Int. J. Appl. Math. Comput. Sci., 2011, Vol. 21, No. 1, 27–39
DOI: 10.2478/v10006-011-0002-5

ENGINEERING INTELLIGENT SYSTEMS ON THE KNOWLEDGE FORMALIZATION CONTINUUM

JOACHIM BAUMEISTER, JOCHEN REUTELSHOEFER, FRANK PUPPE

Institute of Computer Science
University of Würzburg, Am Hubland, 97074 Würzburg, Germany
e-mail: {baumeister,reutelshoefer,puppe}@informatik.uni-wuerzburg.de

In spite of their industrial success, the development of intelligent systems is still a complex and risky task. When building intelligent systems, we see that domain knowledge is often present at different levels of formalization—ranging from text documents to explicit rules. In this paper, we describe the knowledge formalization continuum as a metaphor to help domain specialists during the knowledge acquisition phase. To make use of the knowledge formalization continuum, the agile use of knowledge representations within a knowledge engineering project is proposed, as well as transitions between the different representations, when required. We show that a semantic wiki is a flexible tool for engineering knowledge on the knowledge formalization continuum. Case studies are taken from one industrial and one academic project, and they illustrate the applicability and benefits of semantic wikis in combination with the knowledge formalization continuum.

Keywords: knowledge engineering, knowledge acquisition, semantic wiki.

1. Introduction

Intelligent systems, especially knowledge-based solutions, are successfully implemented in today's enterprises. We experience the application of intelligent systems in different ”flavors”, ranging from (semantically) enriched information systems to sophisticated model-based expert systems. Consequently, the knowledge embodied in these different types of systems differs strongly from natural language text to explicit knowledge in the form of rules or models.

Although such systems proved their applicability and benefits in many domains, the engineering and maintenance of the underlying knowledge bases is still a complex and costly task. In this work, we introduce the *knowledge formalization continuum* as a conceptual metaphor that gives domain specialists a flexible mental model of the knowledge that is planned to be formalized. The knowledge formalization continuum emphasizes that usable knowledge ranges from very informal representations (such as text and images) to very explicit representations (such as logic formulae or consistency-based models). The metaphor frees specialists and engineers to commit to a particular degree of knowledge formalization at an early stage of the development project but offers a versatile understanding of the formalization process. Furthermore, we clam that a semantic wiki (Schaffert *et al.*, 2008) is an appropriate workbench for the development of intelligent systems using the knowledge formalization continuum, since it supports the creation and evolution of knowledge in various facets and thus supports the idea of the knowledge formalization continuum.

The rest of the paper is organized as follows. In Section 2, we introduce the concept of the knowledge formalization continuum in greater detail. We explain possible representations on the knowledge formalization continuum and we present methods that facilitate transitions between the representations. We further present the KnowWE semantic wiki in Section 3 and we discuss its application within the knowledge formalization continuum. A number of examples and case studies using the continuum are reported in Section 4, and we conclude the paper in Section 5.

2. Knowledge formalization continuum

In general, a *continuum* describes a “continuous sequence in which adjacent elements are not perceptibly different from each other, but the extremes are quite distinct” (*Oxford English Dictionary of Current English*, 2008).

We interpret this definition of a continuum in the

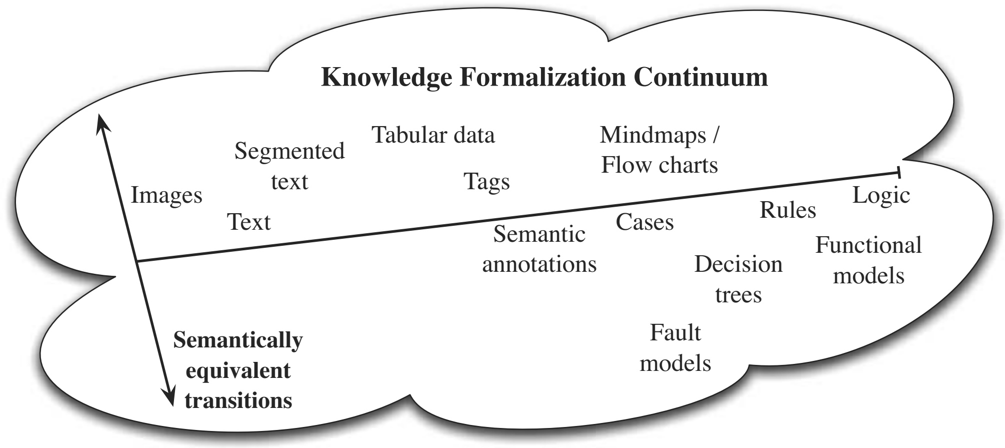


Fig. 1. Possible knowledge representations of the knowledge formalization continuum.

context of knowledge representations as follows. The same domain knowledge can be displayed in various types of knowledge representations, where adjacent representations are similar to each other—for example, tabular data and XML, but more extreme representations are quite distinct, for instance, *text* vs. *logic rules*. In such a *knowledge formalization continuum*, gradual transitions on formalization degrees of the same knowledge are possible, but the knowledge to be modelled experiences no abrupt changes or "discontinuities". Albeit we interpret the continuum as a mental concept of alternative knowledge representations, it is important to notice that the represented domain knowledge remains the same. The knowledge formalization continuum also emphasizes that knowledge undergoes an ongoing, continuous development where changes can also imply a modification of the representation.

We note that the knowledge formalization continuum is neither a physical model nor a methodology for developing knowledge bases. It should rather be seen as a metaphor of the knowledge development process. It helps domain specialists to see even plain data, such as text and multimedia, as first-class knowledge that can be transformed by gradual transitions to more formal representations *when* required. On the one hand, data given by *textual documents* denote one of the lowest possibilities of formalization. On the other, *functional models* store knowledge at a very formal level.

Figure 1 illustrates different knowledge representations possible within the knowledge formalization continuum. This enumeration of representations is certainly not intended to be exhaustive, neither is, the depicted order of representations between data and knowledge meant to be explicit. In fact, it appears difficult/impossible to define the 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 systems built using the particular representation. For example, text can be used for a standard keyword-based search and retrieval, whereas semantically annotated text allows semantic queries and navigation. At the right end, knowledge based on rules supports even more complex reasoning capabilities.

Every level of formalization 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 textual knowledge is not possible with current state-of-the-art methods, and knowledge can be retrieved only by using string-based matching methods, but not by applying semantic queries. Logic rules or models are well suited for automated reasoning, and queries can be processed at the semantic level. In contrast to textual knowledge, the acquisition of rules and models is typically a complex and time-consuming task. Authors need prior training before effectively building knowledge bases at the explicit level with respect to knowledge engineering principles as well as tools that support such knowledge modeling. For a given knowledge base, which is formalized using a particular knowledge representation, there often exist semantically equivalent transitions (indicated by the second axis in Fig. 1). For example, a fault model based on set-covering models can be often also represented by a rule base which in turn may be modelled by a special purpose logic dialect. Knowledge is often brought to a semantically equivalent representation in order to enable extensions by additional domain knowledge. For example, a knowledge base represented by fault models can be transitioned to a rule base in order to allow 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 formalization. Any degree can be the most useful representation for a specific application project.

For a given project, it is a difficult but important task to select the most appropriate formalization level as the target representation. Since knowledge (or its fragments) is often is 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 knowledge, its original nature remains. Thus, moving to a more formal representation can require a more explicit description of 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 knowledge remains basically the same.

2.1. Transitions on the knowledge formalization continuum. Transitions between different levels of formalization, for example, from text to cases, can be done a manual and sometimes in a semi-automated way. Transitions from explicit knowledge to less explicit levels of knowledge are supported by the following methods:

- *natural language generation* techniques (Reiter and Dale, 2000),
- *visualization techniques* (Geroimenko and Chen, 2006; Zacharias, 2007; Card, 2003; Baumeister and Freiberg, 2010),
- *knowledge explanation* methods (Roth-Berghofer, 2004).

The direction of transition into a less formal knowledge representation is required, for example, to review/evaluate the developed knowledge bases. Here, less formal but precise visualizations of the knowledge base help to understand the semantics of the implemented knowledge.

A typical direction in knowledge engineering projects is the transition of hardly structured/unstructured data into a more explicit representation; here, the following disciplines propose useful methods:

- *Text mining*, *ontology learning*, and *natural language processing* in general for machine–enabled extraction of concepts from texts, their taxonomic ordering, and the discovery of basic relations between concepts found (e.g., Dale *et al.*, 2000; Buitelaar et al., 2005; Feldman and Sanger, 2006).
- *Controlled languages* to automatically interpret a restricted subset of natural language text as logic formulae, (Fuchs *et al.*, 2008).
- *Templates/forms* to enter knowledge on a basic level by using as much natural language as possible.
- *Refactoring* methods to support manual changes explicit knowledge without changing the intended semantics (e.g., Gil and Tallis, 1997; Baumeister *et al.*, 2004; Reutelshoefer *et al.*, 2009). They are often used to accomplish vertical transitions to a semantically equivalent version within the same knowledge representation, but are also helpful to restructure knowledge to a less/more formalized level.
- *Manual knowledge elicitation* methods, which are applied when it is not reasonable or tractable to use (semi-)automated methods as those sketched above.

In an example application project we can imagine to have knowledge already available contained in a textual form such as MS Word documents and semi-structured MS Excel sheets. By using ontology learning methods we are able to extract relevant ontological concepts and basic relations afterwards. Accordingly, 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.

In the future, we envision that the methods mentioned above will work perfectly together to build and evolve knowledge bases. Most state-of-the-art methods, however, are currently only capable of providing a seed for knowledge base development that requires manual knowledge refinement and formalization.

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 knowledge to a level where its *cost/benefit principle* (Lidwell *et al.*, 2003, p. 56) is (in the best case) optimal. In typical projects, prior knowledge of the domain is already at hand, often in the 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 domain knowledge based on these documents. The actual utility and applicability of knowledge usually depends on a particular instance.

The knowledge formalization continuum does not postulate the transformation of the entire collection into a knowledge base at a specific degree but the performance of transitions on *parts* of the collection when it is possible *and* appropriate. This takes into account the fact 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 is required to support the knowledge engineering process at different levels of formalization. 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* there is a need for a possibility to transform parts of knowledge to a level of formalization 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 knowledge but also should be able to keep references between the less and more formalized parts of the entire knowledge collection.

2.3. Related approaches. The presented concept of the knowledge formalization continuum is referenced in the literature in a more general context. Uschold and Gruninger (2004) discuss different kinds of ontologies ranging from terms to general logic. They also name this spectrum of possible representations a *continuum* but do not propose transitions between the particular kinds, nor propose to use them jointly when developing intelligent systems. Similarly, McGuinness (2003) introduces the *ontology spectrum* but concentrates on the expressiveness of ontology languages. Gruber (2008) also correlates the formalization degree of data with the depth of inference, and he discusses the "value/cost" trade-off of the captured knowledge. He puts the matrix of the formalization degree and the development costs in the context of Social Semantic Web applications.

Further, Schaffert *et al.* (2006) distinguish between the 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 representative, to heavy-weight ontologies with very-expressive constraints as the most expressive one. 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. Millard *et al.* (2005) describe an approach that investigates the formality of documents of hypertext systems. A vector-based model of the formality of semantics of these documents is given. Their scale defines a subset of the presented continuum ranging from plain text to RDF data, which is sufficient in the context of the domain considered.

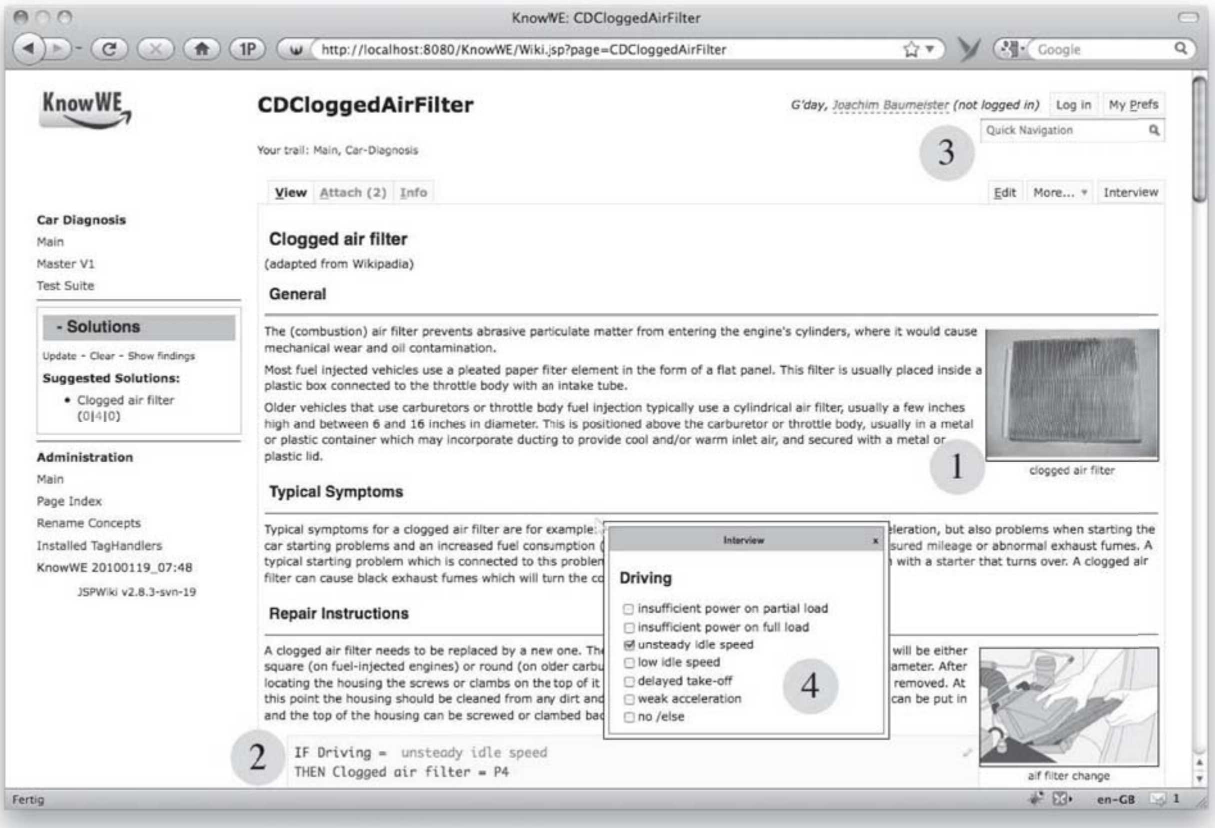
In the context of the CommonKADS methodology, Schreiber *et al.* (2001) describe a matrix of knowledge formalizations for knowledge elicitation techniques. Here, the matrix correlates tacit and explicit knowledge with concept knowledge and process knowledge. In this way, the process of knowledge development is explicitly considered. However, the matrix does not take into account transitions between the particular elements.

3. Engineering the knowledge formalization continuum with semantic wikis

In the previous section, we introduced the ideas of the knowledge formalization continuum and stated that multiple knowledge representations have to be considered during the development of a knowledge base. In this section, we first describe the concept of semantic wikis, and we show how such systems are used as knowledge engineering tools. Consequently, we demonstrate the use of semantic wikis for developing knowledge systems within the knowledge formalization continuum.

3.1. Knowledge engineering with semantic wikis. In recent years, the advent and success of Web 2.0 applications, such as wikis, blogs, and social networks, has changed the way people use the Internet. When compared with traditional web sites, Web 2.0 applications explicitly involve users as primary contributors to the system. Thus, the value of particular systems usually grows with the increasing contribution of users. Not only private life, but also daily business is influenced by the success of Web 2.0 approaches. One prominent example is the wide-spread use of wikis as flexible knowledge management tools, both in personal life and business environments. The content of a wiki can be created and modified by clicking an (often mandatory) *edit* button located on the web page. Due to the simple markup (less verbose than HTML), users can easily author web pages. Wikipedia is certainly the most popular example of wiki systems, where informal *world knowledge* is created and updated by a wiki.

In spite of their success, standard wiki systems show limitations, especially when using and sharing the knowledge included. Usually, only a full-text search is possible for knowledge retrieval, and knowledge across different wiki pages cannot be aggregated to a unified result. This issue motivated the development of semantic wikis that extend standard wikis by an explicit ontological layer defined by semantic annotations of the wiki content. Thanks to semantic annotations, knowledge reuse is improved by a semantic search and semantic navigation (Schaffert *et al.*, 2008). At the same time, semantic wikis successfully serve as ontology development tools, that provide a simple, web-based interface to build semantic applications. Current examples of semantic wiki implementations are, for instance, IkeWiki (Schaffert, 2006), KnowWE (Baumeister *et al.*, 2010), MoKi (Ghidini *et al.*, 2009), PlWiki (Nalepa, 2009), Semantic MediaWiki (Krötzsch *et al.*, 2006), and SweetWiki (Buffa *et al.*, 2008).



KnowWE: CDCloggedAirFilter

http://localhost:8080/KnowWE/Wiki.jsp?page=CDCloggedAirFilter

KnowWE

CDCloggedAirFilter

G'day, Joachim Baumeister (not logged in) Log in My Prefs

Quick Navigation

3

Your trail: Main, Car-Diagnosis

View Attach (2) Info Edit More... Interview

Car Diagnosis
Main
Master V1
Test Suite

- Solutions
Update - Clear - Show findings
Suggested Solutions:
- Clogged air filter (0|4|0)

Administration
Main
Page Index
Rename Concepts
Installed TagHandlers
KnowWE 20100119_07:48
JSPWiki v2.8.3-svn-19

Clogged air filter
(adapted from Wikipadia)

General

The (combustion) air filter prevents abrasive particulate matter from entering the engine's cylinders, where it would cause mechanical wear and oil contamination.

Most fuel injected vehicles use a pleated paper fiter element in the form of a flat panel. This filter is usually placed inside a plastic box connected to the throttle body with an intake tube.

Older vehicles that use carburetors or throttle body fuel injection typically use a cylindrical air filter, usually a few inches high and between 6 and 16 inches in diameter. This is positioned above the carburetor or throttle body, usually in a metal or plastic container which may incorporate ducting to provide cool and/or warm inlet air, and secured with a metal or plastic lid.

1

clogged air filter

Typical Symptoms

Typical symptoms for a clogged air filter are for example: … leration, but also problems when starting the car starting problems and an increased fuel consumption (… sured mileage or abnormal exhaust fumes. A typical starting problem which is connected to this problen … with a starter that turns over. A clogged air filter can cause black exhaust fumes which will turn the co

Interview ×

Driving

☐ insufficient power on partial load
☐ insufficient power on full load
☑ unsteady idle speed
☐ low idle speed
☐ delayed take-off
☐ weak acceleration
☐ no /else

4

Repair Instructions

A clogged air filter needs to be replaced by a new one. The … will be either square (on fuel-injected engines) or round (on older carbu … ameter. After locating the housing the screws or clambs on the top of it … removed. At this point the housing should be cleaned from any dirt and … can be put in and the top of the housing can be screwed or clambed bac

air filter change

2

```
IF Driving =  unsteady idle speed
THEN Clogged air filter = P4
```

Fertig en-GB 1

Fig. 2. KnowWE semantic wiki showing an article that describes the car fault *clogged air filter* by using multiple forms of knowledge: (1) plain text, images, and (2) explicit rules. Knowledge can be used by (3) a simple text search and by (4) an interactive knowledge-based interview.

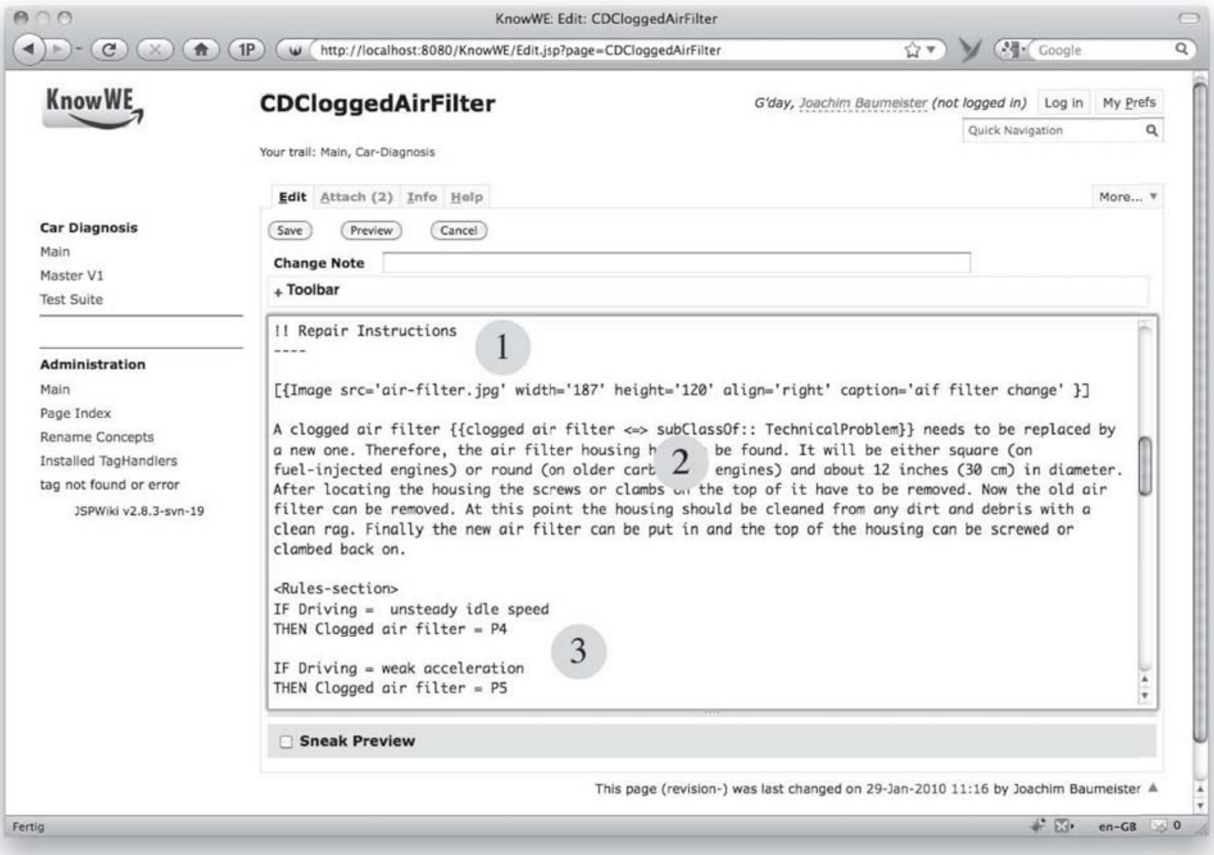
The knowledge in a semantic wiki is typically organized as follows. Every concept of the ontology is represented by a wiki article, and the content of the article informally describes the concept. Properties of the concept are defined by explicit semantic annotations within the article, where the annotations often link to other articles and concepts, respectively. In general, most semantic wiki systems are capable of developing and maintaining ontologies with the expressiveness of a subset of the ontology language OWL (Antoniou and van Harmelen, 2003).

Figure 2 shows the KnowWE semantic wiki depicting an example system for capturing knowledge about car faults diagnosis. Each car fault (here *clogged air filter*) is represented by a wiki article containing multiple knowledge formalizations: using (1) plain text and images and (2) explicit rules. The knowledge can be used by (3) a simple text search interface and (4) an interactive interview provided based on explicit knowledge.

In Fig. 3, we see the same wiki article in the edit mode: here, we notice that special markup is used to integrate different kinds of knowledge into the page. It (1) includes images depicting an air filter, (2) shows a semantic annotation stating that *clogged air filter* is a subclass of the concept *TechnicalProblem*. In (3), a block of rules is defined that are used to derive the solution *clogged air filter* in a given case.

Thus, a wiki article captures both informal and formal sources of knowledge. During knowledge formalization, the knowledge is typically distributed over the entire wiki, i.e., the knowledge base is partitioned into logical units and embedded in the corresponding wiki articles. For the partitioning of knowledge, no prior restrictions are defined and the actual organisation of the knowledge base depends on the characteristics of the domain. For instance, in many projects, partitioning with respect to solutions is more appropriate, i.e., defining an article for each solution, where problem-solving knowledge for this solution is also embedded. In other projects, an organization among the cardinal symptoms of the domain may be more appropriate.

Besides its flexible organization of knowledge, KnowWE semantic wiki also provides the possibility to define variants of the distributed knowledge base. When specifying authoritative knowledge bases in the wiki, we use the metaphor of *masters* and *servants*: a large coherent knowledge base—usually intended to be exported out of



CDCloggedAirFilter

```
!! Repair Instructions
----

[{Image src='air-filter.jpg' width='187' height='120' align='right' caption='aif filter change' }]

A clogged air filter {{clogged air filter <=> subClassOf:: TechnicalProblem}} needs to be replaced by
a new one. Therefore, the air filter housing h[illegible] be found. It will be either square (on
fuel-injected engines) or round (on older carb[illegible] engines) and about 12 inches (30 cm) in diameter.
After locating the housing the screws or clambs [illegible] the top of it have to be removed. Now the old air
filter can be removed. At this point the housing should be cleaned from any dirt and debris with a
clean rag. Finally the new air filter can be put in and the top of the housing can be screwed or
clambed back on.

<Rules-section>
IF Driving =  unsteady idle speed
THEN Clogged air filter = P4

IF Driving = weak acceleration
THEN Clogged air filter = P5
```

This page (revision-) was last changed on 29-Jan-2010 11:16 by Joachim Baumeister

Fig. 3. Edit mode of the KnowWE semantic wiki: text, images, and formal knowledge can be edited using a simple markup.

the wiki—is defined on the basis of imports of knowledge bases contained in other wiki articles. This *master* then includes all knowledge of the imported articles, i.e., the *servants*, and it can be interpreted as a single, testable knowledge base. In this way, we are also able to define *variants* of knowledge bases by defining different masters importing varying collections of wiki articles. Figure 4 depicts an example of two masters of a wiki. On the whole, the wiki shown contains four servant wiki articles with knowledge bases, i.e., *Article 1* to *Article 4*. Additionally, the wiki page *Article 5* defines the master knowledge base *Master 1* by including the knowledge bases from the servant articles 1, 2, and 3. The alternative master *Master 2* defined on the page *Article 6* only includes the knowledge bases from the servant articles 3 and 4, and thus represents a different view of the entire knowledge base.

3.2. Semantic wikis and the knowledge formalization continuum. As we showed above, a semantic wiki offers flexible ways to simply combine less formal forms of knowledge (text, images) with more explicit knowledge representations (rules, models, etc.). Less formal representations of knowledge serve the development process in many ways: (i) as a startup document at the beginning of a project to informally collect knowledge about the domain, (ii) as a documentation of knowledge engineering decisions taken, (iii) as underlying tacit knowledge expressing the informal counter-part, and (iv) as describing information for concepts represented by the article. In practice, the inclusion of less formal knowledge also improves the overall maintainability of the distributed knowledge base.

In the following we demonstrate the use of a wiki when building an example sports recommender system. Here, the wiki collects articles about forms of sport and also captures explicit knowledge that recommends an appropriate form of sport for an entered user profile. The example shows subsequent transitions within the knowledge formalization continuum. 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 *Cy-*

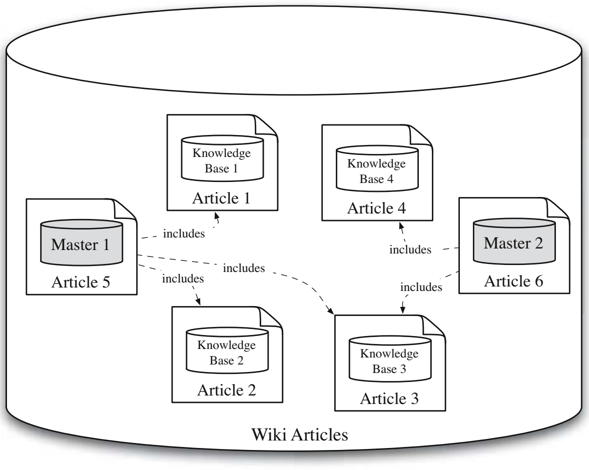


Fig. 4. Definition of knowledge variants by the specification of *masters* and *servants* of the wiki.

cling. 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 instance, allergies or muscles. In general, it is reasonable to set up one distinct wiki article for each distinct concept from 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 in 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 it 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 the semantic search and navigation, which 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 text available. 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 boldface):

> ...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 example given, the text phrases *water sports*, *trains the back muscles* and *skin allergies* are annotated with 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 sports forms defined before. In the example, we introduced the properties *hasFindings* and *isContradictedBy*. With the properties defined in the wiki, an extended version of the semantic search and navigation becomes possible. For example, we are now able to query findings (as text phrases) that exclude a specific form of sports, 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 presented 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 sports. If these concepts are defined in advance, then natural language processing methods can be used for semi-automatic annotation of the text. In consequence, a full-fledged semantic search and navigation become possible, where the relation of a specific finding value to all available sports concepts can be queried, for example.

Generating problem-solving knowledge. In some cases, the use of semantic annotations is not sufficiently expressive for a given application project. Then, transformation a higher level of formalization by generating and extending strong problem-solving knowledge out of the existing annotations becomes necessary. 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 example given, we collect the properties *hasFinding* and *isContradictedBy*. The KnowWE semantic wiki offers scripts that automatically convert these properties either into set-covering models or rules. For further properties with different semantics, the scripts certainly have 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 offers 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 (Peng and Reggia, 1990), which describes all typical/relevant findings for a solution. The given textual markup to be used in wikis was introduced by Baumeister *et al.* (2007). In our example, the solution concepts correspond to those 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 the property *hasFinding* is represented as a simple line (denoting the positive expectation of this finding), for example, *Trained muscles = back*. For the property *isContradictedBy*, 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 describe the type of sports and the running costs. The explicit representation in the model points to an extension of formalized knowledge, although this information is already available in the text of the wiki article.

Rules. In the following example block, a 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 naive conversion does not necessarily conform to the intended semantics of the annotations made, and therefore it 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
```

Transition to more expressive knowledge representations such as set-covering models and rules becomes necessary when complex relations 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 to a given problem are derived based on an interactive interview.

In the example above, we showed stepwise formalization of knowledge along the knowledge formalization continuum. In the next section, we present some case studies and we also claim that the counter direction along the knowledge formalization continuum poses interesting applications.

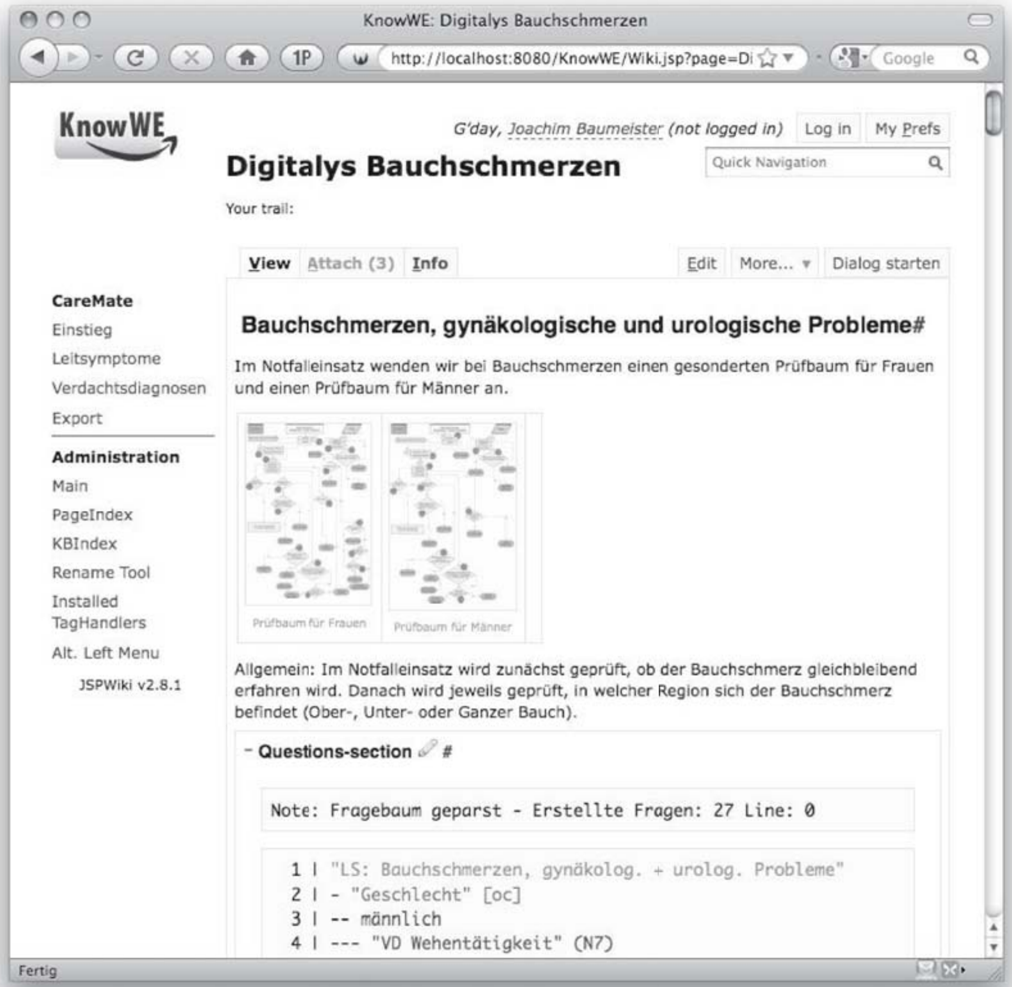
4. Case studies

In the following, we describe two case studies and demonstrate the application of the knowledge formalization continuum and the KnowWE semantic wiki, respectively. The first case study considers the development process of the medical decision-support system CareMate, whereas the second example shows the use of the knowledge formalization continuum in the context of the e-learning application HermesWiki.

4.1. Medical decision-support with CareMate.

The decision-support system *Digitalys CareMate* is commercially sold by the company Digitalys (http://`www.digitalys.de`) as part of an equipment kit for medical rescue trucks. It is a consultation system for medical rescue missions when the problem definition of a particular rescue service is complex and a second opinion becomes important. The knowledge base currently contains about 260 findings distributed over 33 questionnaires. The system is able to derive 145 distinct solutions. The compiled knowledge base resulted in a (merged) decision tree having 2051 possible diagnostic paths.

The knowledge base is developed using the KnowWE semantic wiki. In the wiki, the knowledge base



Digitalys Bauchschmerzen

Bauchschmerzen, gynäkologische und urologische Probleme#

Im Notfalleinsatz wenden wir bei Bauchschmerzen einen gesonderten Prüfbaum für Frauen und einen Prüfbaum für Männer an.

Allgemein: Im Notfalleinsatz wird zunächst geprüft, ob der Bauchschmerz gleichbleibend erfahren wird. Danach wird jeweils geprüft, in welcher Region sich der Bauchschmerz befindet (Ober-, Unter- oder Ganzer Bauch).

- Questions-section #

```
Note: Fragebaum geparst - Erstellte Fragen: 27 Line: 0
```

```
"LS: Bauchschmerzen, gynäkolog. + urolog. Probleme"
- "Geschlecht" [oc]
-- männlich
--- "VD Wehentätigkeit" (N7)
```

Fig. 5. Article of the cardinal symptom *stomach pain* (“Bauchschmerzen”) showing some text explaining the general structure of the corresponding decision tree and an overview image, which can be enlarged on click (the original screenshot is depicted in the German language).

is partitioned by their cardinal symptoms. Accordingly, every cardinal symptom is represented by a distinct wiki article. Furthermore, there exists one wiki article describing the solutions of the system. On every page of a cardinal symptom, there is some explanation text given in addition to the formalized version of the knowledge as a heuristic decision tree. For larger cardinal symptoms, there are further pages linked from the original article, where modular parts of the decision tree are formalized. In Fig. 5, the wiki article of the cardinal symptom *stomach pain* (“Bauchschmerzen”) is shown.

At the beginning of the project, the knowledge was originally captured by transferring textbook knowledge and expert knowledge into flowchart diagrams using the software MS Visio. The domain specialist was neither experienced in using (specialized) computer software nor trained in knowledge representation. The use of standard office software, such as MS Visio, appeared to be the most natural approach to capture the initial structure of the knowledge base, since no new software distracted the specialist but he or she was able to concentrate on the actual knowledge structuring. After defining the initial structure and logic of the knowledge base over a number of diagrams, we decided to transform the knowledge into a decision tree representation to allow automated execution of the knowledge by the d3web reasoner (Baumeister *et al.*, 2008).

Within the semantic wiki, both knowledge formalizations can be represented very easily: the decision trees were defined directly within the wiki text using a specialized markup (Baumeister *et al.*, 2010), whereas the original MS Visio diagrams were embedded as attachments to the corresponding wiki articles. Additional texts of the articles explain relations and design decisions. For instance, Fig. 5 shows the article of the logical unit *stomach pain* ("Bauchschmerzen"), where MS Visio diagrams, organizational comments, and decision tree knowledge of the same knowledge entity are combined. The MS Visio diagrams can be viewed in and downloaded from the wiki. Comments in the article additionally state that the decision tree logic was divided into two decision trees handling the diagnosis of stomach pain for women and for men, separately. The lower part Fig. 5 shows an excerpt of the formalized knowledge base, where first the sex ("Geschlecht") of the patient is asked.

The CareMate system runs on a touch-screen tablet, and therefore releases of the knowledge base need to be exported from the wiki into a runtime version. Here, KnowWE provides a deployment feature that compiles a predefined *master* article into an executable knowledge system.

Before the deployment of a new version of the CareMate system, all possible derivation paths of the knowledge base needed to be verified. For this task, we utilized the visualization technique DDTrees (Baumeister and Freiberg, 2010) to transfer the available knowledge into a less formalized representation. In this way, we move the knowledge to the representation level that was most appropriate for the particular task.

To summarize, the ideas of the knowledge formalization continuum were applied in many ways during the development of the CareMate knowledge base. Starting with MS Visio flowcharts formalized from textbook knowledge, the domain specialist could concentrate on knowledge organization and structuring. In a second step, the flowcharts were semi-automatically transferred to the formal knowledge representation *heuristic decision trees* in order to allow an automated reasoning process. For the verification process, the knowledge was visualized by a less knowledge-based formalization, here—DDTrees.

4.2. HermesWiki e-learning platform. HermesWiki (Reutelshoefer *et al.*, 2010) is an e-learning platform developed by historians of the Department of Ancient History from the University of Würzburg, Germany, supported by the Department of Intelligent Systems of the Institute of Computer Science. HermesWiki is a KnowWE implementation and consists of a concise and reliable overview of ancient Greek history. The knowledge provided here is intended to serve as teaching material for undergraduate students. Presently, HermesWiki contains about 800 pages covering essays, translated excerpts of original sources, and a glossary.

HermesWiki is an interesting example of the application of knowledge formalization continuum: the domain specialists—researches and teachers of ancient history—originally started by filling a standard wiki with a textual content and images. They defined categories and relations not by using explicit markup, but by formatting the wiki text following a discussed and agreed convention. This convention simplified automated extraction of knowledge from the text entered. Due to the extensibility of the wiki (Reutelshoefer *et al.*, 2009) it was easy to extend the engine by specialized plugins to automatically parse and formalize the available text.

The following block gives an example of the employed "markup by convention", where time events of history are defined by special text formatting:

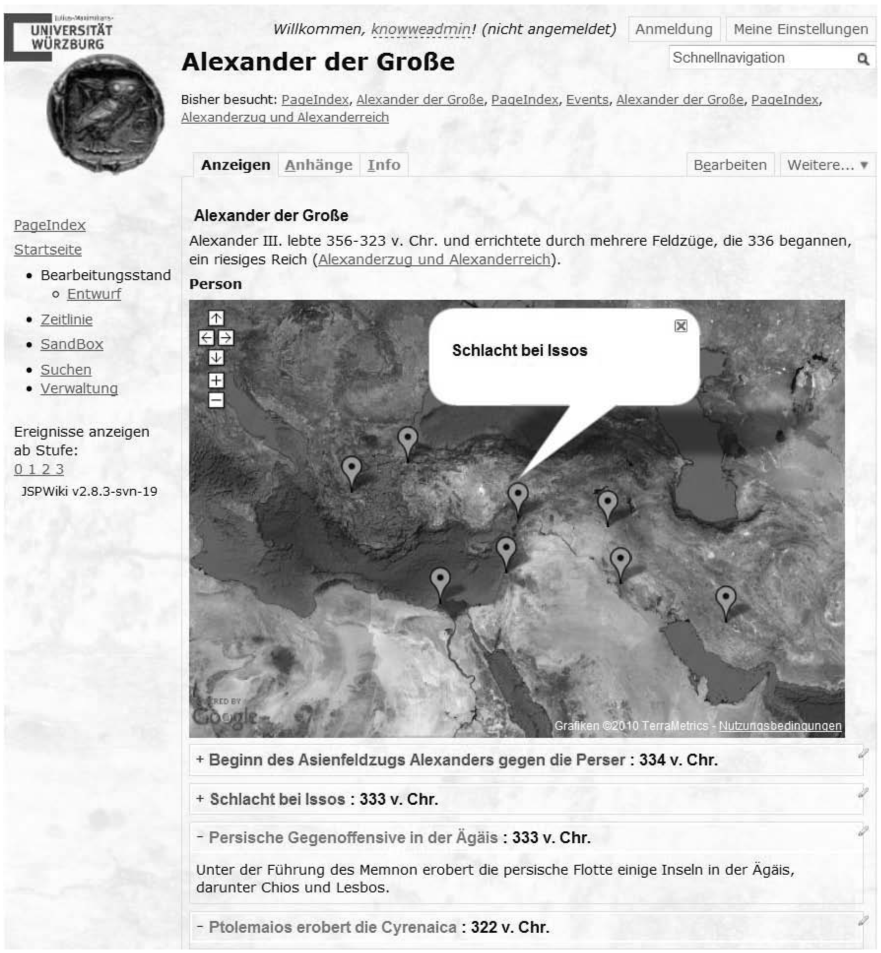
```
<<Lamian War
323b-322b

After Alexander's death the Greeks
revolted against Macedonian
rule under the lead of the Athenians.
[....]

SOURCE: Paus::1,25,3-6
SOURCE: Diod::18,8-18
>> ...
```

The example describes the event *Lamian War*, given as the title in the first line. In the second line the time event is annotated by a time stamp specifying the duration of the event considered. The text "323b-322b" means that the event occurred 323 BC until 322 BC. The following lines consist of the *body* of the event, describing its description in free text. Optionally, historical sources are appended to the text, explicitly annotated by the keyword *SOURCE*. In our example, two sources—from *Diodor* and *Pausanias*—are included.

As a benefit, the markup is human readable as free text, but it is sufficiently formal to be parsed as ontological descriptions. The ontology—generated by the markups—has multi-faceted applications: the knowledge is used for semantically enriched wiki pages, for a timeline browser, for semantically annotating Google Maps, and a dynamically generated quiz for students.

Figure 6 shows a HermesWiki article about Alexander the Great ("Alexander der Große"). Here, ontology is used to geographically highlight on a Google Map important events, in which Alexander was involved; a less formal knowledge representation (map image) is enriched by ontological knowledge. Underneath, the visualization of time events—defined within the page using the markup discussed above—is shown as collapse boxes.

The HermesWiki project is an interesting example, where the concept of the knowledge formalization continuum was intuitively applied: Starting with simple text and images, the wiki was filled very easily, some existing documents were already available and could be taken over into the wiki. Using the light-weight approach of "convention over configuration", simple markups were agreed, and explicit ontological knowledge was extracted from the formatted text. The ontological knowledge in turn was



UNIVERSITÄT WÜRZBURG

Willkommen, knowweadmin! (nicht angemeldet) Anmeldung Meine Einstellungen

Alexander der Große

Schnellnavigation

Bisher besucht: PageIndex, Alexander der Große, PageIndex, Events, Alexander der Große, PageIndex, Alexanderzug und Alexanderreich

Anzeigen Anhänge Info Bearbeiten Weitere...

PageIndex
Startseite
- Bearbeitungsstand
  - Entwurf
- Zeitlinie
- SandBox
- Suchen
- Verwaltung

Ereignisse anzeigen ab Stufe:
0 1 2 3
JSPWiki v2.8.3-svn-19

Alexander der Große

Alexander III. lebte 356-323 v. Chr. und errichtete durch mehrere Feldzüge, die 336 begannen, ein riesiges Reich (Alexanderzug und Alexanderreich).

Person

+ **Beginn des Asienfeldzugs Alexanders gegen die Perser : 334 v. Chr.**

+ **Schlacht bei Issos : 333 v. Chr.**

- Persische Gegenoffensive in der Ägäis : **333 v. Chr.**

Unter der Führung des Memnon erobert die persische Flotte einige Inseln in der Ägäis, darunter Chios und Lesbos.

- Ptolemaios erobert die Cyrenaica : **322 v. Chr.**

Fig. 6. HermesWiki: article on Alexander the Great showing a map generated by the ontology. The most important events in the life of Alexander are geographically highlighted on the map.

used to enrich existing maps by visualizing aggregated views of the curriculum vitae of particular persons.

5. Conclusions

This article introduced the concept of the knowledge formalization continuum, which is a mental issue that helps knowledge engineers and domain specialists during the development process of intelligent systems. The knowledge formalization continuum is neither a physical model of knowledge nor a methodology for the knowledge formalization process. It rather combines alternative knowledge representations of the same subject into a continuum, ranging from free text to explicit logic rules. We also claimed that a semantic wiki is an appropriate tool to support knowledge engineering following the ideas of the knowledge formalization continuum. We demonstrated its application by the KnowWE semantic wiki and two case studies: the first was taken from the medical domain and considered the development and evaluation of the decision-support system CareMate; the second one demonstrated the use of the knowledge formalization continuum and KnowWE in the context of the HermesWiki platform—an e-learning project in ancient history.

The full power of the knowledge formalization continuum cannot be fully exploited with current state-of-the-art methods. In the future, automated methods, as we sketched in Section 2.1, will need further improvement and practical applicability. For example, adapted and effective NLP methods are necessary to declaratively and accurately transfer free text knowledge into more explicit forms. On the other hand, it becomes interesting to provide declarative methods that allow a flexible and tailored definition of explanations and visualizations of more explicit representations of knowledge.

References

Antoniou, G. and van Harmelen, F. (2003). Web ontology language: OWL, *in* S. Staab and R. Studer (Eds.), *Handbook on Ontologies in Information Systems*, Springer-Verlag, Berlin.

Baumeister, J. and Freiberg, M. (2010). Knowledge visualization for evaluation tasks, *Knowledge and Information Systems*, (in press).

Baumeister, J., Reutelshoefer, J. and Puppe, F. (2007). Markups for knowledge wikis, *SAAKM'07: Proceedings of the Semantic Authoring, Annotation and Knowledge Markup Workshop, Whistler, Canada*, pp. 7–14.

Baumeister, J., Reutelshoefer, J. and Puppe, F. (2010). KnowWE: A semantic wiki for knowledge engineering, *Applied Intelligence*, DOI: 10.1007/s10489-010-0224-5.

Baumeister, J., Seipel, D. and Puppe, F. (2004). Refactoring methods for knowledge bases, *in* E. Motta, N. Shadbolt, A. Stutt and N. Gibbins (Eds.), *EKAW'04: Engineering Knowledge in the Age of the Semantic Web: 14th International Conference*, Lecture Notes in Artificial Intelligence, Vol. 3257, Springer, Berlin, pp. 157–171.

Baumeister, J., Betz, C. and Wolber, M. (2008). The knowledge modeling environment d3web.KnowME, `http://d3web.sourceforge.net`.

Buffa, M., Gandon, F., Ereteo, G., Sander, P. and Faron, C. (2008). SweetWiki: A semantic wiki, *Web Semantics* **8**(1): 84–97.

Buitelaar, P., Cimiano, P. and Magnini, B. (2005). *Ontology Learning from Text: Methods, Evaluation and Applications*, Frontiers in Artificial Intelligence and Applications, Vol. 123, IOS Press, Amsterdam.

Card, S. (2003). Information visualization, *in* J.A. Jacko and A. Sears (Eds.), *The Human-Computer Interaction Handbook*, Lawrence Erlbaum Associates, Hillsdale, NJ, pp. 544–582.

Dale, R., Moisl, H. and Somers, H. (Eds.) (2000). *A Handbook of Natural Language Processing: Techniques and Applications for the Processing of Language as Text*, Marcel Dekker Inc, New York, NY.

Feldman, R. and Sanger, J. (2006). *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*, Cambridge University Press, Cambridge.

Fuchs, N.E., Kaljurand, K. and Kuhn, T. (2008). Attempto Controlled English for knowledge representation, *in* C. Baroglio, P.A. Bonatti, J. Małuszyński, M. Marchiori, A. Polleres and S. Schaffert (Eds.), *Reasoning Web, Fourth International Summer School 2008*, Lecture Notes in Computer Science, Vol. 5224, Springer, Berlin, pp. 104–124.

Geroimenko, V. and Chen, C. (Eds.) (2006). *Visualizing the Semantic Web*, 2nd Edn., Springer, Berlin.

Ghidini, C., Kump, B., Lindstaedt, S.N., Mahbub, N., Pammer, V., Rospocher, M. and Serafini, L. (2009). MoKi: The enterprise modelling wiki, *in* L. Aroyo, P. Traverso, F. Ciravegna, P. Cimiano, T. Heath, E. Hyvnen, R. Mizoguchi, E. Oren, M. Sabou and E. Paslaru Bontas Simperl (Eds.), *ESWC'09: The Semantic Web: Research and Applications*, Lecture Notes in Computer Science, Vol. 5554, Springer, Berlin, pp. 831–835.

Gil, Y. and Tallis, M. (1997). A script-based approach to modifying knowledge bases, *AAAI/IAAI'97: Proceedings of the 14th National Conference on Artificial Intelligence and the 9th Innovative Applications of Artificial Intelligence Conference, Menlo Park, CA, USA*, pp. 377–383.

Gruber, T. (2008). Collective knowledge systems: Where the Social Web meets the Semantic Web, *Web Semantics* **6**(1): 4–13.

Krötzsch, M., Vrandecić, D. and Völkel, M. (2006). Semantic MediaWiki, *in* I. Cruz, D. Allemang, L. Aroyo, S. Decker, P. Mika, C. Preist, D. Schwabe and M. Uschold (Eds.), *ISWC'06: Proceedings of the 5th International Semantic Web Conference*, Lecture Notes in Artifical Intelligence, Vol. 4273, Springer, Berlin, pp. 935–942.

Lidwell, W., Holden, K. and Butler, J. (2003). *Universal Principles of Design*, Rockport Publishers, Minnespolis, MN.

McGuinness, D.L. (2003). Ontologies come of age, *in* D. Fensel, J.A. Hendler, H. Lieberman and W. Wahlster (Eds.), *Spinning the Semantic Web*, MIT Press, Cambridge, MA, pp. 171–194.

Millard, D.E., Gibbins, N.M., Michaelides, D.T. and Weal, M.J. (2005). Mind the semantic gap, *HYPERTEXT '05: Proceedings of the 16th ACM Conference on Hypertext and Hypermedia, Salzburg, Austria*, pp. 54–62.

Nalepa, G.J. (2009). PlWiki—A generic semantic wiki architecture, *in* N.T. Nguyen, R. Kowalczyk and S.-M. Chen (Eds.), *ICCCI'09: Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems*, Lecture Notes in Computer Science, Vol. 5796, Springer, Berlin, pp. 345–356.

Oxford English Dictionary of Current English (2008). 3rd Edn., Oxford University Press, Oxford.

Peng, Y. and Reggia, J.A. (1990). *Abductive Inference Models for Diagnostic Problem-Solving*, Springer, Berlin.

Reiter, E. and Dale, R. (2000). *Building Natural Language Generation Systems*, Cambridge University Press, Cambridge.

Reutelshoefer, J., Baumeister, J. and Puppe, F. (2009). A data structure for the refactoring of multimodal knowledge, *KESE'09: 5th Workshop on Knowledge Engineering and Software Engineering, Paderborn, Germany*, pp. 33–45.

Reutelshoefer, J., Lemmerich, F., Baumeister, J., Wintjes, J. and Haas, L. (2010). Taking OWL to Athens—Semantic Web technology takes ancient Greek history to students, *in* L. Aroyo, G. Antoniou, E. Hyvnen, A. ten Teije, H. Stuckenschmidt, L. Cabral and T. Tudorache (Eds.), *ESWC'10: Proceedings of the 7th Extended Semantic Web Conference*, Lecture Notes in Computer Science, Vol. 6088, Springer, pp. 333–347.

Reutelshoefer, J., Lemmerich, F., Haupt, F. and Baumeister, J. (2009). An extensible semantic wiki architecture, *SemWiki'09: 4th Workshop on Semantic Wikis—The Semantic Wiki Web, Hersonissos, Greece,* pp. 155–169.

Roth-Berghofer, T.R. (2004). Explanations and case-based reasoning: Foundational issues, *in* P. Funk and P.A. Gonzlez-Calero (Eds.), *Advances in Case-Based Reasoning*, Lecture Notes in Computer Science, Vol. 3155, Springer-Verlag, Berlin, pp. 389–403.

Schaffert, S. (2006). IkeWiki: A semantic wiki for collaborative knowledge management, *STICA'06: 1st International Workshop on Semantic Technologies in Collaborative Applications, Manchester, UK*.

Schaffert, S., Bry, F., Baumeister, J. and Kiesel, M. (2008). Semantic wikis, *IEEE Software* **25**(4): 8–11.

Schaffert, S., Gruber, A. and Westenthaler, R. (2006). A semantic wiki for collaborative knowledge formation, *Proceedings of the SEMANTICS 2005 Conference, Vienna, Austria*.

Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., de Velde, W.V. and Wielinga, B. (2001). *Knowledge Engineering and Management—The CommonKADS Methodology*, 2nd Edn., MIT Press, Cambridge, MA.

Uschold, M. and Gruninger, M. (2004). Ontologies and semantics for seamless connectivity, *SIGMOD Record* **33**(4): 58–64.

Zacharias, V. (2007). Visualization of rule bases—The overall structure, *I-KNOW '07: Proceedings of the International Conference on Knowledge Management, Graz, Austria*.

Joachim Baumeister received his Ph.D. in 2004 from the University of Würzburg, Germany, where he gives lectures on the Semantic Web, algorithms, and knowledge-based systems. He has authored and co-authored more than 70 reviewed publications in the field of knowledge engineering and the Semantic Web. His main research interests are related to pragmatic knowledge engineering techniques, knowledge base evaluation, and industrial applications of knowledge systems. He has organized several national conferences and workshops and served as a program committee member of international conferences and workshops.

Jochen Reutelshoefer is a Ph.D. student at the University of Würzburg, Germany (since 2007). His research interest is in (pattern-based) knowledge formalization in semantic wikis. With his colleagues from the Department of Intelligent Systems, he works on the development of the KnowWE semantic wiki. He has authored and co-authored about 15 reviewed publications in the field of knowledge engineering and semantic wikis. He was a co-organizer of the national *FGWM* workshop on knowledge management in 2009 and of the International Workshop on Semantic Wikis *(SemWiki)* in 2010.

Frank Puppe is a full professor of artificial intelligence and applied informatics at the University of Würzburg, Germany. His research interests include all aspects of knowledge and information systems and their use in medical and technical domains. Towards this goal he has developed with colleagues several tool kits and frameworks for knowledge-based diagnosis, scheduling, subgroup analysis, multiagent simulation, case-based training, and human-computer-interaction. He is the author or a co-author of more than 150 publications and books.

Received: 12 April 2010
Revised: 22 November 2010