Engineering on the Knowledge Formalization Continuum

Joachim Baumeister, Jochen Reutelshoefer, and Frank Puppe

University of Würzburg, Am Hubland
97074 Würzburg, Germany

Abstract. Usually, domain knowledge is available at different levels of formality, for example such as documents, data bases, and (business) rules. We argue, that today's systems limit the knowledge engineering process to a fixed level of formality and expressiveness, respectively, and that these limitations hinder effective knowledge acquisition and use. In consequence, we introduce the *knowledge formalization continuum* as a metaphor, that embraces the fact that knowledge is available in different formalities. We motivate that a semantic wiki is a suitable tool to work on the knowledge formalization continuum, and we introduce KnowWE as an example wiki implementation.

1 Introduction

In today's enterprises we see that knowledge management systems and knowledge-based solutions are already implemented with reasonable success. There still exists a great deal of interest to build knowledge-based solutions. Typically, "knowledge" is already available in different representations ranging from technical documents, construction plans, sheets and experiences of human experts. However, the knowledge acquisition bottleneck, i.e., the problem of formalizing existing knowledge into a machine processable model, is still present and often prevents successful developments. Experiences in many projects over the last years showed that the implementation often faces differently favorable but conflicting options, thus creating the following dilemmas:

1. *The Single Expert Dilemma.* The motivation and sophistication of *single domain specialists* are often the driving forces of successful knowledge acquisition and evolution. Although, high-quality experts can guarantee the construction of a high-quality knowledge base, these persons are often short in time and motivation. A distribution of the workload would decrease this problem, but will open the risk of also reducing the overall quality of the formalized knowledge. Furthermore, the collaboration among a group of specialists is not supported sufficiently in many industrial systems concerning the distributed development of a knowledge base. Here, the dilemma exists of favoring a distributed over a monolithic development process.
2. *The Flexibility Dilemma.* Current state-of-the-art tools are often constrained to a specific knowledge representation and acquisition interface for developing the knowledge base. In consequence, the tools are commonly not flexible enough to map the mental model of the domain specialists that are responsible to formalize

the knowledge in the given application project. Additionally, "knowledge" can appear in diverse forms, such as textual and tabular data but also explicit rules. On the one hand, the mapping of the particular mental model of the specialists to the provided knowledge representation and interfaces, respectively, often turned out to be difficult, complex and time-consuming. On the other hand, a tool having the maximal flexibility regarding the user interfaces and provided knowledge representations, typically would increase the complexity of its use and therefore decreases the effectivity of the developers [1, p. 86]. In consequence, we face the dilemma of demanding for a tool with maximal flexibility vs. a tool with maximal productivity.

This paper introduces approaches to weaken the two dilemmas by first introducing the metaphor of a *knowledge formalization continuum* in order to give domain specialists a flexible mental model of the knowledge that is planned to be formalized. It frees the developers to commit to a particular knowledge formalization at an early stage, but offers a versatile understanding of the formalization process. Second, we propose the use of a *semantic wiki* as a suitable development tool for knowledge bases, that is able to scale from one single domain specialist to the collaboration of multiple domain specialists without changing the existing working process.

2 The Knowledge Formalization Continuum

A *continuum* can be seen as "a nonspatial whole in which no part or portion is distinct or distinguishable from adjacent parts"; alternatively a continuum can be understood as "anything that goes through a gradual transition from one condition, to a different condition, without any abrupt changes"[1].

We use these definitions of a continuum to explain the idea of a *knowledge formalization continuum*, where gradual transitions on formalization degrees of the same knowledge are possible, but where the knowledge to be modelled experiences no abrupt changes or "discontinuities".

The idea is quite simple: When starting with the development of a knowledge base, the considered knowledge is already available in varying forms, for example documents, recorded application cases or often in the heads of the experts. The usual task is to find an appropriate knowledge model which serves as the target of knowledge formalization. Here, often the issue arises, when some parts of the knowledge cannot be formalized into the selected model, or its formalization would be too costly with respect to the *cost/benefit principle* [1, p. 56]. The knowledge formalization continuum frees developers from putting the knowledge into a single, fixed representation scheme at the very beginning, but asserts that even text and figures are "first class" knowledge objects. The original nature of the knowledge itself remains the same no matter if it is represented by a textual document or by a rule base. Thus, the formalization from the textual form to an explicit rule base means a gradual transition, but the original nature of the considered knowledge remains unchanged.

It is important to notice that the knowledge formalization continuum is neither a physical model nor a methodology for developing knowledge bases. It rather should

[1] see WordNet/Wikipedia for full definitions/explanations

be seen as a metaphor of the knowledge development process. It helps the domain specialists to see even raw data, such as text and multimedia, as first-class knowledge that can be transformed by gradual transitions to more formal representations when required.

In summary, knowledge can be represented at different degrees of formality, and within the knowledge formalization continuum transitions of these degrees are proposed. In the extreme cases knowledge about a domain is given as data at a very informal level (images, text) or the knowledge is represented by formal knowledge representations such as decision trees or functional models. On the one hand, data given in *textual documents* denotes the lowest possibility of formality. On the other hand, *functional models* store knowledge at a very detailed formality. See Figure 1 for an example depiction of the different knowledge representations possible in the knowledge formalization continuum. This is certainly not an exhaustive enumeration of all possible representations of knowledge here, and the depicted order of representations between data and knowledge is not meant to be explicit. In fact, it appears difficult/impossible to define a total order of the representations in a general manner. The depicted order was motivated by the level of possible expressiveness with respect to the reasoning power of built system using the particular representation as knowledge. For example, text can be used for standard keyword-based search and retrieval, whereas semantically annotated text allows for semantic queries and navigation. At the right end, knowledge based on rules allows for even more complex reasoning capabilities.

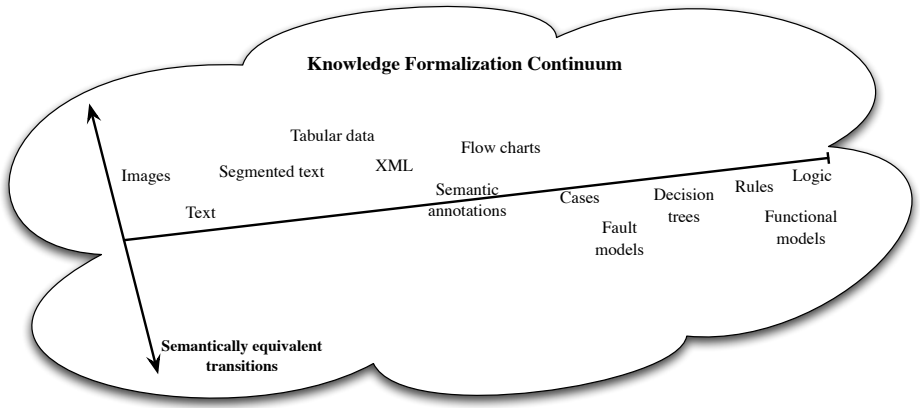

Fig. 1. Possible knowledge transitions within the knowledge formalization continuum.

Every level of formality has its own advantages and drawbacks. For example, textual knowledge can be easily elicited and often is already available in the domain. No prior knowledge with respect to tools or knowledge representation is necessary. However, automated reasoning using the textual knowledge is not possible with current state–of–the–art methods, and the knowledge can be retrieved only by using string-based matching methods but not by semantic queries. Logic rules or models are well-suited for auto-

mated reasoning, and queries can be processed on the semantic level. In contrast to textual knowledge, the acquisition of rules and models is a complex and time-consuming task. Usually, authors need prior training before effectively building knowledge bases on the explicit level with respect to knowledge engineering principles as well as tools that support such knowledge modeling. For a given knowledge base, that is formalized in a particular knowledge representation, there often exist semantically equivalent transitions (indicated by the second axis in Figure 1). For example, a fault model based on set-covering models can be often also represented in a rule base which in turn may be modelled by a special purpose logic dialect. However, representations on the right side are usually able to store more expressive knowledge. Often, the knowledge is brought to a semantically equivalent transition in order to simplify the extension by additional domain knowledge. For example, a knowledge base represented by fault models can be transferred to a rule base in order to allow for a fine-grained definition of conditioning findings for a target concept.

Between the two extremes (text vs. logic) there exists a wide range of formats representing knowledge at different degrees of formality. Any degree can be the most useful representation for building a knowledge base in a specific application project. For a given project it is an important and difficult task to select the most appropriate transition as the target representation. Since often (fragments of) knowledge are already available in textual or tabular form, the development process focuses on bringing the existing forms to an appropriate level. Although, it typically becomes necessary to fill in missing parts of the knowledge, the original nature of the knowledge remains. Thus, moving to a more formal transition can require the more explicit description of the knowledge and can enrich the resulting knowledge with additional semantics made explicit. It is worth noticing, that every transition is a distinct operation that modifies the knowledge representation. However, the mental model of the knowledge remains basically the same.

2.1 Methods for the Knowledge Formalization Continuum

The movement between two transitions is supported by already existing and established methods. Results from the following research areas can be applied, when going from explicit transitions to less explicit levels of knowledge:

- *Natural language generation* techniques, for example [2].
- *Visualization techniques*, for example [3].
- *Knowledge explanation* methods, for example [4].

The transition of the available knowledge to a less formal level is sometimes required for a number of reasons: For example, in commercial systems the built knowledge base needs to be reviewed by external specialists before deployed into practice. The transformation to a natural language text in addition to visualizations can help to present an understandable but precise version of the knowledge base for non-knowledge engineers. Furthermore, the presented methods are useful to produce a human-understandable output of the derived facts of the knowledge base, thus giving explanations of the system's behavior.

Typically, little structured/unstructured information is transformed to a more explicit level; here methods from the following disciplines will be helpful:

- *Text Mining*, *Ontology Learning*, and *Natural Language Processing* in general for the machine–enabled extraction of concepts from texts, their taxonomic ordering and the discovery of basic relations between found concepts, see [5] for example.
- *Controlled Languages* to automatically interpret a restricted subset of natural language text as logic formulas, for example an overview is given in [6].
- *Refactoring* methods to support manual changes of explicit knowledge without changing the intended semantics, for example [7]. They are often used to accomplish vertical transitions to a semantically equivalent version within the same knowledge representation, but are also helpful to restructure the knowledge to a less/more formalized level.
- *Manual Knowledge Elicitation* methods, that are applied when it is not reasonable or tractable to use (semi-)automated methods sketched above.

In an example application project we have knowledge already available contained in a textual form such as Word files and semi-structured Excel sheets. By using ontology learning methods we are able to extract relevant ontological concepts and basic relations afterwards. In subsequence, strongly formalized models are (manually) defined to formulate enhanced relations between the concepts. The initial textual knowledge is still available but now annotated by the added forms of formalized knowledge.

2.2 Implications

The knowledge formalization continuum embraces the fact that knowledge is usually represented at varying levels of formality. The continuum supports the entrance of the knowledge engineering process at an arbitrary level of formality and offers possible transitions of the knowledge to a level where its *cost/benefit principle* [1, p. 56] is (in the best case) optimal. In typical projects, prior knowledge of the domain is already at hand, often in form of text documents, spreadsheets, flow-charts and data bases. These documents build the foundational reference of the classic knowledge engineering process, where a knowledge engineer models the domain knowledge based on these documents in addition to further knowledge provided by domain specialists. The actual utility and applicability of the knowledge usually depends on the particular application.

The knowledge formalization continuum does not postulate to transform the entire collection into a knowledge base at a specific degree, but to perform transitions on *parts* of the collection when it is possible *and* appropriate. This takes into account that sometimes not all parts of a domain can be formalized at a specific level or that the formalization of the whole domain knowledge would be too complex, considering costs and risks. In consequence, a system working on the knowledge formalization continuum need to be able to support the knowledge engineering process at different levels of formality. However, it also should be able to support the knowledge sharing process, i.e., its actual usage, at varying formalization levels.

Following, the *cost/benefit principle* it need to be possible to transform the parts of the knowledge to a level of formality, where the (knowledge engineering) costs are minimized and the benefits of using the system are maximized. Therefore, the knowledge

formalization continuum not only needs to support the transitions of particular parts of the knowledge but also should be able to keep references between the less and more formalized parts of the entire knowledge collection.

3 A Semantic Wiki as an Integrated Tool to Support the Knowledge Formalization Continuum

We motivate that an extensible semantic wiki is useful to serve as a knowledge engineering tool on the knowledge formalization continuum, since it allows the integration of knowledge at different levels of formality. Thus, it tries to weaken the *flexibility dilemma* described in the introduction. The use of a semantic wiki additionally helps to target the first dilemma, i.e., the *single expert dilemma*. A semantic wiki naturally allows for a distribution of the development process over a group of domain specialists due to its open and web-based implementation. Collaboration is supported by many standard features of wikis, for instance versioning and discussion pages. However, the dilemma is only weakened by providing a technical platform for a collaborative engineering process, the interesting question of how to ensure a certain level of quality of the knowledge remains, and needs to be solved by appropriate evaluation methods, for example see [8]. The following example demonstrates the idea of the knowledge formalization continuum by a possible engineering trail of a recommendation system that is built using the semantic wiki KnowWE [9].

3.1 The Semantic Wiki KnowWE

In most semantic wikis every concept is represented by a distinct wiki page, where the concept is described by textual documents and multimedia content. Text phrases are semantically annotated with properties of a given wiki ontology; in most cases new properties can be defined in an ad-hoc way. Recent examples of semantic wikis are Semantic MediaWiki [10], IkeWiki [11], and SweetWiki [12]. The semantic knowledge wiki KnowWE [9] further allows for the intuitive capture and use of explicit problem-solving knowledge that is applied to derive particular concepts. In addition to the possibility to express strong problem-solving knowledge, such as rules and models, it also provides alternative interfaces to engineer knowledge at lower levels of formality.

Figure 2 shows a sports advisor wiki, which is an example system for demonstrating the functionality of KnowWE. Here, the form of sports *Swimming* is shown by a describing text, a picture and interactive elements within the text. A visitor can use the wiki as a recommender system in order to get a proposal of a sports form for an entered user profile. For example, the vistor enters new facts by clicking on links in the system; in the shown example (2a) the user enters some values for the question *Motivation*. The system instantly derives solutions (recommendations) when new facts (attributes of the user profile) are entered. Here, the solutions can be inferred based on the given findings (2b). All appropriate solutions are shown in the right pane of the wiki, for example the solution *Swimming* was derived as a suitable solution, but the solution *Jogging* is also suggested for further consideration. By clicking on the solution names the user can easily navigate to the corresponding wiki articles describing the sport forms in more detail.

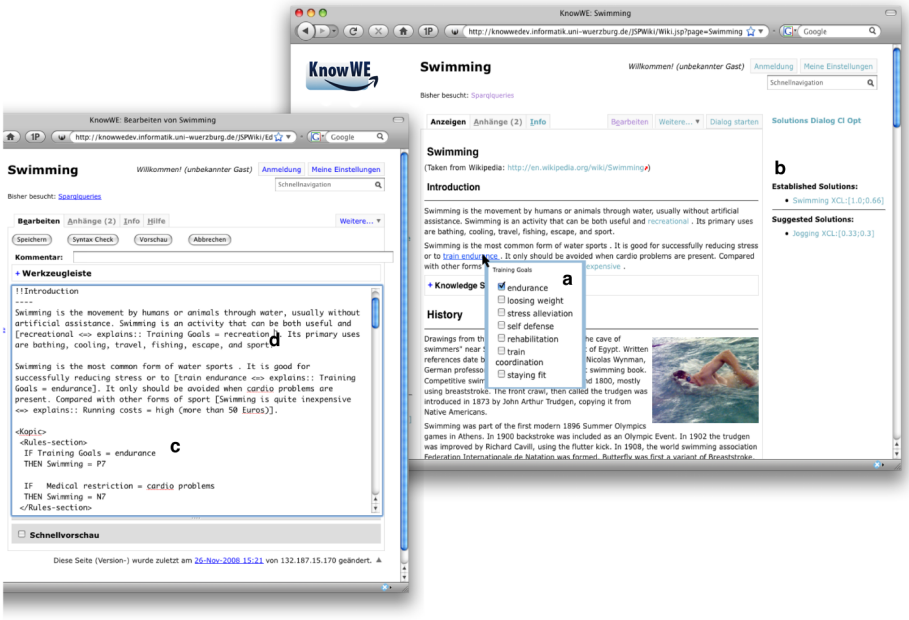

Fig. 2. The semantic wiki KnowWE at a glance: Interactive interviews with the user (a) are used to derive new concepts as solutions (b). Knowledge is entered, for example, by inline annotations (d) or explicit problem-solving knowledge such as rules (c).

The derivation knowledge for every solution is entered together with the remaining content of the wiki article. By clicking the edit button the wiki page and its corresponding content, respectively, can be modified. Here, the user can insert a special knowledge topic (Kopic) into the standard text, for example, to enter rules describing the domain knowledge (2c), but he is also able to semantically annotate particular text phrases with concepts of the application ontology (including solutions and findings of the shown sports advisor example, see 2d). Existing annotations are used for inline answers in the view mode of the wiki.

3.2 Building a Sports Recommender based on the Knowledge Formalization Continuum

The following example shows subsequent steps that describe transitions within the knowledge formalization continuum. We use the sports advisor demonstration sketched above in our example. Here, the knowledge already available in the continuum describes relevant facts about the forms of sports, such as accomplished training goals, costs, and medical restrictions. In subsequent steps we drive the existing knowledge to more formalized transitions.

Initial Filling. We start by filling the wiki with text and multimedia (pictures and movies) describing the different forms of sport, for example *Running*, *Swimming*, and

Cycling. It is reasonable that for every form of sports a wiki page should be created, i.e., after the initial filling phase there exist pages about running, swimming, and cycling. However, also wiki articles about further domain facts exist, for example allergies or muscles. In general, it is reasonable to set up one distinct wiki article for each distinct concept of the domain, thus following a common paradigm of (semantic) wikis. For example, an excerpt of an article about swimming is as follows

> ...Swimming is the most common form of water sports. In particular it is recommended for people with back problems because it trains the back muscles ... However, people with skin allergies should avoid swimming. ...

At this point the wiki can be used as a simple and traditional information system specialized on sports, where users can search and browse through the available content.

Annotating Articles. We propose to annotate every wiki article with its semantic concept, thus making explicit that a specific article *is about* a specific concept. For instance, we annotate the article about swimming with the concept *Swimming*. At this point, only a very general ontology of concepts is required to represent the domain concepts already contained in the wiki. As a benefit of this step it becomes possible to offer a low-end version of semantic search and navigation, that will be more useful when concepts are carefully structured in a hierarchy.

Annotation by Properties. The next step tries to identify the typical features of every concept described in the available text. These findings are then annotated as properties of the article's concept. In the example above the text about swimming would then transform as follows (new/changed text is given in bold letters):

> ...Swimming is the most common form of **[hasFinding::water sports]**. In particular it is recommended for people with back problems because it **[hasFinding::trains the back muscles]**. ...
> However, people with **[isContradictedBy::skin allergies]** should avoid swimming. ...

In the given example, the text phrases *water sports*, *trains the back muscles*, and *skin allergies* are annotated by the properties *hasFinding* and *isContradictedBy*, respectively. Each annotation performs the creation of an RDF triple with the article's concept (here, *Swimming*) as the subject, the property's name as the predicate, and a reference to the particular text phrase as the object. The use of properties implies the extension of the simple domain ontology of sport forms defined before. In the given example, we introduced the properties *hasFindings* and *isContradictedBy*. With the properties defined in the wiki an extended version of semantic search and navigation becomes possible. For example, we are now able to query findings (as text phrases) that exclude a specific form of sport, i.e., "return all text phrases that represent the contradiction of a given sports form".

In a further step, it is reasonable to "semantify" the text phrases representing the particular properties of a concept. Thus, we gradually extend the existing annotation by explicit concepts describing the ranges of the properties.

...Swimming is the most common form of water sports [hasFinding:: **Medium = in water**]. In particular it is recommended for people with back problems because it trains the back muscles [hasFinding:: **Trained muscles = back**]. ... However, people with skin allergies [isContradictedBy:: **Medical restrictions = skin allergy**] should avoid swimming. ...

In the shown example, the text phrase *trains the back muscles* is moved out of the annotation and replaced by the explicit concept *Trained muscles* having a concrete value *back*. Furthermore, the last annotation describes that the text phrase *skin allergies* is annotated by the value *skin allergy* assigned to the concept *Medical restrictions*. This implies the extension of the ontology by appropriate concepts representing the findings for the different forms of sport. If these concepts are defined in advance, then natural language processing methods can be used for a semi-automatic annotation of the text. In consequence, a full-fledged semantic search and navigation becomes possible, where the relation of a specific finding value to all available sport concepts can be queried, for example.

Generation of Explicit Problem-Solving Knowledge. In some cases, the use of semantic annotations is not sufficiently expressive for a given application project. Then, it becomes necessary to transform to a higher level of formality by generating and extending strong problem-solving knowledge out of the existing annotations. In the following we aim to define knowledge to actually *derive* particular forms of sports based on entered user findings. For this reason, we collect all properties, that set a form of sport in relation with a finding that can be entered by the user. In the given example, we collect the properties *hasFinding* and *isContradictedBy*. The semantic wiki KnowWE offers scripts that automatically convert these properties either into set-covering models or rules. For further properties with a different semantics the scripts certainly need to be adapted. In the initial step, such a conversion denotes the transition of the available knowledge into an (almost) semantically equivalent version. However, in this case the target representation allows for richer possibilities to represent further elements of the knowledge base.

Set-Covering Models. The following shows a transition of the annotation to a set-covering model [13], where a set-covering model describes all typical/relevant findings for a solution. The given textual markup to be used in wikis was introduced in [14]. In our example, the solution concepts are corresponding to the concepts representing the wiki articles, and findings are defined as the target concepts of the included properties. Each of the collected properties is compiled by the script into a line of the set-covering model. The value of a *hasFinding* property is represented as a simple line (denoting the positive expectation of this finding), for example *Trained muscles = back*. For a *isContradictedBy* property the conversion additionally adds a *[- -]* at the end of the generated line in order to represent the negative expectation of this finding, for example see *Medical restrictions = skin allergy*.

```
Swimming {
   Medium of sports = water
   Type of sport = individual
   Trained muscles = back
   Running costs >= medium
   Medical restrictions = skin allergy [- -]
}
```

Bold-faced letters are (hand-crafted) additions to the model, that have been made after the transition. For instance, two further findings are describing the type of sport and the running costs. The explicit representation in the model points to an extension of the formalized knowledge, although this information is already available in the text of the wiki article.

Rules. In the following example block, a simple rule-based version of the annotations made is shown. In this simple example, one rule is created by a script collecting all *hasFinding* properties as well as one rule for every *isContradictedBy* property. Of course, this simple conversion not necessarily conforms with the intended semantics of the made annotations and therefore is meant as a starting point for further (manual) adaptations.

```
if Medium of sports = water
   and Type of sport = individual
   and Trained muscles = back
   and Running costs >= medium
then derive Swimming

if Medical restrictions = skin allergy
then exclude Swimming
```

The transition to a more expressive knowledge representation such as set-covering models and rules becomes necessary when artifacts of the domain cannot be expressed by semantic annotations anymore. As a benefit, the knowledge can then be used for more effective reasoning ranging from complex semantic queries to the generation of problem-solving interviews, where appropriate solutions for a given problem are derived based on an interactive interview.

4 Conclusions

Domain knowledge is commonly available at different levels of formality. We introduced the knowledge formalization continuum to cope with this problem, and we sketched methods to work with the continuum. The semantic wiki KnowWE was introduced as a tool to support the knowledge formalization continuum, whereas many other semantic wikis also can be used to serve as suitable platform. The presented idea of the knowledge formalization continuum is related to the ontology classification using a three-dimensional matrix as introduced in [15]. Here, a categorization of the knowl-

edge to be modelled is given when designing a knowledge-based system. Schaffert et al. distinguish between *model scope*, *model acceptance* and the *level of expressiveness*, where the latter defines a subspace of the presented knowledge formalization continuum. The level of expressiveness ranges from light-weight ontologies, with term lists as the least expressive one, to heavy-weight ontologies with very-expressive constraints as the most expressive representative. Whereas functional models and logic programs can be interpreted as "very-expressive constraints" in some ways, the knowledge formalization continuum also considers textual documents as less expressive occurrences of knowledge apart from term lists.

References

1. Lidwell, W., Holden, K., Butler, J.: Universal Principles of Design. Rockport Publishers (October 2003)
2. Reiter, E., Dale, R.: Building Natural Language Generation Systems. Cambridge University Press, Cambridge (2000)
3. Geroimenko, V., Chen, C., eds.: Visualizing the Semantic Web. 2 edn. Springer (2006)
4. Roth-Berghofer, T.R.: Explanations and Case-Based Reasoning: Foundational issues. In Funk, P., González-Calero, P.A., eds.: Advances in Case-Based Reasoning, Springer-Verlag (September 2004) 389–403
5. Dale, R., Moisl, H., Somers, H., eds.: A Handbook of Natural Language Processing: Techniques and Applications for the Processing of Language as Text. Marcel Dekker Inc. (2000)
6. Fuchs, N.E., Kaljurand, K., Kuhn, T.: Attempto Controlled English for knowledge representation. In Baroglio, C., Bonatti, P.A., Małuszyński, J., Marchiori, M., Polleres, A., Schaffert, S., eds.: Reasoning Web, Fourth International Summer School 2008. Number 5224 in Lecture Notes in Computer Science, Springer (2008) 104–124
7. Baumeister, J., Seipel, D., Puppe, F.: Refactoring methods for knowledge bases. In: EKAW'04: Engineering Knowledge in the Age of the Semantic Web: 14th International Conference, LNAI 3257, Berlin, Springer (2004) 157–171
8. Baumeister, J., Nalepa, G.J.: Verification of distributed knowledge in semantic knowledge wikis. In: FLAIRS'09: Proceedings of the 22th International Florida Artificial Intelligence Research Society Conference. (2009)
9. Reutelshoefer, J., Baumeister, J., Puppe, F.: Ad-hoc knowledge engineering with semantic knowledge wikis. In: SemWiki'08: Proceedings of 3rd Semantic Wiki workshop - The Wiki Way of Semantics (CEUR Proceedings 360). (2008)
10. Krötzsch, M., Vrandecić, D., Völkel, M.: Semantic MediaWiki. In: ISWC'06: Proceedings of the 5th International Semantic Web Conference, LNAI 4273, Berlin, Springer (2006) 935–942
11. Schaffert, S.: IkeWiki: A semantic wiki for collaborative knowledge management. In: STICA'06: 1st International Workshop on Semantic Technologies in Collaborative Applications, Manchester, UK (2006)
12. Buffa, M., Gandon, F., Ereteo, G., Sander, P., Faron, C.: SweetWiki: A semantic wiki. Web Semantics **8**(1) (2008) 84–97
13. Peng, Y., Reggia, J.A.: Abductive Inference Models for Diagnostic Problem-Solving. Springer, Berlin (1990)
14. Baumeister, J., Reutelshoefer, J., Puppe, F.: Markups for knowledge wikis. In: SAAKM'07: Proceedings of the Semantic Authoring, Annotation and Knowledge Markup Workshop, Whistler, Canada (2007) 7–14
15. Schaffert, S., Gruber, A., Westenthaler, R.: A semantic wiki for collaborative knowledge formation. In: Proceedings of SEMANTICS 2005 Conference, Trauner Verlag (2006)