Collaborative Web-Publishing with a Semantic Wiki

Rico Landefeld and Harald Sack

Abstract. Semantic Wikis have been introduced for collaborative authoring of ontologies as well as for annotating textual and multimedia wiki content with semantic metadata. In this paper, we introduce a different approach for a Semantic Wiki based on an ontology metamodel that has been especially customized for the deployment within a wiki. For optimal usability client-side technologies for graphical user interface have been combined with a simple and intuitive semantic query language. Single fragments of a wiki page can be annotated in an interactive and rather intuitive way to minimize the additional effort that is necessary for adding semantic annotation. Thus, the productivity and efficiency of a Semantic Wiki system will open up for non expert users as well, which is important for fostering the popularity of Semantic Wiki systems.

1 Introduction

The very first browser to access the World Wide Web (WWW) provided an important function that soon sank into oblivion again: web pages could not only be read, but also written and thus be changed directly. Several years ago, wiki systems [14] picked up that very same idea again by providing the possibility for each visitor to change the content of wiki pages in a simple way. Wiki systems are lean content management systems that administrate HTML documents. The user of a wiki

Rico Landefeld
Jena University Language and Information Engineering (JULIE) Lab,
Friedrich-Schiller-Universität Jena, Fürstengraben 30, 07743 Jena, Germany
e-mail: Rico.Landefeld@uni-jena.de

Harald Sack
Hasso Plattner Institut for IT Systems Engineering, Universität Potsdam,
Prof.-Dr.-Helmert-Str. 2–3, 14482 Potsdam, Germany
e-mail: harald.sack@hpi.uni-potsdam.de

S. Schaffert et al. (Eds.): Networked Knowledge - Networked Media, SCI 221, pp. 129–140.
springerlink.com

system is able to generate or change wiki documents only by using the facilities of a simple web browser. In this way, wiki documents are developed and maintained collaboratively by the community of all users without the need of having specialized IT expertise. Wiki systems don't give formal guidelines for generating or structuring their content. This lack of formal rules might have been responsible for their fast growth of popularity as can be seen, e.g., in the free online-encyclopedia Wikipedia[1] that is one of the most popular websites world-wide. On the other hand, if a wiki system is growing as rapidly as Wikipedia does, lack of formal rules necessitates frequent restructuring to keep the content always well arranged and usable.

Typical wiki systems only provide a limited number of functions for structuring the content. As a rule users create special pages with overviews or class systems for structuring and aggregating the wiki content. But the maintenance of this manually created categorization system becomes rather expensive. Moreover, it stimulates misusage, as e.g., you may find many categories in Wikipedia that have been created to subsume entities that share merely one special feature [23]. Similar problems have been reported for intranet wikis [4]. In general, most of the mentioned problems in wikis can be reduced to the fact that their content is encoded in HTML (Hypertext Markup Language) or some simplified version of it. HTML only formalizes formatting and (limited) structuring of documents without the possibility of formalizing any semantics that is required for automated aggregation and reuse of data.

Semantic Wikis try to combine traditional wiki systems with semantic technology as a building block of the currently emerging Semantic Web [2]. They connect textual and multimedial content with a knowledge model by formalizing the information content of a wiki page with a formal knowledge representation language. In this way, the content of wiki pages becomes machine readable and in some limited sense even machine understandable. Semantic Wikis show one possible way to overcome the aforementioned problems related to traditional wikis in general while at the same time enabling collaborative generation and maintenance of formal knowledge representations (ontologies). But, the arbitrary wiki user usually is not an expert knowledge engineer. Therefore, usability and 'ease of use' become a essential factors for the design of the user interface of a Semantic Wiki.

Current Semantic Wiki projects have chosen different ways to deploy formal knowledge representations within a wiki. From our point of view the ratio of cost and effect for the user is most important. The cost refers to the cognitive and factual work that the user has to invest to generate and maintain semantic annotations. On the other side, the effect subsumes all the advantages that the user might get from a system that deploys this semantic annotation. Cognitive and factual work is mostly determined by the design of the user interface and the underlying ontology metamodel of the Semantic Wiki. At the same time the semantic expressiveness of the annotations determines the efficiency of the achieved functionality. Therefore, the ontology metamodel of a Semantic Wiki always represents a compromise between complexity and expressiveness. In addition, the integration of semantic annotations

[1] `http://www.wikipedia.org`

into wiki systems also demands new concepts of user interaction that help to limit the necessary effort for authoring.

Existing Semantic Wiki systems have several deficiencies: either, their underlying ontology metamodel is mapping elements of the knowledge representation language directly to wiki pages, or they are using a simplified ontology metamodel that results in rather limited semantic functionality. Most projects are based on traditional wiki systems and therefore inherit also their user interaction facilities.

We propose a Semantic Wiki concept that combines the following three concepts:

1. a simplified ontology metamodel especially customized to be used within a wiki system,
2. a WYSIWYG-Editor (What You See Is What You Get) as a user interface for both text- and ontology editing, and
3. preferably a most simple semantic query language of sufficient expressiveness.

A prototype of our Semantic Wiki *Maariwa*[2] has been successfully implemented.

The paper is organized as follows: Section 2 covers related work and in particular discusses different ontology metamodels and user interaction concepts of existing Semantic Wiki systems. In Section 3 we introduce the Semantic Wiki project Maariwa, while Section 4 resumes our results and discusses future work.

2 Related Work

Traditional wiki systems administrate structured text and multimedia content connected by untyped hyperlinks. Semantic wikis complement the traditional wiki concept by providing the ability to capture additional information about the wiki pages and their relations. Besides the extension of traditional wikis with semantic annotation, we also have to consider approaches for collaborative authoring of ontologies based on wiki technology that date back to a time before the Semantic Web initiative even started (cf. [9, 21, 1]). We therefore distinguish two different Semantic Wiki approaches depending on their focus either on textual (or multimedia) wiki content or (formal) knowledge representation. The *Wikitology* paradigm [**?**] refers to wiki systems acting as a user interface for collaborative authoring of ontologies. There, a wiki page represents a concept and hyperlinks between wiki pages represent relationships between concepts. Thus, the wiki system acts merely as tool to author and to manipulate the ontology. In difference, so called *ontology-based wiki systems* are Semantic Wiki systems that focus on traditional wiki content, while using knowledge representations to augment navigation, searchability, and above all reusability of information.

Another differentiating factor can be determined by the adaption of the ontology metamodel for the use within the wiki and the coverage of the underlying knowledge representation languages (KRL). The *ontology metamodel* of a Semantic Wiki defines a mapping between the elements of the KRL and the application model.

[2] Maariwa can be accessed at `http://stemnet0.coling.uni-jena.de/Maariwa/app`

Moreover, it determines the semantic expressiveness of annotation and serves as a basis for querying information. Semantic annotation can be maintained together with or separate from the textual wiki content. [17]. Next, we will introduce and discuss relevant Semantic Wiki implementations and their underlying ontology metamodel.

PlatypusWiki [5] is one of the earliest Semantic Wiki implementations. It maps a wiki system to a RDF (resource description framework) graph [12]. Wiki pages represent RDF resources and hyperlinks represent RDF properties. Semantic annotation is maintained together with the textual wiki content within a separate text field as RDF(S) (RDF Schema) [3] or OWL (Web Ontology Language) [16] in XML serialization format.

Rhizome [20] supports semantic annotations by using a special Wiki Markup Language (WikiML). The entire wiki content including text, structure, and metadata internally is encoded in RDF. Rhizome also allows direct editing of RDF data with an external RDF editor. In contrast to traditional wiki systems, Rhizome supports a fine-grained security model. Beside creation and manipulation of meta data Rhizome does not offer any functionality that utilizes this semantic annotation.

Rise [6] is customized for requirement analysis in the process of software engineering. It is based on an ontology that represents different document types and their relationships. Templates determine structure and relationships of wiki pages that represent instances of a document type. The Rise ontology can be extended by adding new templates to the existing ones. Semantic annotations can be created and edited with an extended WikiML and are used for consistency checks and navigation.

Semantic MediaWiki (SMW) [23, 13] is an extension of the popular MediaWiki[3]. The online-encyclopedia Wikipedia is the most prominent example of a MediaWiki application. SMW aims to improve the structure and searchability of the Wikipedia content by deploying semantic technologies. Therefore, SMW follows Wikipedia's user interface to attract a broad user community. SMW extends the WikiML with attributes, types, and relationships. To represent classes, SMW utilizes existing Wikipedia categories. By assigning a wiki page to a given category it becomes an instance of the class being represented by this category. Relationships between instances are implemented via typed hyperlinks. In addition, SMW provides a set of units of measurement and customizable data types. Attributes, types, and relationships are represented in seperate wiki pages. Semantic search is implemented in SMW with a proprietary query language (WikiQL) that closely reflects the annotation syntax. WikiQL allows the creation of dynamic wiki pages of automatically aggregated content. SMW has become the most popular out of all Semantic Wikis and serves as the basis for numerous domain specific Semantic Wiki applications.

MaknaWiki [7] uses RDF triples (subject, predicate, object) for annotating wiki pages. RDF triples can be added to a text page by using a customized WikiML or within a separate form. RDF triples' subjects or objects refer to textual wiki pages that represent a concept each. MaknaWiki only supports maintenance and manipulation

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

of instances, but no class definitions. It extends JSPWiki[4] and its WikiML with typed links and literals. MaknaWiki's semantic annotation is utilised for navigation based on RDF triples and for limited semantic search (e.g., for searching instances of classes with distinct properties).

IkeWiki [19] tries to bring together application experts and knowledge engineers. Therefore, the user interface offers separate views for textual content and semantic annotation. The annotation editor supports the assignment of classes to wiki pages and typed links for representing relationships. Furthermore, classes, properties, and resources can be freely created and manipulated. The ontology metamodel closely reflects the underlying KRL (RDF(S) and OWL). The user interface for editing metadata supports automatic completion of terms.

SweetWiki [4] combines social tagging [11] and semantic technologies into what they call *semantic tagging*. Wiki pages are annotated with user tags that form not only a collective index (a so called *folksonomy* [22]), but a formal ontology. This is achieved by regarding each user tag as a concept of an ontology. Relationships between concepts are not determined by users, but by designated experts. The user does not interact with the ontology directly, but is merely able to create and to assign user tags. For the user, there is no distinction between instances and classes, because tags can represent both. Sweet Wiki's user interface provides a WYSIWYG editor for manipulating the wiki content.

Besides the above mentioned projects there exist several alternative Semantic Wiki implementations that can also be arranged within the framework given by our examples ranging from wikitologies to ontology-based wikis[5]. As a rule, early implementations such as PlatypusWiki strictly separate textual content from knowledge representations, while later projects (besides IkeWiki, which separates annotations from content) integrate knowledge representations and textual content by utilizing customized WikiML. Above all, IkeWiki and MaknaWiki provide dynamic authoring support. The ontology metamodels of PlatypusWiki, MaknaWiki and IkeWiki merely provide a direct mapping of the underlying KRL without any covering. Only MaknaWiki offers (limited) semantic search facilities. If elements of RDF(S) or OWL are utilized directly, RDF query languages such as SPARQL [18] can be applied. IkeWiki enables data export via SPARQL without providing a search interface. MaknaWiki and PlatypusWiki use elements of RDF(S) and OWL without making any use of their semantic expressiveness.

Contrariwise, SMW deploys a simplified ontology metamodel based on OWL-DL with an easy to use query language (compared to SPARQL) The SMW user interface keeps the connection between classes and their attributes covered, but offers no further editing assistance to the user (beside the provision of templates). Sweet Wiki's ontology metamodel does not distinguish between classes, instances, datatype properties, or object properties – everything is mapped on tags. Therefore, information can only be provided via tag search or via direct SPARQL queries with

[4] `http://jspwiki.org/`

[5] A list of current Semantic Wiki Systems is available at http://semanticweb.org/wiki/Semantic_Wiki_State_Of_The_Art

the consequence that information being distributed over several wiki pages can not be queried.

We propose the Semantic Wiki *Maariwa* that utilizes an ontology metamodel covering the underlying OWL-Lite language that enables information reuse as well as semantic queries over distributed information. In the following chapter we introduce Maariwa's underlying concepts and give a brief sketch on its implementation.

3 The Maariwa Concept – Architecture and Implementation

Maariwa is a Semantic Wiki project developed at the Friedrich Schiller University in Jena, Germany, with the objective to implement an augmented wiki system that enables simultaneous creation and manipulation of wiki content and ontologies. Maariwa's semantic annotation is utilized to put augmented navigation and semantic search into practice for reuse and aggregation of the wiki content. In Maariwa ontologies are used to structure the textual wiki content for providing access paths to the knowledge being represented in the wiki. Maariwa's access paths can be addressed via *MarQL*, a simple semantic query language, to enable semantic search. We therefore refer to Maariwa's underlying concept as *ontology-based web publishing*. Maariwa is geared towards user communities without expert knowledge in knowledge representations and knowledge engineering. Furthermore, Maariwa facilitates access by utilizing a WYSIWYG-editor for wiki content and ontology manipulation.

In this section, we introduce the Maariwa ontology metamodel and show how its elements are integrated into the wiki by illustrating important facets of the user interface. Finally, we comment on the implementation of ontology versioning and introduce MarQL syntax and semantics.

3.1 Maariwa's Ontology metamodel

The ontology metamodel of Maariwa is designed to enable simple and efficient searchability, as e.g., answering questions like 'What physicists were born in the 19th century' or 'Which cities in Germany have more than 100.000 inhabitants?'. To answer these questions, the ontology metamodel has to provide the semantic means to express these queries, while on the other hand it must be simple enough so that the arbitrary user without expert knowledge can comprehend and apply it. Tab. 1 shows a subset of OWL-Lite elements that are utilized for Maariwa's ontology metamodel.

We refer to datatype properties and object properties as attributes and relationships. Furthermore, both attributes and relationships are not defined as global entities but only local within their classes. This enables attributes and relationships with the same name in different classes without worrying the non-expert user with naming conflicts and disambiguation. Attributes are determined by a datatype with an optional measuring unit. For simplicity, only numbers, strings, and dates have been considered for datatypes.

Table 1 Mapping of OWL-Lite elements to Maariwa's ontology metamodel

OWL-Lite element	Maariwa element
class	class
datatype property	attribute
object property	relationship
class instance	wiki page representing a class being instantiated as an instance-page
subClassOf	superclass /subclass
individual	wiki page describing an individual, being an instance of one or more classes
datatype property instance	attribute value of an instance-page
object property instance	relationship value of an instance-page

By integrating the ontology metamodel into the wiki system, wiki pages can be annotated with concepts of ontologies to formalise their content. Classes, individuals, and sets of individuals can be described by wiki pages. A page that describes

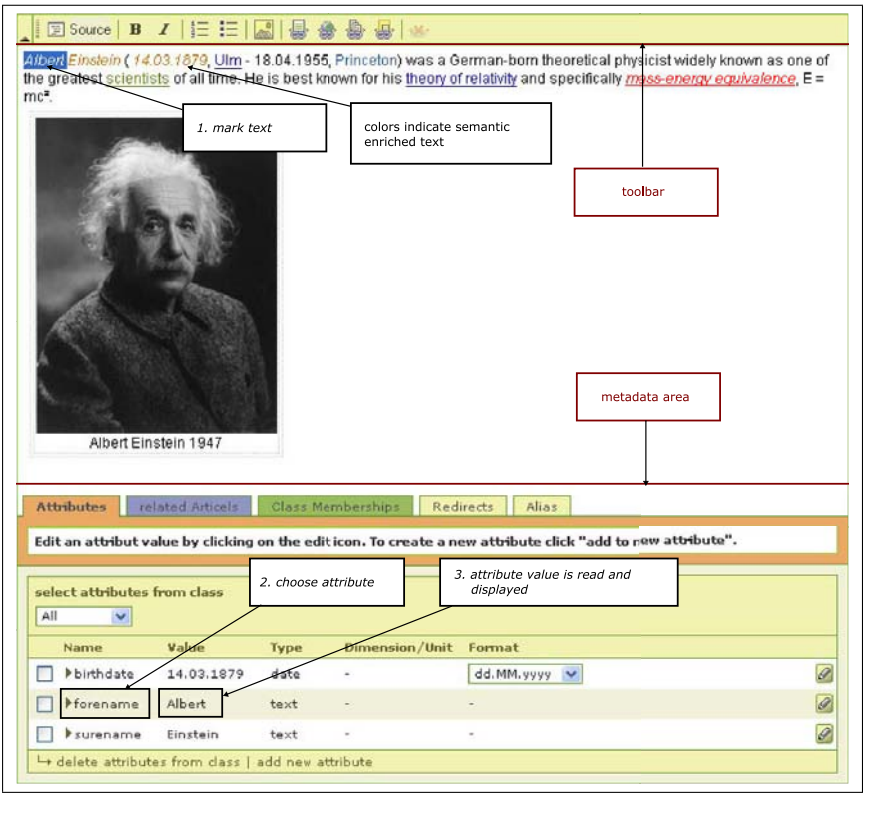

Fig. 1 Editing of a Maariwa wiki page with textual content (above) and ontologies (below)

a class is associated with attributes, relationships, and superclasses. Pages that describe individuals are associated with one class at least. To denote a set of individuals, a page has to be associated with a MarQL expression (see section 3.4). Typed links between pages may refer to relationships. Classes may use instance-pages as simple tags by associating neither a class hierarchy nor relationships, nor attributes.

3.2 Integration of Semantic Annotation and Textual Content

The graphical Maariwa editor is designed to minimize the user's additional effort for adding and maintaining semantic annotations. Repeated input of text or ontology concepts is avoided most times simply by copying existing concepts from the page context. New concepts are created in dialogue with the user. Thus, Maariwa enables the development of textual and ontological resources in parallel.

Maariwa provides two alternatives for creating semantic annotations: concepts of an ontology can be created from textual content of a wiki page, while on the other hand concepts might be defined first providing a textual description in the wiki page afterwards. Schema and instance data can be manipulated in parallel without interrupting the editing process of the wiki page (see Fig. 1). Maariwa's WYSIWIG-editor is implemented as so called *Rich Internet Application* [15] that provides desktop functionality for web applications and adopts the role of the traditional WikiML-based user interface.

In Maariwa, semantic annotations are directly displayed within the wiki page (see. Fig. 2). Different colors are used to denote the semantics of typed hyperlinks. Text that contains attribute values as well as links that represent relationships is highlighted. In addition, tooltips (i.e., pop-up information windows) display the semantic of an annotated text fragment if it is touched with the mouse pointer. Navigation within class hierarchies and class relations is enabled with a superimposed class browser. The semantic annotation of each wiki page can be separately accessed and exported in RDF/XML encoding via an own URL. Also export and import of ontologies as a whole is supported.

3.3 The MarQL Semantic Query Language

The syntax of SPARQL reflects the characteristics of the RDF data model while adopting the pattern of the simple database query language SQL. RDF statements are represented as triples and the RDF document can be interpreted as a graph. SPARQL traverses the RDF graph and as result delivers the nodes that satisfy the constraint given in the SPARQL query. Because RDFS and OWL are based on the RDF syntax SPARQL can also be used to query RDFS- and OWL-files, but without exploiting their semantic expressiveness.

MarQL is a customized semantic query language for the Maariwa ontology metamodel. MarQL syntax does not refer to RDF triples but directly addresses ontology elements such as classes, attributes and relationships. The underlying RDF encoding of the data remains hidden. In comparison to SPARQL, the syntax of MarQL is

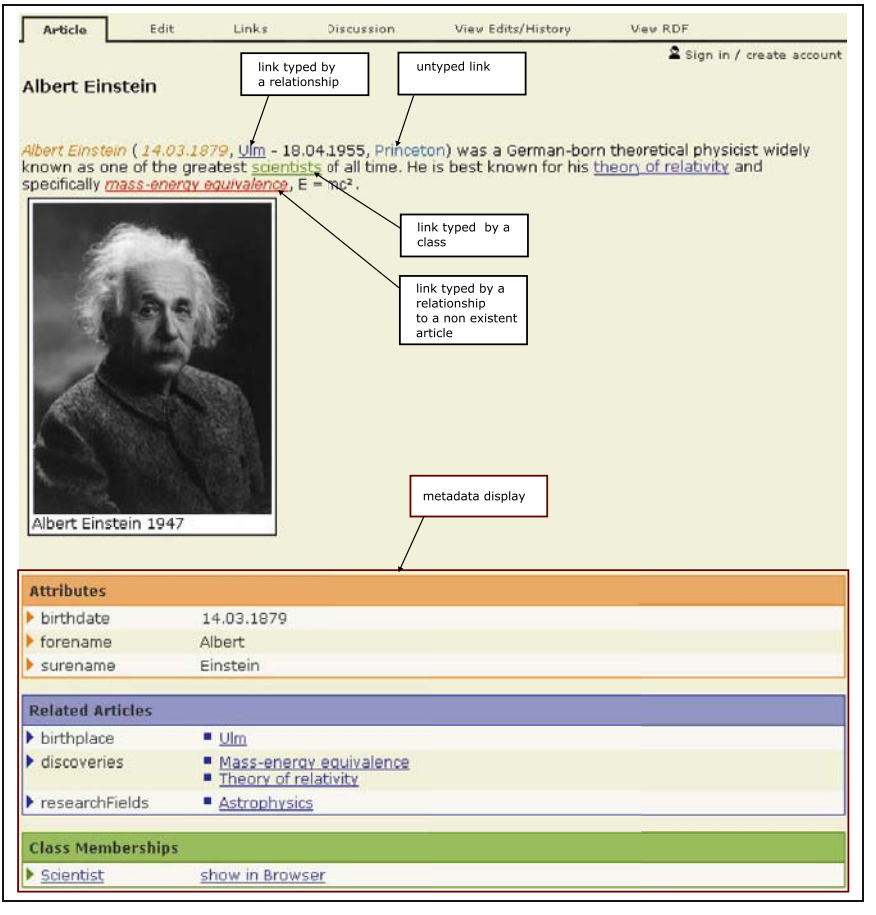

Fig. 2 A wiki page with semantic annotation in Maariwa

much more compact but less flexible. MarQL only implements a fixed set of query patterns. A MarQL query results in a set of wiki pages that refer to individuals, which satisfy the constraints of the MarQL query.

The structure of a MarQL query can be shown with an example: the expression *Scientist.institution.location.country = Germany* refers to wiki pages about scientists that work at an institute being located in Germany. *Scientist* refers to a class with a relationship *institution*. Relationships can be applied recursively and are denoted as a path expression. In this way, *institution* and *location* are connected. This means that there must exist a class, which is the target class of a relationship with *institution*, while in addition having a relationship with *location* . In the same way *location* and *country* are connected, while *country* can either be an attribute or a relationship and therefore *Germany* might denote an individual or an attribute value.

MarQL provides logical operators as well as string operators and operators for comparison. E.g., the query *City.population* $\geq$ *100.000* results in a set of all wiki

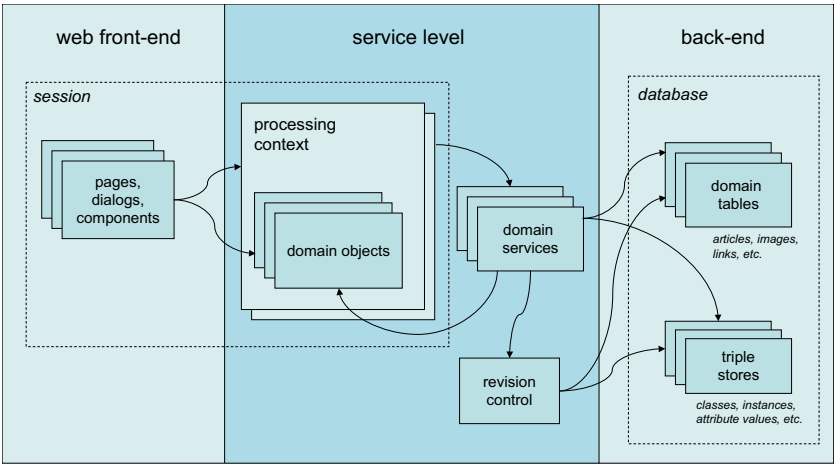

Fig. 3 Architecture of the Maariwa system

pages that describe cities with more than 100.000 inhabitants, or *Scientist.birthdate* $<$ *1.1.1900 AND Scientist.birthdate* $\geq$ *1.1.1800* results in a set of wiki pages with scientists that are born in the 19th century. The latter example results in a list of instances of a class and can also be referred to as a simple *tag*.

3.4 Maariwa Architecture and Implementation

Maariwa does not extend one of the existing traditional wiki systems but is a proprietary development based on Java. The application core of Maariwa implements a service level that realizes a wiki system as a set of loosely coupled services (see Fig. 3). Beside data management and revision control of the wiki pages and the related ontologies, keyword search and semantic search are also implemented as services. MarQL queries are translated into SPARQL queries. Manipulation of the ontologies in the editor is organized in different dialog levels. Within a dialog updates of one or more objects can be performed. These updates can either be revoked or they will also be adopted on the subjacent levels. Therefore, the service level offers cascading manipulation levels that implement this functionality with the help of local copies and snapshots.

Back-end data processing is achieved with a relational database management system. The service level stores ontology elements in various RDF triple stores and all other objects in separate database tables. The service level is used by the components of the web front-end that constitute Maariwa's user interface. Web server and browser client communicate asynchronously via AJAX [10] to achieve better usability. The WYSIWYG-editor represents the text of the wiki page as XHTML [8] fragment and is based on Javascript.

As a rule, wiki systems deploy a *revision control system* (RCS) to prevent abuse of unprotected write access. Maariwa adapts this RCS for ontologies, too. The RCS

maintains two levels: schema level and instance level. The schema level comprises all changes on classes, relationships and attributes, while the instance level covers changes of individuals including their attribute values and relationship values. Both levels are tightly coupled, because each schema update might have effects on all derived individuals. Therefore, a schema version additionally includes all updates on the instance level that occurred since the last update of the schema. Each version of a wiki page besides the update of the page content also contains updates of the associated concepts.

4 Conclusion

In this paper, we have introduced the Semantic Wiki approach Maariwa based on an ontology metamodel customized especially for the deployment within a wiki. For optimized usability recent client-side technologies have been combined with a simple semantic query language. The user can annotate text fragments of a wiki page in an interactive and rather intuitive way to minimize the additional effort that is necessary for adding semantic annotation. Thus, the productivity and efficiency of a Semantic Wiki system will open up for non-expert users as well. The ontology metamodel enables the formulation of access paths to wiki pages as well as the reuse of already implemented relationships in the underlying knowledge representation. The simple query language MarQL uses Maariwa's annotations and the contained access paths for implementing a semantic search facility. Ontology metamodel and query language are synchronized to support annotations on different levels of expressiveness. Thus, enabling simple tags as well as complex ontologies with attributes and relationships.

Currently, Maariwa is extended to include meta data also directly within the textual wiki content, as e.g., tables with self-adjusting data aggregations. Also natural language processing technology is considered to be utilized in the WYSIWYG-editor for automated suggestions as well as for translating natural language queries into MarQL. For deploying a Semantic Wiki, the user considers always the ratio of cost and effect that is caused by the additional effort of providing semantic annotation. In doing so, it is not important whether the creation of semantically annotated textual content or the creation of mere ontologies is focussed, but how both can be integrated within an system that provides a reasonable and efficient interface for user access.

References

1. Arpírez, J.C., Corcho, O., Fernández-López, M., Gómez-Pérez, A.: WebODE: a scalable workbench for ontological engineering. In: 1st Int. Conf. on Knowledge Capture (KCAP 2001), Victoria, Canada (2001)
2. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American 284(5), 34–43 (2001)
3. Brickley, D., Guha, R.V.: RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation, RDF Core Working Group, W3C (2004)

4. Buffa, M., Gandon, F.: Sweetwiki: semantic web enabled technologies in wiki. In: Proc. of the 2006 international symposium on Wikis, pp. 69–78 (2006)
5. Campanini, S.E., Castagna, P., Tazzoli, R.: Platypus wiki: a semantic wiki wiki web. In: Semantic Web Applications and Perspectives, Proc. of 1st Italian Semantic Web Workshop (2004)
6. Decker, B., Ras, E., Rech, J., Klein, B., Höcht, C.: Self-organized reuse of software engineering knowledge supported by semantic wikis. In: Workshop on Semantic Web Enabled Software Engineering (SWESE), at ISWC 2005, Galway, Ireland (2005)
7. Dello, K., Paslaru, E., Simperl, B., Tolksdorf, R.: Creating and using semantic web information with makna. In: Proc. of the 1st Workshop on Semantic Wikis – From Wiki To Semantics ESWC 2006 (2006)
8. Pemberton, S., et al: XHTML 1.0 The Extensible HyperText Markup Language. W3C Recommendation, W3C, 2nd edn. (August 2002)
9. Farquhar, A., Fikes, R., Rice, J.: The Ontolingua server: A tool for collaborative ontology construction. Int. Journal of Human-Computer Studies 46(6), 707–727 (1997)
10. Garrett, J.J.: Ajax: A new approach to web applications (2005), http://www.adaptivepath.com/publications/essays/archives/000385.php
11. Golder, S., Huberman, B.A.: The structure of collaborative tagging systems. Journal of Information Sciences 32(2), 198–208 (2006)
12. Klyne, G., Carroll, J.J.: Resource Description Framework (RDF): Concepts and abstract syntax. W3C Recommendation, W3C (February 2004)
13. Krötzsch, M., Vrandecic, D., Völkel, M.: Wikipedia and the semantic web - the missing links. In: Proc. of the 1st Int. Wikimedia Conf., Wikimania (2005)
14. Leuf, B., Cunningham, W.: The Wiki Way: Quick Collaboration on the Web. Addison-Wesley, Reading (2001)
15. Loosley, C.: Rich Internet Applications: Design, Measurement, and Management Challenges. Technical report, Keynote Systems (2006)
16. McGuinness, D.L., van Harmelen, F.: Owl Web Ontology Language: Overview. W3C Recommendation, World Wide Web Consortium (February 2004)
17. Oren, E., Delbru, R., Möller, K., Völkel, M., Handschuh, S.: Annotation and Navigation in Semantic Wikis. In: Proc. of the 1st Workshop on Semantic Wikis, Workshop on Semantic Wikis. ESWC 2006 (June 2006)
18. Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF (W3C Recommendation). W3C Recommendation, W3C (January 2008)
19. Schaffert, S., Gruber, A., Westenthaler, R.: A semantic wiki for collaborative knowledge formation. In: Semantics 2005, Vienna, Austria (2005)
20. Souzis, A.: Rhizome position paper. In: Proc. of the 1st Workshop on Friend of a Friend, Social Networking and the Semantic Web (2004)
21. Swartout, B., Ramesh, P., Russ, T., Knight, K.: Toward distributed use of large-scale ontologies. In: Symp. on Ontological Engineering of AAAI, Stanford, California (March 1997)
22. Vander Wal, T.: Folksonomy Explanations (2005), http://www.vanderwal.net/random/entrysel.php?blog=1622
23. Völkel, M., Krötzsch, M., Vrandecic, D., Haller, H., Studer, R.: Semantic wikipedia. In: Proc. of WWW 2006, Edinburgh, Scottland (2006)