

HYBRID WIKIS: EMPOWERING USERS TO COLLABORATIVELY STRUCTURE INFORMATION

Florian Matthes, Christian Neubert and Alexander Steinhoff

*Software Engineering for Business Information Systems (sebis), Technische Universität München
Boltzmannstr. 3, 85748 Garching, Germany*

Keywords: Enterprise wikis, Semantic wikis, Enterprise information management, Enterprise 2.0, Web collaboration.

Abstract: Wikis are increasingly used for collaborative enterprise information management since they are flexibly applicable and encourage the contribution of knowledge. The fact that ordinary wiki pages contain pure text only limits how the information can be processed or made accessible to users. Semantic wikis promise to solve this problem by capturing knowledge in structured form and offering advanced querying capabilities. However, it is not obvious for business users, how they can benefit from providing semantic annotations which are not familiar to them and often difficult to enter. In this paper, we first introduce the concepts of hybrid wikis, namely attributes, type tags, attribute suggestions, and attribute definitions with integrity constraints. Business users interact with these concepts using a familiar user interface based on forms, spreadsheet-like tables, and auto-completion for links and values. We then illustrate these concepts using an example scenario with projects and persons and highlight key implementation aspects of a Java-based hybrid wiki system (Tricia). The paper ends with the description of practical experiences gained in two usage scenarios, a comparison with related work and an outlook on future work.

1 MOTIVATION AND PROBLEM STATEMENT

To keep pace with the growing amount of digital information that has to be managed, enterprises have to adopt new tools and methods (Edmunds and Morris, 2000). In the recent past, wikis are increasingly used as lightweight shared knowledge repositories that allow to collaboratively gather and consolidate information that was previously scattered across emails, files on personal computers and paper documents (Stocker and Tochtermann, 2009). Having this information integrated in a central place, being able to search it and to connect related pieces of information with hyperlinks is in fact a major advance.

However, with a growing knowledge base soon the demand arises to access information in more structured ways that classical wikis do not support. For example it is not possible to query a wiki for a company's research projects that started in the year 2010 or to export data about these project to a spreadsheet. So even if only rudimentary structured querying functionality is required, the enterprises have to resort to separate applications, usually specialized to manage information of particular domains (like em-

ployees, projects or customers) or they have to develop customized solutions. In both cases the advantages of storing information in a central repository are lost.

Technically, semantic wikis are promising tools to tackle this problem. They allow to combine the textual content with structured data. Typically, users have to provide this data in the form of semantic annotations to wiki pages or parts thereof. The structured part of the information in the wiki can then be queried similar to the contents of a database. However, in practice they are rarely used as a general purpose tool that dynamically adapts to new needs. In contrast semantic wikis often are pre-configured by experts to solve rather specific problems. Although, from a theoretical point of view, they can be used to structure arbitrary information, there are several barriers users are facing when editing content:

- Usually, a special syntax has to be used to add semantic annotations which makes it difficult and cumbersome to edit structured content.
- The modelling concepts are not familiar to the users.
- It is not obvious for users how they can benefit

from providing semantic annotations.

This paper describes a novel approach to mitigate these problems.

In Section 2, our approach of so-called *hybrid wikis* is presented and illustrated using an example scenario. The term ‘hybrid’ expresses that a subset of the features of semantic wikis are integrated into classic wiki software. The main modelling concepts are described and the limitations of the approach are discussed. Important and interesting technical aspects of the implementation are covered in Section 3. Section 4 contains two case studies demonstrating the practicability of the approach. In section 5, we give an overview of related work and highlight some examples of semantic wikis that use different approaches to facilitate data entry. The paper concludes with a short summary and an outlook on further research and planned improvements of the prototype.

2 HYBRID WIKIS

In this section, our approach to the problem of facilitating the management of structured data for non-expert users is presented. Before we describe the structuring mechanisms that are available in the current software implementation on a conceptual level, we first give an overview of our goals, assumptions and motives that guided its development. Finally, we demonstrate how the modelling concepts can be applied in practice using a small example scenario.

2.1 General Ideas and Design Rationale

In the development of hybrid wikis, our primary goal is to lower the barriers for non-expert users, that have been described in Section 1. This means for using the wiki neither special knowledge of wiki syntax or modelling concepts should be required nor should the user be forced to learn a query language to utilize the structured part of the wiki content. We try to find ways to enable all users to enter structured data, in contrast to a two-phase process where unexperienced users enter textual content that is later enriched with annotations by experts.

From our point of view, all attempts to translate between the expressivity of classic, established semantic web formalisms (like RDF¹ or OWL²) and the user by the means of new user interfaces are unsatisfactory. Our approach primarily focuses on the user

¹<http://www.w3.org/RDF/>; visited on May 1st 2011.

²<http://www.w3.org/2004/OWL/>; visited on May 1st 2011.

and accepts that there are limitations in the complexity of modelling concepts users can be expected to understand. We try to start from lightweight structuring concepts and metaphors that users are familiar with and then, in a second step, we examine how the data the users provide by these simple means can be exploited by the system to offer features that usually require a formally defined data model.

Therefore, we rely mainly on simple keyword-like annotations of wikipages, dynamically compiled and easily extensible forms for data entry, and the presentation of data in automatically generated tabular views. In turn, we try to avoid the notion of semantic annotations being something that is optionally appended to pure text content and that is defined in a separately maintained ontology or schema. Instead we attempt to allow the user to implicitly provide semantics by filling data in particular fields of a form or a table, by optionally creating new such fields on demand, and by the way the data is queried and displayed in different contexts.

Since we do not require the users to explicitly maintain a data model, we focus on dynamically mining the wiki to ‘guess’ the model and provide users with input options to guide them towards a consistent data model and vocabulary. However, advanced users can impose a schema and define certain integrity constraints.

For querying and browsing we provide a general search interface that allows a faceted drill-down based on the structured contents of the wiki pages. Furthermore, contents can be accessed in a spreadsheet-like tabular form. Our assumption is that users feel familiar with this representation and in consequence are less inhibited in manipulating the data.

2.2 Structuring Concepts

Hybrid wikis were implemented as an extension to the wiki component of the commercial enterprise collaboration platform *Tricia*³. Wiki pages basically consist of a name and some rich-text content. Pages can be organized using text labels (*tags*) and page/sub-page relationships. We added two means for structuring the information on a wiki page: *attributes* and so-called *type tags*. Both can be used in combination or independently. They are described in the following.

2.2.1 Attributes

In their simplest form, attributes are key-value pairs that can be added to wiki pages. They consist of an attribute name – the key – and a value, being either a

³<http://www.infoasset.de/>; visited on February 1st 2011.

short text literal or a link to another wiki page. Attributes do not represent metadata but constitute the structured part of the content. For the user, the attributes are presented in a box containing a list of key-value pairs at the right border of the page (see Figure 1). The appearance is inspired by a kind of templates widely used in Wikipedia⁴ – and thus the MediaWiki software⁵ – to structure the contents of pages describing objects of the same type, like for example cities, countries or planets. Since the box forms the structured part of the page, it can also be compared to the so-called fact box of Semantic MediaWiki, that summarizes the facts being expressed by annotations in the text.

However, in hybrid wikis neither does the attribute box reflect facts defined somewhere else nor is the selection of attributes specified by a template. This means that new attributes can be added to individual pages simply by adding a new entry to the list of attributes. It is possible to assign multiple values for one attribute. The values are ordered and can be a mix of literals and links.

In order to encourage users to structure the content they create, we strived for a convenient user interface: When viewing a wiki page, the user is offered a selection of names of attributes she might want to add. These suggestions are shown together with the already assigned attributes at the bottom of the attribute box so that the user only has to provide proper values. The list of suggestions consists of the most frequent attribute names used on similar pages (we will cover page similarity later). Furthermore, the input fields for attribute names and values display suggestions as the user is typing. When an attribute value is typed, the respective attribute name is factored in to improve the quality of the suggestions. This makes it comfortable to contribute structured content and additionally fosters consistent usage of terms.

To avoid redundancy, and thus inconsistencies, and to foster navigation in the wiki, references from other pages are shown in an additional ‘references’ section of the attribute box. For example if a page for a country S references the capital M , i.e., it contains an attribute with one value being a link to M , the page for M will show an additional attribute entry ‘capital of S ’ having the value M . The references section of the attribute box is similar to the incoming and outgoing links as displayed in the KiWi system⁶.

⁴<http://www.wikipedia.org>; visited on February 1st 2011.

⁵<http://www.mediawiki.org>; visited on February 1st 2011.

⁶<http://www.kiwi-project.eu>; visited on February 2st 2011.

2.2.2 Type Tags

Type tags allow the user to make a statement about the type of the object being described on the page. They are shown at the top of the attribute box as shown in Figure 1. Like for the normal tags used in the wiki (i.e., arbitrary text labels assigned to pages for categorization), users may choose an unlimited number of terms they consider appropriate.

One of the most important functions of type tags is to provide a reliable indicator of page similarity for the automatically generated attribute suggestions described above. They determine which additional attributes the user is recommended to fill in.

In the simplest case, exactly one type tag is assigned to a page. The system then determines the set of attributes used in combination with this type tag and displays the most frequent attribute names as suggestions.

In the case of multiple type tags, first the attributes of pages having at least one of them assigned are determined. Then, to select the most relevant attribute names for the current page, not only the frequency of occurrence is considered, but additionally, attribute names are favoured when they occur together with many or all of the given type tags.

In effect this makes it very convenient to create a new page of a type previously used somewhere else in the system. Once a type is assigned, providing the attributes is hardly more demanding for the user than filling out a form. From this point of view, type tags can be considered a more flexible alternative to templates.

It is important to note that on the one hand it was consciously avoided to force users to explicitly create relationships between type tags and attribute names as it would be done when defining a template. On the other hand, for experienced users it is still possible to assign a list of attribute names to a type tag. These attribute names are then highlighted to the user whenever the respective type tag is used. For each attribute name associated with a type tag it is further possible to specify integrity constraints like the type of value, the number of values or allowed value ranges.

Type tags are also used to generate lists of pages describing objects of the same type. While this is possible with standard tags as well, for example by searching for all pages tagged ‘university’, the results are more precise when type tags are used: users might use the same tag ‘university’ to categorize pages describing for example people working at a university, software products targeted at an academic audience or the concept of a university. Thus, type tags allow the user to make an important distinction when tagging



Figure 1: Type tags and attributes in the context of a wiki page.

wiki pages by stating that the tagged page describes an individual instance of the respective type – in contrast to vaguely relating it to a broad topic.

Retrieving a list of pages for a specific type tag can be achieved by simply clicking on the tag. If the pages contain attributes, they are displayed in a tabular view (as shown in Figure 2). The table can be sorted by all columns. Using the faceted search functionality of the Tricia platform it is further possible to combine type tag and attribute filters. In this way it is possible to define very specific subsets of the wiki contents. The results can optionally be presented in tabular form embedded in any wiki page.

2.3 Limitations

Since the expressivity of established semantic web technologies is sacrificed in favour of a better user experience, modelling capabilities of hybrid wikis are constrained. In particular it is not possible to explicitly define type-subtype relationships between type tags. If a type tag is the generalisation of another type tag in the system, it has to be manually ensured that each page with the more specific type tag also has the more general type tag assigned. However, with very little effort it is possible to configure a wiki page that can be used as a dashboard to monitor violations of such integrity constraints.

For attributes it is not possible to specify any se-

mantic relation. Since no reasoning capabilities are provided by the system it would have no effect to add properties like symmetry or transitivity. However, as mentioned above, attributes containing links are also visible on the target pages as a reference. This means that relationships between two pages are always owned by one of them and it is often not trivial to determine which one should be the owning side (does capital city link to country or country to capital?). This depends on the multiplicity of the relation and in particular cases on the access rights of the pages. However, in order to lower the barriers for entering structured data, the fact that this decision possibly has important implications is not explicitly communicated to the user. In contrast, it is relied on the ability to invert these links later when it becomes necessary.

Finally, querying capabilities are limited in so far that results can only be filtered according to the very attributes of pages. It is for example not possible to express a query targeting all pages having the attribute 'Owned by' set to a page with the type tag 'Company'. Requiring that the attribute points to a specific page is of course possible. From a technical point of view, attribute (and type tag) filters could be flexibly combined using boolean operators. However, to keep the user interface simple the current implementation supports only conjunction of filters. From our experience this is sufficient for browsing the wiki contents.

Wikis » sebis Public Website » Type Tags » research project

Wiki Pages with Type Tag **research project** in sebis Public Website

Showing 1 to 10 of 11 entries

	Team members (11)	Research area (11)	Project start (11)	Contact (10)	Partners (4)	Sponsors (4)	Project end (2)	Name (1)
EAM for the German Public Sector	Sabine Buckl Prof. Dr. Florian Matthes	EAM	2010	Sabine Buckl	Bundesministerium des Inneren (BMI)			
EAM in the Context of Mergers and Acquisitions	Christopher Schulz Andreas Freitag	EAM	2009	Christopher Schulz Andreas Freitag				
EAM Pattern Catalog	Sabine Buckl Dr. Josef Lanke Christian M. Schweda	EAM	2008	Sabine Buckl Christian M. Schweda			2010	
EAMML - Enterprise Architecture Management Method Library	Christian M. Schweda	EAM	2009	Sabine Buckl		Bausparkasse Schwäbisch Hall BOC Deutschland	2010	
Hybrid Wikis	Christian Neubart Thomas Buchner Christian M. Schweda Sabine Buckl	SoSo EAM	2008	Christian Neubart				

Figure 2: Tabular view of wiki page attributes.

2.4 Example Scenario

In the following, we illustrate the modelling capabilities of hybrid wikis taking the example of a small company’s intranet wiki. Among other things, the wiki is used for gathering the knowledge about projects and people relevant for the company. We assume that while there may be many pages holding information about a person or project, there is one dedicated page for each such entity that can be considered the primary page which holds the basic information about it and optionally links to pages with more specific information. We further assume that in the beginning, there is only little content in the wiki and no type tags and attributes are used.

As the number of projects increases over time, there is a growing demand for a more structured view on project related wiki content. Attributes for project start and end dates are thus added to the respective pages and the type tag ‘Project’ is assigned. Having marked all project pages with the type tag, an overview table of all projects is instantly available showing sortable columns for the date attributes. Attribute values can be changed directly in this overview table. By this means, consolidating the project data, i.e., adding missing information or standardising the representation of attribute values, is facilitated.

Let’s assume many of the employees have created profile pages for themselves in the wiki to provide so-

me information about their specific skills and experiences. Some of them independently start to add attributes to the project pages to express their relationship with these projects, for example that they were members of the project team or project managers. In the beginning, people use different terms to describe their roles. As these inconsistencies become visible in the overview table, they are quickly harmonized by the wiki users. It is not necessary to navigate to the respective pages but the table cells can be edited directly. As a result, if somebody now creates a new wiki page and assigns the type tag ‘Project’, she is offered to provide values for the attributes ‘Start date’, ‘End date’, ‘Project manager’ and ‘Team’. If she now adds a link to her profile page to the ‘Project manager’ attribute, this reference will be automatically visible on her page in the ‘References’ section of the attribute table. A new entry ‘Project manager of’ with a single value being the link to the project is displayed. If she is already manager of other projects, the list will simply be extended by one entry.

Starting from this basic schema, the company can further adapt it to new needs: Besides adding more attributes, the existing ones can be refined. The attribute ‘Project manager’ can be made mandatory for the type tag ‘Project’ so users are additionally reminded to provide this attribute. If it is omitted, the page is flagged as invalid. It can be further specified that only a single link to a page having the type tag

‘Person’ is accepted as a value.

New type tags can be added to distinguish different types of projects like ‘Research project’ or ‘Internal project’. On the one hand, this allows the generation of lists and tables containing only the respective subset of the projects, on the other hand, the system is supported in offering the user more relevant attribute suggestions when editing structured data. For example for a research project the attribute ‘Field of research’ could be suggested whereas attributes only relevant for internal projects are not shown. It is also possible to add a temporal dimension to the project types by adding type tags like ‘Project in preparation’, ‘Current project’ or ‘Completed project’. Using type tags instead of a status attribute has the advantage that again attribute constraints can be related to the types. For example each page of a completed project can be required to contain the actual end date of the project.

3 IMPLEMENTING HYBRID WIKI

The concept of hybrid wikis is developed by members of our chair since 2009. The system is built on the experiences made with classic and semantic Wiki technology as well as integrated Enterprise 2.0 platforms and it is currently realized based on the modelling frameworks provided by Tricia enabling the model-driven development of web cooperation systems (Büchner et al., 2010). Hybrid wikis extend the Tricia wiki functionality by a few mechanisms for classification, linking, consistency checking and visualization which can be flexibly combined, as described in Section 2.

The data model underlying the implementation of hybrid wikis is shown in Figure 3. The illustrated concepts are currently mapped to a SQL database⁷ by means of Tricia’s object relational mapping mechanisms. We will explain the data model of the hybrid wiki elements and some impacts on the system behaviour in the following.

For each wiki page multiple attributes can be assigned. Each of the attributes has a non-empty list of ordered values being either string- or link-values. A wiki page can have multiple tags assigned. A tag is a simple character sequence and either a normal tag or a type tag. WikiPage, TypeTag, Attribute and Value constitute the set of concepts which are provided for structuring data in hybrid wikis.

TypeTagDefinition, AttributeDefinition,

⁷SQL is the default storage implementation, but other implementations are also supported, e.g., NoSQL storages

and Validator are additional concepts to enable users to specify integrity constraints. A type tag definition contains the set of attributes that the users are urged to provide in combination with the respective type tag. These attribute definitions can be further used to restrict the range of attribute values with validators.

A type tag definition is loosely coupled to a (type) tag by name, the same applies to attribute and attribute definition. The fact that an attribute definition x is bound to a type tag definition y means that on pages with a type tag y an attribute suggestion x is shown, which takes precedence over the default attribute suggestion mechanism, i.e. it is not required that a page having both, tag x and attribute y , assigned exists in the system. Thus, new attributes can be seeded top-down over typed pages by means of suggestions.

Furthermore, for particular attribute definitions, integrity constraints can be specified using validators. The system currently supports three kinds of validators: the EnumerationValidator checks if the values of an attribute conform to a given set of string- and link-values. The HyperlinkValidator checks if the values of an attribute are hyperlinks, additionally it can be defined that these hyperlinks have to be links to a wiki page with specific type tags. A MultiplicityValidator allows to specify how many values an attribute should have, e.g., at-least-one, at-most-one, exactly-one.

Validators apply when a wiki page is displayed. That means, if a wiki page has attribute x , a type tag y , and a type tag definition for y exists having an attribute definition with name x and some validators, these validators will be matched with the current values of attribute x . In case of validation failure the validationMessages of the validators are shown in the context of attribute x . The situation that two type tags specifying contradicting validators are assigned to a wiki page is currently not prevented by the system and users have to solve this conflict manually.

Beside showing validation messages for wiki page attributes, the integrity constraints are also used to improve the input support for values when an attribute is edited. For instance, if an enumeration validator applies to an attribute in the value auto-completion control only the elements specified in this validator (e.g., {‘open’, ‘closed’}) are offered.

Since hybrid wiki validators never prevent users from entering values that contradict the specified validation rules, we call them *soft validators*. The system assists users in systematically searching for contradictions. As a side-effect, it is possible to import information from arbitrary data sources to Tricia, e.g., from a Microsoft Excel data sheet, without being re-

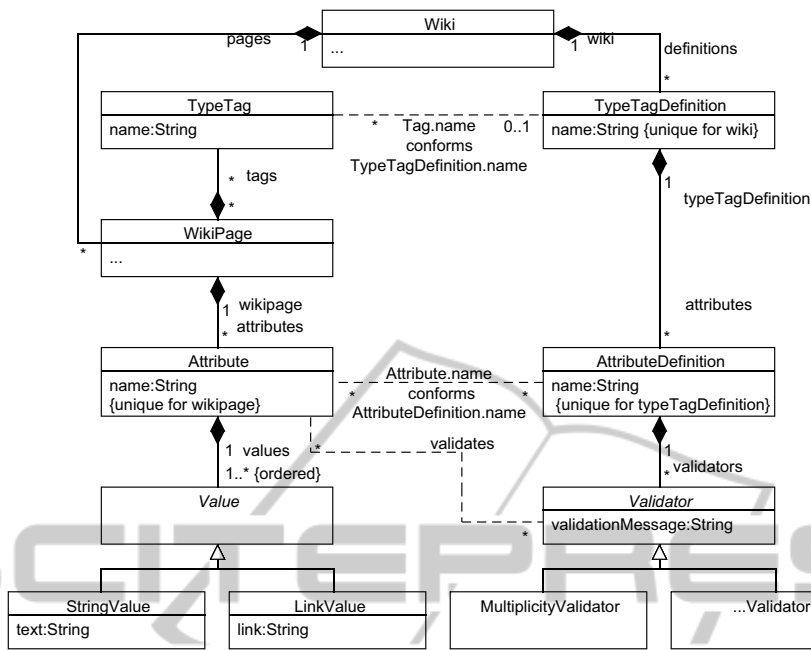


Figure 3: The Data model of Hybrid Wikis.

strained by hard integrity constraints. Any conflicts with the schema can be fixed after the import.

Since the main purpose of Tricia is to enable users to find relevant information within enterprises quickly, the content of all elements (e.g files, wikis, blogs) is indexed by means of the Lucene information retrieval software library⁸. Beside simple text queries, querying for metadata like tags or date of last modification is supported in this way. For the implementation of hybrid wikis Lucene plays a critical role in the dynamic generation of attribute suggestions and overview tables. Lucene supports both features by providing flexible filtering capabilities together with a very fast access to particular fields of the indexed entities. For example to display a table of all pages having a specific type tag, Lucene can efficiently determine the respective set of pages (filtered by the access rights of the user) and provide the set of attribute names used on each page. From the analysis of the frequency of attribute names the selection and order of the table columns is then determined.

It is possible to import and export hybrid wiki data by using the standardized Eclipse Model Framework (EMF) exchange format. Models can be imported and exported as ecore-files, model instances as xmi-files. Furthermore, all hybrid wiki contents can be imported from and exported to spreadsheets (e.g., Microsoft Excel). Additionally, it is possible to visualize hybrid wiki content by means of an external service (System

⁸<http://lucene.apache.org>; visited on February 1st 2011.

Cartography Service⁹) that is accessed via RESTful API calls.

4 PRACTICAL EXPERIENCES

Hybrid wikis are currently evaluated in several industrial and research projects. In the following, we briefly describe the usage scenarios of the hybrid wiki technology in two selected cases. The first is taken from one of our research projects *Wiki4EAM*¹⁰ where hybrid wikis are applied for the collaborative documentation of system landscapes in enterprises in order to evolve models representing these landscapes bottom-up, the second from the application of a hybrid wiki as a simple issue tracker in a German software development company, the InfoAsset AG¹¹. For each of these cases we sketch the models being developed bottom-up via attribute and type suggestions as well as the (soft-) constraints which were defined afterwards.

⁹<http://www.matthes.in.tum.de/wikis/sebis/syca>; visited on February 1st 2011.

¹⁰<http://www.matthes.in.tum.de/wikis/sebis/wiki4eam>; visited on February 1st 2011.

¹¹<http://www.infoasset.de>; visited on February 1st 2011.

4.1 Wiki4EAM Community

In the Wiki4EAM community, founded in December 2010 at TUM, members share their experiences regarding the use of a hybrid wiki in the context of enterprise architecture management (EAM). A prerequisite for adequate management of the enterprise architecture (EA) is to capture its current state in a model. Since the knowledge of all different elements to be considered in the model (e.g., business processes, applications, organizational units) is spread over all the different stakeholders in the company, the documentation of the current state of the EA remains a challenging task. Hybrid wikis enables these stakeholder to document the particular parts of the EA they are responsible for by means of (hybrid) wiki pages. Thereby the data model emerges bottom-up by creating, editing, linking and structuring these particular pages. A preliminary data analysis in two German companies from December 2010 to January 2011 showed the following model evolution:

- Company A: 12 concepts with 63 attributes (textual and links), 100 wiki pages, 2 (soft-) constraints were created within two weeks by four participating stakeholders (editors)
- Company B: 18 concepts with 60 attributes (textual and links), 120 wiki pages, 28 (soft-) constraints were created within one month by five participating stakeholders (editors)

Although these numbers are not a founded empirical evaluation, they allow to assume that in a relatively short period models can emerge bottom-up by the collaborative documentation of the particular elements by using lightly structured wiki pages. Indeed, feedback from Wiki4EAM community members indicates that a hybrid wiki is well suited in cases when the target model is not completely known. They also confirmed that attribute suggestions facilitate the evolution of the information model.

4.2 InfoAsset Bugtracker

The InfoAsset AG providing the Tricia platform for commercial use, uses hybrid wikis among others things for its issue management. For each issue a hybrid wiki page with tag ‘unprocessed’ and type tag ‘issue’ is created. All unprocessed issues are shown in a list on the Tricia developer dashboard, that is built only using standard features of hybrid wikis. This list is dynamically generated by using an embedded query which filters all wiki pages having the tag ‘unprocessed’ and type tag ‘issue’ assigned. When a new entry appears in this list, i.e. the underlying embedded query yields a new search hit, the developers are

informed by means of a new entry in an RSS-Feed. A responsible person processes these new entries and categorizes them by assigning additional type tags. Currently two kinds of additional type tags are mainly used: ‘Bug’ and ‘Change request’. Furthermore, a person is assigned responsibility for processing this issue by setting the attribute ‘assigned to’ to the respective name and the tag ‘unprocessed’ is deleted in order to remove this issue from the dashboard list. The responsible person is in turn informed by an RSS-Feed. The complete data model as used for the change management is shown as an UML class diagram in Figure 3.

The schema as shown emerged bottom-up by management of the particular wiki pages. Only the following constraints were defined afterwards top-down by means of type tag definitions:

- enumeration types (Status, Priority, Effort), in the beginning these values were plain strings
- ‘exactly-one’ cardinality for the relationship ‘assigned to’ between ‘Issue’ and ‘Person’ instances

Note, that the cardinality in the UML diagram of Figure 3 is given with 0..1. This is due to the fact that new reported issues do not have a responsible person assigned. For these instances a warning message is shown indicating that there should be at least one person defined.

5 RELATED WORK

Wiki templates enable authors to reuse content structures among wiki pages. In (Haake et al., 2005), the authors discuss the need for structure in wikis based on templates, in (Di Iorio et al., 2008) different wiki templating approaches are compared to each other. Although hybrid wikis explicitly do not support templates, attribute and type (-tag) suggestions based on a statistical analysis of typetag and attribute combinations enables authors to reuse well established type structures, similar to templates.

The Semantic MediaWiki project is the most prominent example in the category of semantic wikis. This project adds database-like structuring and querying capabilities on top of an existing wiki, without requiring users to develop or adhere to a rigid database schema when authoring content (Krötzsch et al., 2006). This project also includes various features for browsing, searching, and aggregating the wiki’s content as well as embedding queries in the wiki pages. Additionally, Semantic Media Wiki provides a model export to the standardized format

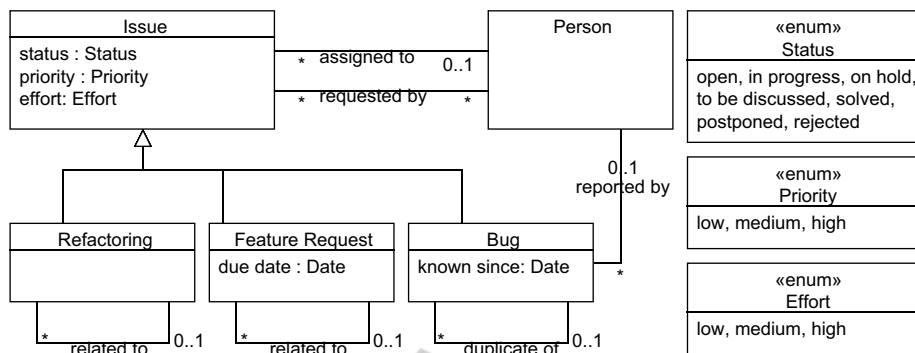


Figure 4: Emerged Concepts of the InfoAsset Bugtracker.

OWL/RDF. Unlike in hybrid wikis, in Semantic MediaWiki meta information can only be added to the pages by directly editing the markup in wiki syntax.

This issue is addressed by Semantic Enterprise Wiki (SWM+)¹², a set of open-source extensions to the Semantic MediaWiki. In this approach, meta data (properties) can be defined by means of a graphical user interface, called ‘semantic toolbar’, so users are not forced to write semantic annotations manually. For the classification of pages in Semantic Enterprise Wiki the concept of categories is used. Categories and properties are shown as a list in the semantic toolbar. The approach differs from hybrid wikis as follows:

- A separate annotation mode (semantic toolbar) is needed to enter structured information.
- Users have to create and edit both, the content of the wiki page and the semantic annotations. Users also have to be aware of content and annotations being synchronized, which is an laborious and error-prone task.
- No attribute (properties) suggestions based on the types (categories) are provided.

AceWiki is a semantic wiki using controlled natural language for ontology management (Kuhn, 2008). In (Paschke et al., 2009) a lightweight approach for editing ontologies is introduced. Like hybrid wikis, both approaches try to facilitate structured data entry. However, in AceWiki this is achieved by natural language processing, in the second case by introducing a lightweight ontology editor.

The open source project TWiki¹³ and its fork Foswiki¹⁴ try to combine the advantages of wikis and database systems by allowing the user to attach data records to wiki pages. Although the schema can be changed at runtime, the effort is considerably higher

¹²<http://wiki.ontoprise.de>; visited on April 27th 2011.

¹³<http://twiki.org>; visited on April 27th 2011.

¹⁴<http://foswiki.org>; visited on April 27th 2011.

than for hybrid wikis and the usage of a special wiki syntax is required.

The research prototype SnoopyDB (Gassler et al., 2010) focuses particularly on recommending attributes to users when they are entering data. Types cannot be specified at all but all attributes are suggested based on attributes of other pages. This makes it very difficult to generate tabular overviews of pages covering certain kinds of things.

MokiWiki (Ghidini et al., 2009) is a plain modelling wiki, used to model the constituent parts of an enterprise collaboratively. With hybrid wikis we focus on facilitating collaborative data and information management within enterprises. Information models emerge bottom-up as a by-product.¹⁵

In sebis Enterprise 2.0 Tool Survey (Büchner et al., 2009) the functional capabilities of integrated E2.0 platforms are compared resulting in a multi-dimensional classification and evaluation framework. Especially the survey 2010 includes functional aspects regarding the ‘Structuring of Content’ as well as ‘Templates for Structured Content’. The fact that leading social software vendors include capabilities for structuring contents indicates an increasing demand in enterprises.

6 SUMMARY & OUTLOOK

In this article, we introduced hybrid wikis, a lightweight approach for data and information management within enterprises facilitating structuring of contents for business users. The main purpose of hybrid wikis is to avoid requiring users to learn complex semantic languages for adding structure to

¹⁵While hybrid wikis are not mainly built for modelling purposes, the models emerging through data management in hybrid wikis can also be used to represent the particular enterprise constituents (c.f. (Buckl et al., 2009) and Section 4).

wiki pages. Instead we provide an easy way for structured data entry with a small set of elements, namely attributes (key-value-pairs) and type tags. Furthermore, attribute suggestions encourage users to provide these structuring elements. Suggestions and auto-completion foster a common vocabulary, so users immediately benefit from the structure through better search capabilities. By means of these mechanisms complex structures emerge bottom-up by the management of particular data sets. To define integrity constraints we introduced the concepts of validators, attribute definitions and type tag definitions. They additionally can be used to define advanced rules and parts of the data model in a top-down manner.

Furthermore, we described the practical experiences made with hybrid wikis in two concrete projects, one from the industry and one as part of our research endeavours. For the latter we sketched the models emerging bottom-up by means of attribute suggestions as well as the (soft-) constraints additionally defined top-down afterwards.

We see potential to improve and validate our approach in the following ways:

- Compare hybrid wikis to similar wiki-based approaches (c.f. Section 5). For instance, we will analyze the revision history in order to compare the evolution of structure compared to hybrid wikis.
- Provide more validators (e.g., boolean, date), which are the basis for advanced controls facilitating the input of structured data.
- Support advanced visualizations of structured data, which also can be used for navigating the wiki contents.
- Improve the way relationships between pages are handled (c.f. Section 2.3).
- Provide type (tag) suggestions beside attribute suggestions.

REFERENCES

- Büchner, T., Matthes, F., and Neubert, C. (2009). A concept and service based analysis of commercial and open source enterprise 2.0 tools. In *KMIS*, pages 37–45.
- Büchner, T., Matthes, F., and Neubert, C. (2010). Data model driven implementation of web cooperation systems with Tricia. In Dearle, A. and Zicari, R., editors, *Objects and Databases*, Lecture Notes in Computer Science, pages 70–84. Springer, Berlin, Heidelberg, Germany.
- Buckl, S., Matthes, F., Neubert, C., and Schweda, C. M. (2009). A wiki-based approach to enterprise architecture documentation and analysis. In *The 17th European Conference on Information Systems (ECIS) – Information Systems in a Globalizing World: Challenges, Ethics and Practices*, 8. – 10. June 2009, Verona, Italy, pages 2192–2204, Verona, Italy.
- Di Iorio, A., Vitali, F., and Zacchiroli, S. (2008). Wiki content templating. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 615–624, New York, NY, USA. ACM.
- Edmunds, A. and Morris, A. (2000). The problem of information overload in business organisations: a review of the literature. *International Journal of Information Management*, 20(1):17 – 28.
- Gassler, W., Zangerle, E., Tschuggnall, M., and Specht, G. (2010). Snoopydb: narrowing the gap between structured and unstructured information using recommendations. In *Proceedings of the 21st ACM conference on Hypertext and hypermedia*, HT '10, pages 271–272, New York, NY, USA. ACM.
- Ghidini, C., Kump, B., Lindstaedt, S., Mahbub, N., Pammer, V., Rospocher, M., and Serafini, L. (2009). MoKi: The Enterprise Modelling Wiki. In Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvönen, E., Mizoguchi, R., Oren, E., Sabou, M., and Simperl, E., editors, *The Semantic Web: Research and Applications*, volume 5554, chapter 65, pages 831–835. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Haake, A., Lukosch, S., and Schümmer, T. (2005). Wiki-templates: adding structure support to wikis on demand. In *WikiSym '05: Proceedings of the 2005 international symposium on Wikis*, pages 41–51, New York, NY, USA. ACM Press.
- Krötzsch, M., Vrandečić, D., and Völkel, M. (2006). Semantic mediawiki. In Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., and Aroyo, L., editors, *The Semantic Web - ISWC 2006*, volume 4273 of *Lecture Notes in Computer Science*, pages 935–942. Springer Berlin / Heidelberg.
- Kuhn, T. (2008). Acewiki: A natural and expressive semantic wiki. Technical Report arXiv:0807.4618, Department of Informatics University of Zurich. Comments: To be published as: Proceedings of Semantic Web User Interaction at CHI 2008: Exploring HCI Challenges, CEUR Workshop Proceedings.
- Paschke, A., Coskun, G., Luczak-Rösch, M., Oldakowski, R., Harasic, M., Heese, R., Schäfermeier, R., and Streibel, O. (2009). Realizing the corporate semantic web: Concept paper. Technical report, Freie Universität Berlin, Berlin.
- Stocker, A. and Tochtermann, K. (2009). Exploring the value of enterprise wikis: A multiple-case study. In *KMIS*.