PlWiki – A Generic Semantic Wiki Architecture*

Grzegorz J. Nalepa

Institute of Automatics,
AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Kraków, Poland
gjn@agh.edu.pl

Abstract. The paper presents a new semantic wiki architecture called *PlWiki*. The most important concept is to provide a strong knowledge representation and reasoning with Horn clauses-based representation. The idea is to use Prolog clauses on the lower level to represent facts and relations, as well as define rules on top of them. On the other hand a higher-level Semantic Web layer using RDF support is provided. This allows for compatibility with Semantic Media Wiki while offering improved representation and reasoning capabilities. Another important idea is provide an extension to already available flexible wiki solution (DokuWiki) instead of modifying existing wiki engine.

1 Introduction

Recently the most important development of the Internet concerned not the lower network network layers, but the higher application or service layers related to the Web technology. This is mainly due to the fact, that while the speed and storage capacities of the Web increased by orders of magnitude, its search and processing capabilities remained almost unchanged on the conceptual level.

This phenomena led, almost a decade ago, to the proposal of the *Semantic Web* [1]. In this architecture a number of higher level semantic facilities built on top of the Web would allow not just to *search data* but to *reason with knowledge*. In fact, this was the point where the focus of the Web development moved from *content* (data) to *knowledge* (in a broad sense). A decade later, number of semantic technologies is available and widely used, starting from the data structuring XML, to meta-data annotations with RDF and ontologies with RDFS and OWL (observe the well-known Semantic Web layer cake). While these technologies provided knowledge encoding and representation solutions, the challenge remains to provide an efficient knowledge processing and reasoning with rules on the Web. This is in fact the point, where most of the current Semantic Web research focuses. Recent rule standards from W3C include RIF and SWRL.

Besides knowledge representation and reasoning, a sensible knowledge engineering solution for the Web is another important challenge. While the Semantic Web initiative targets mainly representation aspects, it does not directly address

* The paper is carried out within the AGH UST Project No. 10.10.120.105.

N.T. Nguyen, R. Kowalczyk, and S.-M. Chen (Eds.): ICCCI 2009, LNAI 5796, pp. 345–356, 2009.

the specific problems stemming from the massively parallel and collaborative nature of the Web. Social networks, that provide specific services on top of the Web and the Semantic Web, try to cope with these problems. Recently the technology of the wiki systems has gained importance with respect to the collaborative knowledge acquisition and engineering. The development of *semantic wikis* [2,3,4] allowed to use the Semantic Web methods and tools on top of the existing content-centered wiki solutions.

Existing semantic wikis allow for introduction of semantic information (e.g. meta-data, ontologies) into a wiki. In fact, they often allow to build a wiki around an ontology, which improves their conceptual coherence. Most of the semantic wikis reached a stage where the reasoning capabilities have to be added. This is where some limitations of existing solutions become exposed.

In this paper a new semantic wiki architecture called *PlWiki* is considered. The most important concept is to provide a strong knowledge representation and reasoning with Horn clauses-based representation. But instead of building directly on top of OWL [5] and SWRL [6] a more generic solution is proposed. The idea is to use Prolog [7] clauses to represent facts and define rules on the lower level. On the other hand a higher level Semantic Web layer using RDF and OWL support is provided. Another important idea is not to develop an entirely new wiki engine (e.g. [2]) but to provide an extension to an existing well-established wiki solution. It is argued, that such a generic architecture is both more flexible and efficient. This approach is loosely based on some preliminary ideas earlier considered in [8].

The rest of the paper is organized as follows: In Sect. 2 wikis as web-based knowledge engineering systems are discussed. Then in Sect. 3 the development of semantic and knowledge wikis is considered. The Sect. 4 gives the main motivation of the research. The requirements for the PlWiki system are specified in Sect. 5. The design of the PlWiki plugin for the DokuWiki system is presented in Sect. 6. The use of the plugin is considered on the example introduced in Sect. 7, and the semantic layer is discussed in Sect. 8. Finally a short evaluation of the approach as well as directions for the future work are given in Sect. 9.

2 Wikis Technology

Wiki systems appeared in the mid 90s. According to Wikipedia the first system called "wiki" (WikiWikiWeb) was established 15 years ago. Tho goal of these systems was to provide a conceptually simple tool for massively collaborative knowledge sharing and social communication. Wikis were meant to help build certain communities interested in given topics. Clearly some of them grew large and general, such as the Wikipedia.

A wiki system is a community-driven collaboration tool. It allows users to build content in the form of the so-called wiki pages, as well as uploaded media files. Wikipages are plain text documents containing special wiki markup (e.g. for structuring content) thus creating the so-called wikitext. The wikitext is simplistic and human readable, making it a much more accessible tool than

HTML/XML. Pages are identified by a unique keyword (name) and usually grouped within the so-called namespaces. Pages are linked to each other and to external websites creating a hyperwikitext structure.

An important feature of wikis is the integrated version control functionality, very helpful in a collaborative environment. It allows registering all subsequent versions of every page, thus allowing to see introduced differences. All wiki edits may be identified by user names and time stamps, so it is possible to recreate any previous state of the wiki at any given time.

From the technical point of view a wiki has a regular web-based client-server architecture. It is run on the web server and accessed by a regular browser. Wikis introduce a range of access control mechanisms from simple ones, to full-fledged ACL (Access Control Lists) solutions. On the server side wikis require different runtime environment (e.g. PHP/JSP/Python), possibly with a relational database system. A comprehensive comparison of different wiki systems can be found on `http://www.wikimatrix.org`.

One of the most interesting wiki systems for developers is DokuWiki (`http://www.dokuwiki.org`). It is designed to be portable, easy to use and set up. Like number of other solutions DokuWiki is based on PHP. However, it does not require any relational database back-end. It allows for image embedding, and file upload and download. Pages can be arranged into namespaces which act as a tree-like hierarchy similar to directory structure. It provides syntax highlighting for in-page embedded code of programming languages such as: C/C++, Java, XML and others, using GeSHi (`qbnz.com/highlighter`). Furthermore, it supports extensive user authentication and authorization mechanisms including ACL. Its modularized architecture allows the user to extend DokuWiki with plugins which provide additional syntax and functionality. A large number of plugins is available. The templates mechanism provides an easy way to change the presentation layer of the wiki.

All wiki systems provide an abstract representation of the content they store. They all provide standard searching capabilities. However, they lack facilities helping in expressing the semantics of the stored content.[1] This is especially important in the case of collaborative systems, where number of users work together on the content. This is why wikis became one of the main applications and testing areas for the Semantic Web technologies.

3 A Challenge of Semantic Wikis

A step in the direction of enriching standard wikis with the semantic information has been performed by the introduction of the so-called *semantic wikis*, such as the IkeWiki [2], OntoWiki [9], SemanticMediaWiki [3], or SweetWiki [4]. In such systems the standard wikitext is extended with semantic annotations. These include relations (represented as RDF triples) and categories (here RDFS is needed). It is possible to query the semantic knowledge, thus providing dynamic

[1] Besides simple tagging mechanisms, that can later be used to create the so-called *folksonomies*.

wiki pages. Ultimately these extension can also allow for building an ontology of the domain to which the content of the wiki is related. This extension introduces not just new content engineering possibilities, but also semantic search and analysis of the content.

However, from the knowledge engineering point of view, expressing semantics is not enough. In fact a knowledge-based system should provide effective knowledge representation and processing methods. In order to extend semantic wikis to knowledge-based systems, ideas to use a problem-solving knowledge have been introduced. An example of such a system is the *KnowWE* semantic wiki [10,11]. In such a system the semantic knowledge is extended with the problem-solving domain-specific knowledge. The system allows for introducing knowledge expressed with decision rules and trees related to the domain ontology. Several semantic wiki systems are available, most of them in the development stage providing demo versions.[2] A recent FP7 project Kiwi (`http://www.kiwi-project.eu`) aims at providing a collaborative knowledge management based on semantic wikis (it is the continuation of IkeWiki effort).

In this paper a generic solution based on the use of Prolog as the language for expressing both the semantics, and the knowledge processing is presented.

4 Motivation and Objectives

The semantic wiki technology is a young one. Multiple systems are developed to test new ideas and features. However, number of conceptual challenges remain. Some of the persistent problems are:

- an expressive yet effective (in terms of inference) knowledge representation, allowing for explicitly representing the semantics of the wiki content,
- powerful query and inference facilities, that enable reasoning on top of the gathered knowledge; in fact they are the main factor limiting the practical usability of the knowledge,
- interfaces helping users to encode and use the knowledge they poses,
- integration with the existing technologies that improves the portability and makes the development of the system easier.

All of the existing semantic wiki solutions address these problems in various manners [4]. Some of the most common approaches to cope with these problems include the extensive use of selected Semantic Web technologies to introduce well-founded semantics. This includes the use of RDF for meta-data, as well as RDFS, and possibly OWL, for ontology management, and SPARQL as the query language [3]. Some other [12] introduce extended knowledge representation for problem solving. User interfaces are usually based on simple forms helping to input semantic annotations, as well as editors highlighting the wiki markup [4]. Most of these solutions modify some existing wiki engines, e.g. Media Wiki that powers Wikipedia. In general, they still lack universal rule representation (mostly

[2] See `http://semanticweb.org/wiki/Semantic_Wiki_State_Of_The_Art`

due to the development stage of SWRL). However, it is an ongoing research where an optimal solution is hard to find.

The approach discussed in this paper is different. The basic idea is to allow the use of a logical knowledge representation based on Horn clauses [13] for facts, relations and rules, as well as dynamic queries. This allows not only to represent facts, but also introduce rules for inference. On top of this the Semantic Web layer with RDF may also be provided. The approach is based on the concept of using the Prolog language interface. This also opens up possibilities of powerful querying mechanism, more powerful than SPARQL (while compatibility layer compatible with SPARQL is provided). The solution is developed as an optional extension to an existing modular wiki engine of DokuWiki.

The main objectives of this approach are to enhance both representation and inference features, allow for a complete rule framework in the wiki, and to use a clean integration approach with an existing system. In the next section more detailed system requirements are presented.

5 PlWiki Requirements

Starting from the motivation outlined in the previous section, the following main requirements concerning the new wiki solution have been formulated:

1. provide rich knowledge representation, including rules,
2. allow for an efficient and flexible reasoning in the wiki,
3. expressive power equivalent to Horn clauses,
4. provide a Semantic Web layer for the knowledge engineer,
5. extend an existing well-established wiki engine,
6. improve speed and portability for an easy deployment,
7. allow for meta-knowledge suitable for wiki knowledge evaluation.

A decision has been made to build the new wiki with use of the Prolog language [7]. This allows to meet first three requirements since Prolog programs assure both generic and flexible representation with Horn clauses, by providing fact and rule representation. Thus it is possible the represent the domain specific knowledge and reasoning procedures with the same generic representation.

The next requirement is a natural one, considering the existing Semantic Web stack, on which semantic wikis are build. This means the support for meta-data encoding with RDF and ontologies with RDFS (possibly with OWL too). Such a solution allows to extend the wiki using existing ontologies, optionally build with other semantic wikis.

The fifth requirement is a conscious decision based on number of experiences with other semantic wiki engines. There are tens of wiki engines available (see `www.wikimatrix.org`). Most of them are similar w.r.t to main concepts and features. However, there are number of differences when it comes to the wikitext syntax, implementation and runtime environment, as well as extra features. This is why, instead of building yet another wiki engine, or modify an existing one, another solution is proposed. The idea is to use a ready, flexible and extensible

wiki engine, that could be optionally extended with knowledge representation and processing capabilities.

The next requirement concerns an easy deployment of the new system. Number of existing solutions impose high requirements on the runtime environment for the wiki, e.g. a database server, a J2EE stack with extra libraries, etc. This makes the installation of such systems harder, limits their portability (i.e. number of hosting solutions do not meet all of these requirements), and lowers their speed (Java-based solutions are often slower than Python or PHP-based ones).

The last requirement concerns the ability to analyze wiki knowledge using procedures specified in the wiki, using the same representation as the wiki contents. This is considered for the future research [14].

In order to meet these requirement a fast, flexible and portable Prolog implementation has been chosen. It provides a rich library stack for the Semantic Web compatibility. In the next section the design of the PlWiki extension for the DokuWiki system built with use of the SWI-Prolog environment is given.

6 PlWiki System Design

The main goal of the new knowledge wiki design is to deliver a generic and flexible solution. Instead of modifying an existing wiki engine or implementing a new one, a development of an extension of the DokuWiki system was chosen. To provide a rich knowledge representation and reasoning for the Semantic Web, the SWI-Prolog environment was selected. The basic idea is to build a layered knowledge wiki architecture, where the expressive Prolog representation is used on the lowest knowledge level. This representation is embedded within the wiki text as an optional extension. On top of it number of layers are provided. These include standard meta-data descriptions with RDF and ontologies specification solutions with RDFS and OWL.

The PlWiki stack can be observed in Fig. 1. The stack is based on a simple runtime including the Unix environment with the Unix filesystem, the Apache web server and the PHP stack. Using this runtime the standard DokuWiki installation is run. The PlWiki functionality is implemented with the use of an optional plugin allowing to enrich the wikitext with Prolog clauses, as well run the SWI-Prolog interpreter. It is also possible to extend the wikitext with explicit semantical information encoded with the use of RDF and possibly OWL representation. This layer uses the Semantic Web library provided by SWI-Prolog. An optional decision rule layer is also considered with the use of the HeaRT runtime for the XTT^2 framework [15,16].

The main layer interfacing with the DokuWiki engine is presented in Fig. 2. The figure shows the dataflow in the DokuWiki system. DokuWiki provides a flexible plugin system, providing five kinds of plugins (see `www.dokuwiki.org/devel:plugins`):

- *Syntax Plugins*, extending the wikitext syntax,
- *Action Plugins*, redefining selected core wiki operations, (e.g. saving wikipages),

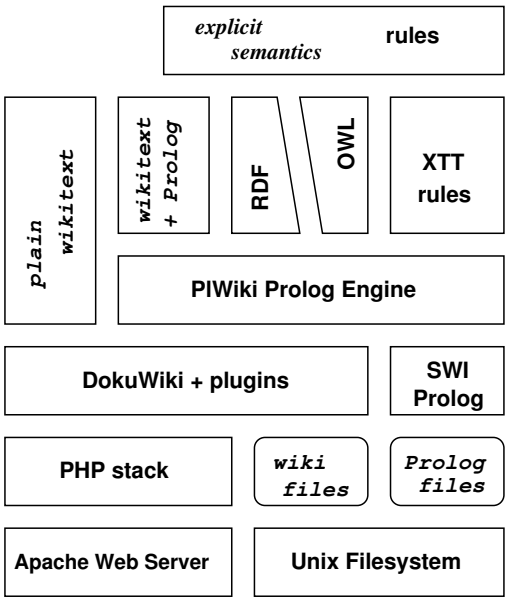

Fig. 1. The PlWiki stack

- *Admin Plugins*, providing extra administration functionality,
- *Helper Plugins*, supporting other plugins with generic functions,
- *Renderer Plugins*, allowing to create new export modes (possibly replacing the standard XHTML renderer).

The current version of PlWiki implements both the *Syntax* and *Renderer* functionality. Text-based wikipages are fed to a lexical analyzer (Lexer) which identifies the special wiki markup. The standard DokuWiki markup is extended by a special `<pl>...</pl>` markup that contains Prolog clauses. The stream of tokens is then passed to the Helper that transforms it to special renderer instructions that are parsed by the Parser. The final stage is the Renderer, responsible for creating a client-visible output (e.g. XHTML). In this stage the second part of the plugin is used for running the Prolog interpreter.

The detailed functionality of the PlWiki Syntax Plugin includes parsing the Prolog code embedded in the wikitext, and generating the knowledge base composed of files containing the Prolog code, where each wikipage has a corresponding file in the knowledge base. The PlWiki Renderer plugin is responsible for executing the Prolog interpreter with a given goal, and rendering the results via the standard DokuWiki mechanism.

The PlWiki framework uses the SWI-Prolog environment, licensed under the Lesser GNU Public License (see `www.swi-prolog.org`). It is a mature implementation widely used in research and education as well as for commercial applications. It provides a fast and scalable development environment, including

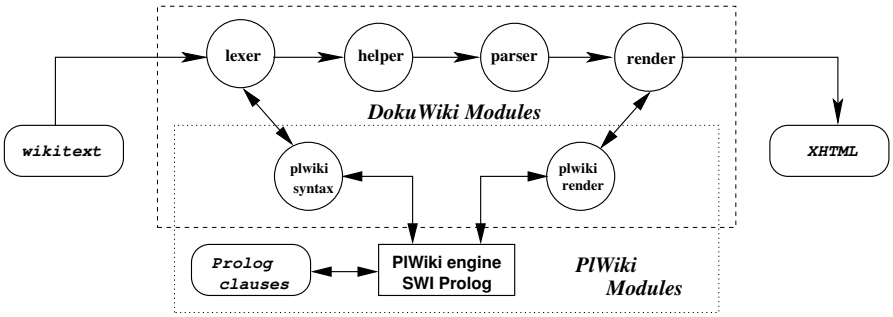

Fig. 2. PlWiki plugins

graphics, libraries and interface packages, portable to many platforms, including Unix/Linux platforms, Windows, and MacOS X. SWI-Prolog provides a rich set of libraries, including the *semweb* library for dealing with standards from the W3C standard for the Semantic Web (RDF, RDFS and OWL). This infrastructure is modular, consisting of Prolog packages for reading, querying and storing Semantic Web documents.

One should keep in mind, that the Prolog-based representation is quite close to the natural language. Not only on the semantical level, but to a degree also on the syntactic level. It is possible thanks to the operator definition.

Currently the PlWiki system is under heavy development. The system is being developed by Michał Kotra as part of his MSc Thesis. See `http://home.agh.edu.pl/gjn` for more information.

7 Prolog Representation Example

Below basic use examples of the generic Prolog representation are given.

```
<pl>
capital(poland,warsaw).
country(poland).
country(germany).
</pl>
```

This simple statement adds two facts to the knowledge base. The plugin invocation is performed using the predefined syntax. To actually specify the goal (query) for the interpreter the following syntax is used:

```
<pl goal="coutry(X),write(X),nl,fail"></pl>
```

It is possible to combine these two, as follows:

```
<pl goal="country(X),write(X),nl,fail">
country(france).
```

```
country(spain).
</pl>
```

It is possible to specify a given *scope* of the query (in terms of wiki namespaces):

```
<pl goal="country(X),write(X),nl,fail"
    scope="prolog:examples"></pl>
```

A bidirectional interface, allowing to query the wiki contents from the Prolog code is also available, e.g.:

```
<pl goal="consult('lib/plugins/prolog/plwiki.pl'),
          wikiconsult('plwiki/pluginapi'),list."></pl>
```

There are several options how to analyze the wiki knowledge base (that is Prolog files built and extracted from wiki pages). A basic approach is to combine all clauses. More advanced uses allow to select pages (e.g. given namespace) that are to be analyzed.

On top of the basic Prolog syntax, semantic enhancements are possible. These can be easily mapped to Prolog clauses.

8 The PlWiki Semantic Layer

Besides the generic Prolog-based knowledge representation features based on pure Prolog clauses, typical semantic wiki features are supported. Semantic Media Wiki (SMW) [3], a standard semantic wiki solution, provides a simple yet flexible mechanism for annotating categories, and properties. In the first version of PlWiki three main features are considered:

- categories definitions as in SMW,
- simple queries from SMW (with SPARQL queries in the future), and
- generic RDF annotations.

To provide a better compatibility with existing solutions parsing of SMW wikitext is provided, with a corresponding Prolog representation available. The wiki user can use the SMW syntax directly in DokuWiki to enter wikitext. The PlWiki plugins transforms the wikitext to Prolog clauses, asserted to the internal knowledge base. In fact these clasues could also be introduced by using the PlWiki `<pl></pl>` tags.

Examples are as follows, with the SMW syntax given first, and the corresponding Prolog representation below.

```
[[Category:Cities]] Warsaw is in Poland.
    wiki_category('Cities,'Warsaw').
Warsaw is [[capital of::Poland]].
    wiki_property(capital_of,subject_page_name,'Poland').
[[attribute::created:=April 22 2009]]
    wiki_attribut(page_uri,created,date(22,april,2009)).
```

The Prolog clauses are asserted to the PlWiki knowledge base by the syntax plugin analyzing the wiki text.

In a similar fashion simple queries are handled. A query for a category or property is simply mapped to a corresponding Prolog query:

```
{{#ask: [[Category:Cities]] [[capital of::Poland}}

wiki_category('Cities',Page),
wiki_property(capital_of,Page,'Poland'),
wiki_out(Page).
```

Plain RDF annotations are also supported. Currently, these are separated from the explicit annotations mentioned above. For compatibility reasons an RDF annotation can be embedded directly in XML serialization, then it is parsed by the corresponding Prolog library, and turned to the internal representation, that can also be used directly. SWI-Prolog's represents RDF triples simply as:

```
rdf(?Subject, ?Predicate, ?Object).
```

So mapping the above example would result in:

```
rdf('Warsaw',capital_of,'Poland').
```

The SWI-Prolog RDF storage in highly optimized. It can be integrated with the provided RDFS and OWL layers, as well as with the *ClioPatria* platform[3] providing also SPARQL queries.

Thanks to the full Prolog engine available in the wiki, the inference options are almost unlimited. Prolog uses backwards chaining with program clauses. However, it is very easy to implement meta-interpreters for forward chaining.

A simple clause finding recently created pages might be as follows:

```
recent_pages(Today,Page) :-
    wiki_attribut(Page,created,date(D,April,2009)),
    I is Today - D,
    I < 7.
```

Compound queries can also be created easily and executed as Prolog predicates.

9 Conclusions and Future Development

This paper presents an original idea of implementing a semantic wiki using Prolog for knowledge representation and inference. In the paper a proof-of-concept prototype implementation for the DokuWiki system is described. The system allows for fact, relations and rule representation, using Prolog. It also allows to use Semantic Web languages such as RDF to provide semantic annotations, opening possibilities to implement full support for OWL-based ontologies.

[3] See http://e-culture.multimedian.nl/software/ClioPatria.shtml

In the current version of the prototype it is not possible to check the syntax of the Prolog code, or assist the user in entering it. In the future debugging and syntax highlighting features are planned.

There are obviously some performance issues regarding knowledge processing within a wiki system based on Prolog code interpretation. Extracting knowledge from many pages and processing it could be a time consuming operation. Some smart caching techniques are evaluated. They are based on the caching mechanism present in the DokuWiki system.

An enhanced direct support for wiki markup present in other solutions (e.g. SemanticMediaWiki) is also planned. This should ultimately allow to import ready to use semantic wikis implemented with SMW, and possibly other wikis.

The user interface is an important area for improvement. Besides Prolog editing support, extended interfaces that use semantic forms are also considered. They should also allow ontology edition and visualization.

An important challenge is the rule framework in the wiki. Thanks to Prolog flexibility, it is possible to support number of rule languages built on top the SWI stack, including Semantic Web languages such as SWRL (with Description Logics reasoners integrated), as well as custom solutions. One of the options that is evaluated is the use of XTT^2 rule framework [16] that has a transparent Prolog implementation.[4]. On the other hand it provides standalone visual knowledge editors, and formal analysis tools (e.g. for rule verification).

Finally, an important research direction is a knowledge evaluation in the wiki. The distributed knowledge development process in a wiki poses new problems in knowledge engineering when compared to the classic development of monolithic knowledge bases. In case of most of the semantic wikis the focus of the current research is on the knowledge representation, integration and authoring. However, with the growing amount of knowledge contained in semantic wikis, the knowledge quality issues seem to be critical [14]. Using the Prolog interpreter several knowledge evaluation plugins are developed. Thus it is possible to analyze the knowledge stored in the wiki w.r.t. to a number of features [14]. This is an experimental feature that is being developed.

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web: Scientific American. Scientific American (May 2001)
2. Schaffert, S.: Ikewiki: A semantic wiki for collaborative knowledge management. In: WETICE 2006: Proceedings of the 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, pp. 388–396. IEEE Computer Society, Washington (2006)
3. Krötzsch, M., Vrandecic, D., Völkel, M., Haller, H., Studer, R.: Semantic wikipedia. Web Semantics 5, 251–261 (2007)
4. Buffa, M., Gandon, F., Ereteo, G., Sander, P., Faron, C.: Sweetwiki: A semantic wiki. Web Semantics: Science, Services and Agents on the World Wide Web (in press) (2008)

[4] See HeKatE project website at `http://hekate.ia.agh.edu.pl`

5. McGuinness, D.L., van Harmelen, F.: Owl web ontology language overview, w3c recommendation 10 february 2004. Technical report, W3C (2004)
6. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., Dean, M.: Swrl: A semantic web rule language combining owl and ruleml, w3c member submission May 21, 2004. Technical report, W3C (2004)
7. Bratko, I.: Prolog Programming for Artificial Intelligence, 3rd edn. Addison Wesley, Reading (2000)
8. Nalepa, G.J., Wojnicki, I.: Proposal for a prolog-based knowledge wiki. In: Nalepa, G.J., Baumeister, J. (eds.) 4th Workshop on Knowledge Engineering and Software Engineering (KESE2008) at the 32st German conference on Artificial Intelligence, Kaiserslautern, Germany, September 23, pp. 43–54 (2008)
9. Auer, S., Dietzold, S., Riechert, T.: Ontowiki - a tool for social, semantic collaboration. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 736–749. Springer, Heidelberg (2006)
10. Baumeister, J., Reutelshoefer, J., Puppe, F.: Knowwe: community-based knowledge capture with knowledge wikis. In: K-CAP 2007: Proceedings of the 4th international conference on Knowledge capture, pp. 189–190. ACM, New York (2007)
11. Baumeister, J., Puppe, F.: Web-based knowledge engineering using knowledge wikis. In: Proc. of the AAAI 2008 Spring Symposium on Symbiotic Relationships between Semantic Web and Knowledge Engineering, pp. 1–13. Stanford University, USA (2008)
12. Reutelshoefer, J., Baumeister, J., Puppe, F.: Ad-hoc knowledge engineering with semantic knowledge wikis. In: Proc. of SemWiki 2008 - The Wiki Way of Semantics, Workshop co-located with the 5th European Semantic Web Conference, Tenerife, Spain (2008)
13. Ben-Ari, M.: Mathematical Logic for Computer Science. Springer, London (2001)
14. Baumeister, J., Nalepa, G.J.: Verification of distributed knowledge in semantic knowledge wikis. In: FLAIRS-22: Proceedings of the twenty-second international Florida Artificial Intelligence Research Society conference, Sanibel Island, Florida, USA, May 19–21 (2009) (to be published)
15. Nalepa, G.J., Ligęza, A.: A graphical tabular model for rule-based logic programming and verification. Systems Science 31(2), 89–95 (2005)
16. Nalepa, G.J., Ligęza, A.: Xtt+ rule design using the alsv(fd). In: Giurca, A., Analyti, A., Wagner, G. (eds.) ECAI 2008: 18th European Conference on Artificial Intelligence: 2nd East European Workshop on Rule-based applications, RuleApps 2008: Patras, July 22, pp. 11–15. University of Patras, Patras (2008)