WBS/mak/pub/wikimania.pdf.

[6] OREN, E. Semperwiki: a semantic personal wiki. In *Proceedings of the 1st Semantic Desktop Workshop at the ISWC2005* (2005).

[7] PRUD'HOMMEAUX, E., AND (EDTS), A. S. Sparql query language for rdf. W3c working draft, W3C, 2005. http://www.w3.org/TR/rdf-sparql-query/.

[8] SAUERMANN, L. The gnowsis-using semantic web technologies to build a semantic desktop. Diploma thesis, Technical University of Vienna, 2003.

[9] SAUERMANN, L., BERNARDI, A., AND DENGEL, A. Overview and outlook on the semantic desktop. In *Proceedings of the 1st Workshop on The Semantic Desktop. 4th International Semantic Web Conference (Galway, Ireland)* (2005), S. Decker, J. Park, D. Quan, and L. Sauermann, Eds.

[10] SAUERMANN, L., AND SCHWARZ, S. Introducing the gnowsis semantic desktop. In *Proceedings of the International Semantic Web Conference 2004* (2004).

[11] SAUERMANN, L., AND SCHWARZ, S. Gnowsis adapter framework: Treating structured data sources as virtual rdf graphs. In *Proceedings of the ISWC2005* (2005).

[12] WIKIPEDIA. Wikipedia, the free encyclopedia. http://en.wikipedia.org/.