IkeWiki: A User-Friendly Semantic Wiki

Sebastian Schaffert, Rupert Westenthaler, and Andreas Gruber
Salzburg Research Forschungsgesellschaft
Jakob Haringer Str. 5/II, A-5020 Salzburg, Austria
http://www.salzburgresearch.at

ABSTRACT

This article describes the architecture and design principles of *IkeWiki*, a Semantic Wiki we developed at Salzburg Research. Outstanding features of IkeWiki are its support for collaborative knowledge engineering, its ease of use, its support for different levels of formalisation ranging from informal texts to formal ontologies, and its sophisticated, interactive user interface.

1. INTRODUCTION

A serious obstacle for the development of Semantic Web applications is the lack of formal ontologies and knowledge represented in other formal languages. Arguably, one of the main reasons for this is the rather high technical barrier for using Semantic Web technologies that deters many domain experts from formalising "their" knowledge.

At the same time, wiki systems are becoming more and more popular as tools for content and knowledge management. Much knowledge is nowadays available in systems like Wikipedia. Unfortunately, this vast knowledge is not accessible for machines. If a small amount of this knowledge would be formalised, wiki systems could provide improved interfaces and advanced searching and navigation facilities.

"Semantic Wikis" aim to contribute to this by combining Wiki and Semantic Web technology. Semantic Wikis exist in many different flavours, of which we only mention a few for space reasons. In this article, we present our feature-rich prototype system *IkeWiki*[1] (*ike* = *knowledge*, *wiki* = *fast*).

Semantic Wikis make the inherent structure of a wiki – given by the strong linking between pages – accessible to machines (agents, services) beyond mere navigation. Such annotations are useful for many purposes, e.g. enhanced presentation by displaying contextual information, enhanced navigation by giving easy access to relevant related information, and enhanced "semantic" search that respects the context in addition to the content.

[1]available at http://ikewiki.salzburgresearch.at

Demos and Posters of the *3rd European Semantic Web Conference (ESWC 2006)*, Budva, Montenegro, 11th - 14th June, 2006

Typing/Annotating of Links. Semantic Wikis allow to annotate links by giving them certain types. The rationale is that a link created by a user almost always carries meaning beyond mere navigation. Some semantic wikis include the annotations as part of the wiki syntax (e.g. *SeMediaWiki* [1]), while others provide a separate editor for adding annotations (e.g. *IkeWiki*).

Context-Aware Presentation. Many semantic wikis can change the way content is presented based on semantic annotations. This can include enriching pages by displaying of semantically related pages in a separate link box, displaying of information that can be derived from the underlying knowledge base (e.g. license information), or even rendering the content of a page in a different manner that is more suitable for the context (e.g. multimedia vs. text content).

Enhanced Navigation. In Semantic Wikis, link annotations describe the relation of two pages. Such information can be used to offer additional or more sophisticated navigation, e.g. in a separate "related information" box.

Semantic Search. Most semantic wikis allow a "semantic search" on the underlying knowledge base. Usually, queries are expressed in SPARQL, and allow users to ask queries like "retrieve all pieces composed by Mozart" or "retrieve all documents where the license permits derivative works".

2. DESIGN PRINCIPLES

IkeWiki has originally been developed as a tool to support knowledge workers in collaboratively formalising knowledge [2]. Its design principles are influenced by this idea:

Easy to Use, Interactive Interface. As domain experts are usually non-technical people, IkeWiki aims to be easy to use. Its interface resembles as closely as possible the familiar Wikipedia interface. Also, IkeWiki offers an interactive WYSIWYG editor (using AJAX technology to communicate with the server backend) in addition to the traditional structured text editor. The WYSIWYG editor also supports interactive typing of links and resources.

Compatibility with Wikipedia. A significant amount of "informal" knowledge is available in Wikipedia. To reuse this knowledge, IkeWiki supports the Wikipedia syntax. This allows users to import existing content from Wikipedia into IkeWiki (e.g. via simple copy and paste) and begin with semantic annotations straight away.

Compatibility with Semantic Web standards. To be able to exchange data with other applications (e.g. ontology editors, Semantic Web services, other wikis), IkeWiki is purely based on existing Semantic Web standards.

Immediate Exploitation of Annotations. An impor-

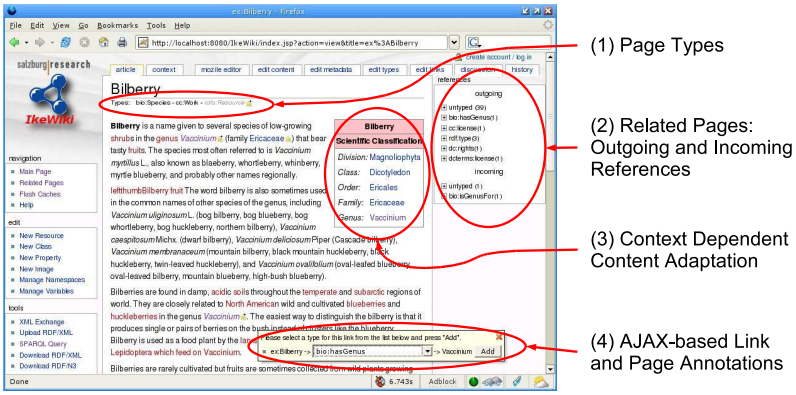

Figure 1: IkeWiki interface (cf. Section 4)

tant motivating aspect of wiki systems is that content is immediately publicly available. Similarly, IkeWiki allows immediate exploitation of semantic annotations for enhanced editing, presentation, navigation, and searching, even if the knowledge base is not yet fully formalised.

Support for Different Levels of Experience. IkeWiki is designed as a tool for collaborative knowledge engineering, where it is common that non-technical domain experts work together with experienced knowledge engineers. Therefore, IkeWiki supports all levels of experience. This means that certain advanced functionalities can be hidden from novice users but are available to experienced users.

Support for Different Levels of Formalisation. Different application areas need different levels of formalisation [2]. One of the goals of IkeWiki is thus to support formalisation of knowledge all the way from informal texts to formal ontologies. Also, this means that parts of the knowledge base might be more formalised than others, and that formal knowledge is in constant evolution.

Support for Reasoning. Unlike most other Semantic Wikis, IkeWiki supports reasoning. This allows to derive additional, implicit knowledge from the knowledge base using pre-defined or user-defined rules.

3. ARCHITECTURE

IkeWiki is implemented as a Java web application using a layered architecture. Data is stored in a Postgres database. When a resource is requested, the XML page content and related RDF data are retrieved and combined in the RenderingPipeline into an enriched XML representation. This XML representation is then either offered as interchange format for other Web services or transformed into HTML for presentation in the user's browser.

Page Store. The page store component serves to store and retrieve page content from the database, and supports versioning. Page content is represented in an XML format we call *WIF* (wiki interchange format). Basic WIF merely describes the page content and structure, but allows to add custom, application specific information.

RDF Store. The knowledge base is represented using the Jena RDF framework. Part of the RDF store is a SPARQL engine that allows for searching of the knowledge base.

Rendering Pipeline. The Rendering Pipeline combines page content with semantic annotations. Its output is a WIF document enriched by relevant semantic annotations. The "pipeline" consists of small "wiklets" that add specific pieces of information to the WIF document. Wiklets can be enabled and disabled and associated with permissions so that only selected users can see the added information.

4. INTERFACE

IkeWiki uses a purely browser-based interface (cf. Figure 1). It currently only supports the Mozilla browser family due to its standards compliance and free availability.

Page View. Figure 1 shows a sample article (copied from Wikipedia) about the "Bilberry" as rendered in IkeWiki. Type information is given below the page title (1). Links to (semantically) related pages are displayed in a separate "references box" on the right hand side (2). The taxonomy box (3) showing the biological classification of the described plant is automatically generated from existing semantic annotations (i.e. Bilberry *hasGenus* Vaccinium) and is an example for context adaptation. Finally, (4) shows interactive typing of links using AJAX technology.

Content Editor. The content editor is available in two flavours: as a traditional structured text editor and as a WYSIWYG editor. The structured text editor is aimed at expert users that are familiar with other wiki systems, and allows to directly copy content from Wikipedia. The WYSIWYG editor is aimed at novice users creating new content and interacts closely with the server backend: links are automatically recognised and verified, and links can be annotated directly in the editor (Fig. 1, item (4)).

Semantic Annotations Editor. Semantic annotations are separated into three editors: the *metadata editor* allows to fill in textual metadata related to a page (like Dublin Core metadata or RDF comments). The *type editor* allows to associate one or more of the types available in the system with a page. The *link editor* allows to annotate outgoing and incoming links.

5. REFERENCES

[1] M. Krötsch, D. Vrandečić, and M. Völkel. Wikipedia and the Semantic Web - The Missing Links. In *Proceedings of the WikiMania2005*, 2005.

[2] S. Schaffert, A. Gruber, and R. Westenthaler. A Semantic Wiki for Collaborative Knowledge Formation. In *Semantics 2005*, Vienna, Austria, November 2005.