DERIS2009:

International Workshop on

Design, Evaluation and Refinement of Intelligent Systems

November 28, 2009, Kraków, Poland

Antoni Ligęza and Grzegorz J. Nalepa (Editors)

AGH University of Science and Technology, Kraków, Poland, 2009

The DERIS 2009 page is available online: http://home.agh.edu.pl/gjn/deris

Preface

Antoni Ligęza and Grzegorz J. Nalepa

Institute of Automatics
AGH University of Science and Technology
Kraków, Poland
ligeza@agh.edu.pl, gjn@agh.edu.pl

Evaluation, Verification, Validation and Refinement have been important issues from the very beginning of the applications of Intelligent systems. These issues were an important research area and engineering aspect in 80' and 90'. A number of conceptual approaches as well as practical tools was developed then.

With time, the focus of research in the design of intelligent systems moved away from these topics, towards knowledge representation and processing, the Semantic Web technologies, and a number of AI-inspired areas. However, recently a number of researches has realized that the lack of systematic methods and formal techniques for the design, evaluation and refinement is often an important reasons for limited applications of even mature intelligent systems. Therefore, there is a growing need to return to some of the basic issues in this field.

In fact today, the classic approach to the Evaluation, Verification, Validation and Refinement have to be assessed from the new perspectives. The practical design issues are of prime importance. The integration of Intelligent Systems with mainstream technologies and design approaches from Software Engineering is especially important. The quality issues need to be considered as early as possible during the design phase of the system.

One of the goals of the workshop was to rebuild the community interested in topics of Evaluation, Verification, Validation and Refinement, as well as attract new researchers to the field. The objective was to focus on the contributions in the above fields and to provide an environment for communicating different paradigms and approaches, thus hopefully stimulating future cooperation and synergistic activities.

Topics of interest were mainly located in the area of Evaluation, Verification, Validation and Refinement and include but are not limited to:

- Principles in knowledge systems and ontology design
- Detecting and handling inconsistencies and other anomalies within knowledge bases
- Fundamentals and formal methods for verification of AI systems
- Fundamentals and formal methods and techniques of validity assessment of AI systems, AI principles, and intelligent behavior in general
- Special approaches to verify and/or validate certain kinds of AI systems: rule-based, case-based
- Special approaches or tools to evaluate systems of a particular application field
- Knowledge base refinement by using the results of evaluation

- Development and evaluation of ontologies
- Maintenance and evolution of knowledge systems and ontologies
- Methods for the evaluation of distributed knowledge bases
- Evaluation of semi-formal knowledge bases
- Problems in system certification
- Ontology and knowledge capture
- Evaluation of Semantic Web applications
- Formal methods in Verification and Evaluation of Intelligent Systems

During the workshop 8 papers have been presented, including 6 regular papers and 2 short papers. Each submission was reviewed by 2 programme committee members.

The organizers would like to thank all who contributed to the success of the workshop. We thank all authors for submitting papers to the workshop, and we thank the members of the program committee for reviewing and collaboratively discussing the submissions. For the submission and reviewing process we used the EasyChair system, for which the organizers would like to thank Andrei Voronkov, the developer of the system.

Antoni Ligęza
Grzegorz J. Nalepa

Kraków, November 28, 2009

Workshop Organization

DERIS2009:
International Workshop on
Design, Evaluation and Refinement of Intelligent Systems
was held as a one-day event
on November 28, 2009 in Kraków, Poland.

Workshop Chairs and Organizers

Antoni Ligęza, AGH UST, Kraków, Poland
Grzegorz J. Nalepa, AGH UST, Kraków, Poland

Programme Committee

Martin Atzmüller, University of Würzburg, Germany
Joachim Baumeister, University Würzburg, Germany
Klaus P. Jantke, Fraunhofer IDMT, Germany
Rainer Knauf, TU Ilmenau, Germany
Antoni Ligęza, AGH UST Kraków, Poland
Grzegorz J. Nalepa, AGH UST, Kraków, Poland
Mieczysław L. Owoc, Wrocław University of Economics, Poland
Silive Spreeuwenberg, LibRT B.V., Amsterdam, The Netherlands
Derek H. Sleeman, University of Aberdeen, United Kingdom

Table of Contents

Regular Papers.

Short Papers.

KNOWTA: Wiki-Enabled Social Tagging for Collaborative Knowledge and Experience Management

Martin Atzmueller[1], Fabian Haupt[1], Stephanie Beer[2], and Frank Puppe[1]

[1] University of Würzburg,
Department of Computer Science,
Am Hubland, 97074 Würzburg, Germany
{atzmueller,fhaupt,puppe}@informatik.uni-wuerzburg.de
[2] University-Hospital Würzburg,
Department of Gastroenterology,
Josef-Schneider-Str. 2, 97070 Würzburg, Germany
beer_s@klinik.uni-wuerzburg.de

Abstract. Social tagging systems are a useful tool for collaborative knowledge management: They enable the flexible collection, annotation, organization, and distribution of resources and information. A combination with wiki systems then provides powerful but easy to use approaches for a broad range of applications. This paper describes a wiki-enabled social tagging system for knowledge and experience management, and presents its application using a case-study in the medical domain.

1 Introduction

Both Web 2.0 and semantic technologies enable the creation of powerful tools for knowledge management. While social tagging systems [1] provide versatile approaches for the distributed collection, annotation, organization, and distribution of resources that are especially easy to use, wiki-based systems allow the collaboration of a community of users in a simple way.

Therefore, combining both approaches seems to be an attractive solution since they complement each other well: The wiki-based system provides the tools for accessing and managing the content, while the tagging application takes care of the annotation, and categorization of the resources. In this way, a community-driven knowledge management approach can be well supported. Additionally, by embedding semantic web [2] technology, e.g., for accessing the tagged resources or modeling the tag relations, a flexible and powerful user experience can be implemented.

This paper presents an approach combining tagging and wiki-editing in a social system for knowledge and experience management. The KNOWTA (*Know*ledge *Ta*gging) system provides advanced tagging functionality including recommendation options and utilizes a semantic wiki component for transparent access to the provided knowledge. KNOWTA is well suited for handling multi-modal knowledge, e.g., containing text and images. Both can be captured by the wiki engine and are transparently tagged (or semantically annotated) using the tagging and wiki functionality, respectively. We describe the system and its capabilities in detail and provide a real-world case study in the medical domain.

The rest of the paper is structured as follows: Section 2 outlines the basic issues of social tagging and the utilized semantic technologies. After that, Section 3 provides an overview on the proposed approach and discusses its elements in detail. Next, Section 4 presents a case study in the medical domain. Finally, Section 5 concludes with a summary and outlines interesting options for future work.

2 Social Tagging Systems

For organizing information and knowledge, the notion of *tagging* has recently received much attention: It has become a useful way for the collection, annotation, organization, and distribution of resources by users. The tags assigned to specific resources are used for navigation, locating the resources and for serendipitous browsing. Due to its ease of use, tagging systems provide an immediate benefit for users: They allow a transparent access to various resources. Additionally, tagging can help for communicating interesting nuggets of information [3].

In this paper we focus on the handling and management of multi-modal resources. We support both textual and image resources that are transparently integrated using (semantic) wiki technology. Therefore, we extend a conventional wiki system by providing extended means for the direct integration of images similar to the *Flickr* [4] system. Users can directly upload images as well as link other resources to the system. In the following, we shortly introduce the main features of social tagging systems, before we describe the KNOWTA system in the next section.

Social tagging systems are based on resources, e.g., bookmarks or images, users and tags. Thus, an entry e in such a system can be regarded as a 3-tuple composed of a resource $r \in R$, a user $u \in U$ and a set of tags $t_1, \ldots t_n \in T$, where R specifies the set of resources, U specifies the set of users and T specifies the set of valid tags.

In general, the set of tags T is unbound and can be extended by the users as needed. This is the case for folksonomies [5], for which arbitrary tags can be assigned. However, there are also other possibilities, especially for *closed communities*: In this case, the set of users U is fixed, and new users cannot join the community on their own. In such circumstances, often a restricted vocabulary is more appropriate. Then, the set T is prespecified by a *super-user* and cannot be freely extended. Usually, only a selected group of users, e.g., domain specialists, are enabled to modify this set of tags. In a knowledge-acquisition step, it is usually generated according to the specific domain and the targeted closed-community of users.

3 Wiki-enabled Social Tagging

This paper proposes the social tagging system KNOWTA for collaborative knowledge management. As a successor to the KNIZR(*Kn*nowledge Orga*niz*e*r*) system [6], KNOWTA includes sophisticated semantic components for collaborative knowledge management. This section first provides an overview on the KNOWTA system. After that, the tagging and semantic capabilities are discussed in detail.

3.1 Overview on KNOWTA

KNOWTA builts upon established Wiki technology based on the *JSPWiki* system [3]; it supports a variety of resource types that can be embedded into a wiki page, in addition to the rich textual edit actions. Specifically, the system enables the management of image data with corresponding texts for knowledge management. Then, image and textual data, i.e., multi-modal information, can both be tagged and semantically annotated using the functionality of the social tagging system and/or the wiki-features, respectively. As an optional extension, the system can be configured for closed-communities such that controlled vocabularies for the tags, i.e., the tag ontology of the system can be defined. This is especially useful for specific application projects with a fixed specific set of contributing users.

In addition to the JSPWiki component, KNOWTA also utilizes the *semantic core* component of the *KnowWE* [7] system. The latter allows for powerful semantic browsing options for accessing the stored resources and content. For semantic browsing and querying users, resources, tags, links, and the content itself can be considered. In addition to including complex handcrafted ontologies, KNOWTA can also include SKOS [8], i.e., the *Simple Knowledge Organization System*, a W3C standard for knowledge organization for the web and especially the semantic web. SKOS is based on (simple) web-standards such as RDF, and provides a formal language for constructing, for example, thesauri or concept taxonomies. At its core it is quite simple, extensible and maintainable. Figure 1 shows an exemplary screenshot of the KNOWTA system.

For general resource management, KNOWTA offers the following functionality:

- Resources can be created by providing a title, an image, a set of tags and/or a description (optional), and a set of tags can be assigned to them based on a describing sentence. This is implemented using the wiki engine and the functionality of the tagging engine. Additionally, as described below, a set of tags can be semi-automatically assigned using recommended selections.
- Resources are defined for certain groups of users: Using an access control schema of JSPWiki, certain user groups and corresponding rights can be defined.
- Resources can be commented on by the users: This enables collaborative knowledge and experience management and can support the collaborative discussion of specific issues regarding the given resource.
- Resources and their associated (multi-modal) information can be semantically annotated: This is accomplished using the features of the (extended) semantic core of the KnowWE system. By adding semantic annotations to the textual information, links and relations between resources can be easily established.

Connections between resources/tags are thus either established using (explicit) links between the resources, implicit relations/links given by the a matching set of tags of different resources, or by considering the relations between the tags defined in a domain ontology (for closed communities). The general knowledge and experience management functions are supported by the wiki-enabled information handling – using unstructured, semi-structured, and structured information. Since the knowledge can be collaboratively extended, the system provides a knowledge-rich collaborative environment for a broad range of applications.

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

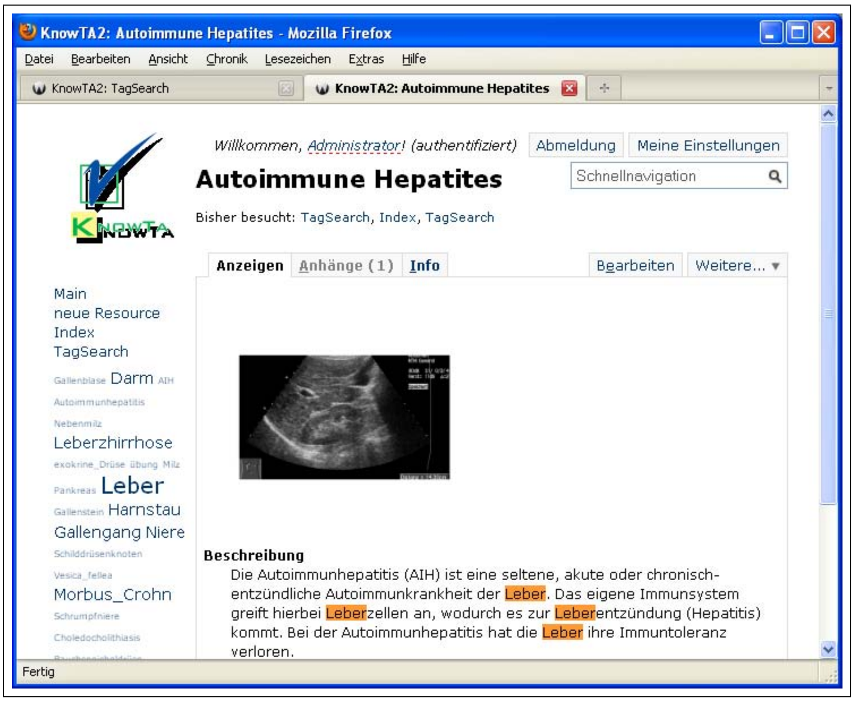

Fig. 1. KNOWTA: Exemplary resource/image with annotation and tags (in german).

3.2 KNOWTA– Tagging, Recommendation, Browsing

For the tag assignments, KNOWTA utilizes a tag recommendation system that takes a short sentence as input, and then proposes a set of suitable tags. This feature is especially important for the targeted medical domain, since the users often do not have a lot of time when entering a new resource, a situation which is quite typical, for example, for medical doctors. Figure 2 shows the simple interface for creating a new resource. Figure 3 shows an example of the medical domain (liver) demonstrating the inline tag edit functionality.

For the tag recommendation, text mining techniques are applied. There are several options for tag recommenders (see [5] for a survey); for occurrence-based tag recommendation the following strategy is applied: The tag assignment information of all users is taken into account for computing the co-occurrence counts of all tags, weighted by their importance, similar to a *tf/idf* schema [9]. After that, the input text (given by a short sentence, and/or a set of tags) is preprocessed by tokenization, removing stop-words, and stemming. Then, a set of tokens with the highest co-occurrence counts (and weights) is retrieved and proposed to the user. The user can then apply these tags and use them for tagging the current article. This schema allows providing a controlled vocabulary, since the 'free tagging' functionality can be disabled in favor of the tag rec-

Fig. 2. KNOWTA: Interface for creating a new resource (screenshot in german): The user can upload a new resource/image, provide a default tag/title (Seitenname), enter a description (Beschreibung) which is used for recommending tags, and enter a list of tags (Tags).

ommendation schema using default occurrence values. Then, only the given tags in the vocabulary can be proposed and used for tagging.

After a set of resources has been defined, a key point are efficient and effective search and browsing capabilities for these resources. According to Shneiderman [10] an effective user interface should implement features for getting an overview first, then zoom on some details, finally getting the details on demand. In this way, the so-called *visual information seeking mantra* is implemented. KNOWTA implements these features by both a powerful search facility and also by using a tag cloud visualization. Additionally, for each resource a set of related tags/resources is shown such that the navigation between them is easily implemented. Then, the user can apply a dynamic query refinement for obtaining a refined set of resources within a more detailed cluster of information.

3.3 KNOWTA– Semantic Representation

As mentioned before, some of the features of KNOWTA are embedded into the semantic core functionality of KnowWE which was extended as needed. The semantic core of KnowWE relies on Sesame[4] as RDF-Triple storage and OWLIM[5] as reasoning engine.

KnowWE (without considering any extensions) is a basic semantic wiki system featuring an annotation syntax and the possibility of SPARQL-query embedding ([7]). It provides several ways of expanding the ontology of the wiki: Ontologies can be introduced to the system, for example, by embedding owl-source directly into a wikipage within `<owlextension>...</owlextension` tags. For more extensive extensions of the ontology it is advisable to use the provided ontology manager interface to

[4] `http://www.openrdf.org/`
[5] `http://www.ontotext.com/owlim/`

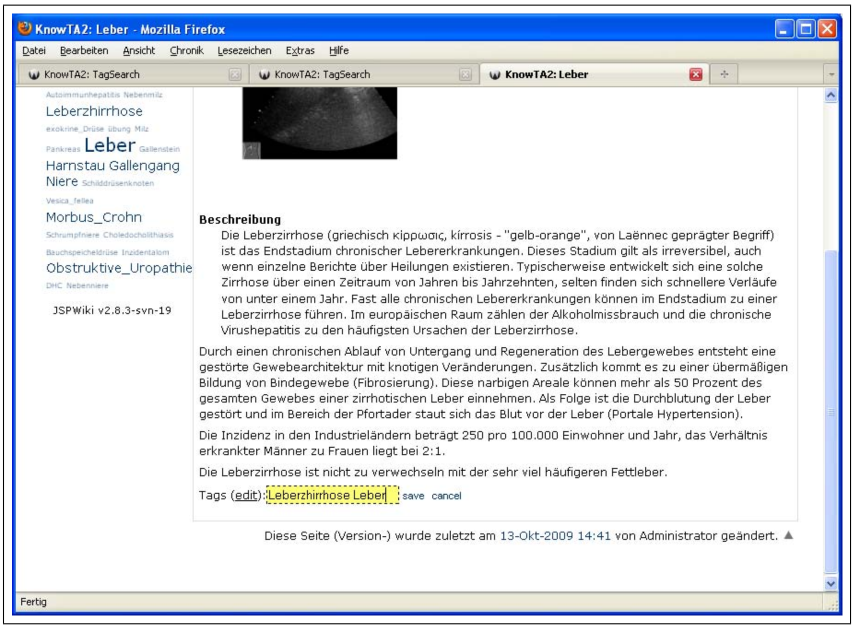

Fig. 3. KNOWTA: Exemplary resource/image with tag-editing mode (in german).

upload owl-files directly into the wikisystem. Another common operation is the introduction of new annotation properties into the system, which is facilitated by the shorthand `<properties>...</properties>`. Properties can be listed within those tags anywhere in the wiki and are then recognized as viable annotation properties for any resources.

As the tagging of a resource is a specialized case of a semantic annotation, the tagging functionality is merely a convenient way to introduce this annotation into the backend ontology. Adding a tag to a resource can be done either by using a AJAX-based tag edit panel or by editing the content of the XML-Tag `<tags>...</tags>` directly. The former saves the tags into the XML tags and the wikisource as well. After updating a wikipage, KnowWE parses the page into a specialized datastructure, which in turn provides an interface to create owl statements from the wikiarticle. Tags are rendered into multiple N-ary statements as introduced in [7].

The following owl snippet, for example, is the result of tagging the page `Main` with the tag `Demo`, i.e., the tag is given in the wiki source by `<tags>Demo</tags>`, see Figure 4 for an example.

```
<rdf:Description rdf:about="lns#Main..Node9_XML_content..MainhasTagDemo">
        <rdfs:isDefinedBy rdf:nodeID="node41"/>
        <rdfs:seeAlso rdf:nodeID="node41"/>
        <rdf:predicate rdf:resource="ns#hasTag"/>
        <rdf:object rdf:resource="lns#Demo"/>
        <rdf:subject rdf:resource="lns#Main"/>
</rdf:Description>
```

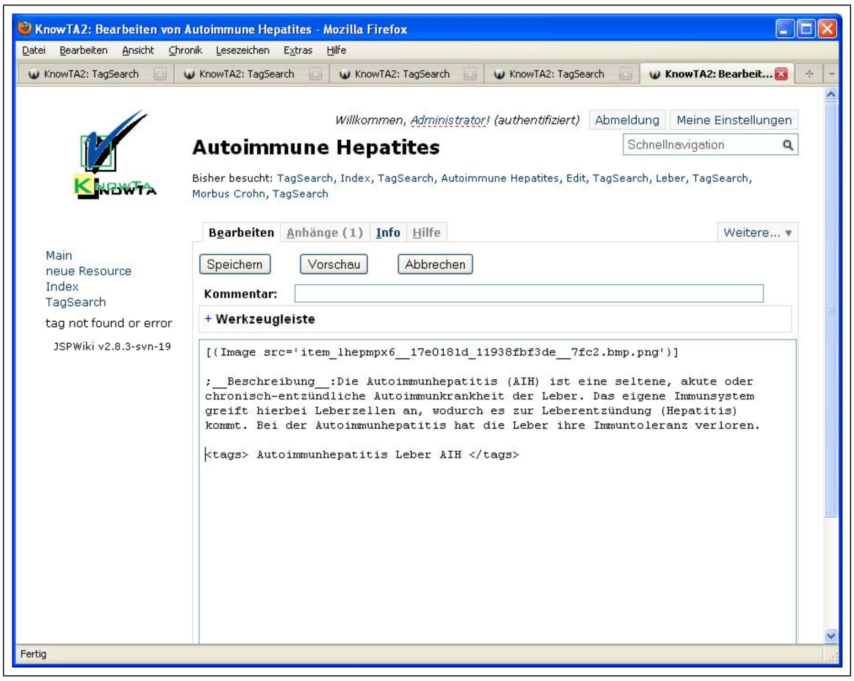

Fig. 4. KNOWTA: Exemplary article with associated tags (in german).

The namespaces used are `lns` which is the namespace of the local wiki installation and `ns` which stands for the KnowWE upper ontology namespace included in in KnowWE. `node41` is a reference to the node in the specialized datastructure in KnowWE. This allows, among others, a precise localization of the annotation, or in this case the tag element within the wiki. The name of the node is automatically generated by convention of the semantic engine.

Due to the fact, that the tags are stored in the ontology of the wiki, it is possible to integrate arbitrary SPARQL-queries for tags or pages tagged with specific tags everywhere in the wiki, for example, within any page or even the menu structure. To get a list of all pages, for example, with the assigned tag `Demo` the following SPARQL-query can be embedded into a wiki page. It results in a list of pages with the `Demo` type.

```
<sparql>
SELECT ?q
WHERE {
    ?t rdf:object lns:Demo .
    ?t rdf:predicate lns:hasTag .
    ?t rdfs:isDefinedBy ?o .
    ?o ns:hasTopic ?q
}
<sparql>
```

To get an overview of all tags used in a wiki, a tag cloud can be rendered anywhere in a wikipage (see figure 5). As expected, the size of the rendered tags corresponds to the tag frequency within the whole wiki. Clicking on a tag searches for all pages that are tagged with the selected tag and allows a tag-based navigation through the wikispace.

Fig. 5. An exemplary tag cloud showing the tags of a wikispace.

4 Case Study

The case study presents an application of the KNOWTA system for extended knowledge and experience management. The application domain is the medical domain of sonography, for which sonographic images are collected, annotated and commented in order to serve as instructive examples for typical but also exceptional features of certain disorders. In this way, effective tutoring and discussion between the examiners can be initiated. So far, about 500 images have been obtained, which comprehensively cover the problem domain and provide an experience base for supporting medical training and consultation by knowledge and experience management. Additionally, the system provides the capability to semi-automatically export the collected data to the *CaseTrain* training system [11].

4.1 Knowledge and Experience Management

The context of the application is given by the intelligent documentation and consultation system SONOCONSULT system [12, 13] – a medical system for sonography. The system is in routine use in several hospitals, for example, in the DRK-hospital in Berlin/Köpenick and in the University Hospital of Würzburg.

For a sonography (ultrasound) examination, the examiners need to closely inspect the ultrasound images, for example, see Figure 6 in order to document the correct findings. Since this process is highly subjective and also significantly dependent on the experience of the examiner, there are often discrepancies between beginners and more senior examiners concerning the correct findings, c.f., [14]. Therefore, a system for the collection and annotation of *interesting*, i.e., exceptional or typical images for certain medical phenomena with associated textual descriptions provides for a powerful tool.

In this context, KNOWTA thus provides an ideal framework for implementing an image pool for tutoring, targeted training and general knowledge management. Special cases of ultrasound phenomena can be uploaded to the system. Figure 7 shows an example for a set of liver-related phenomena.

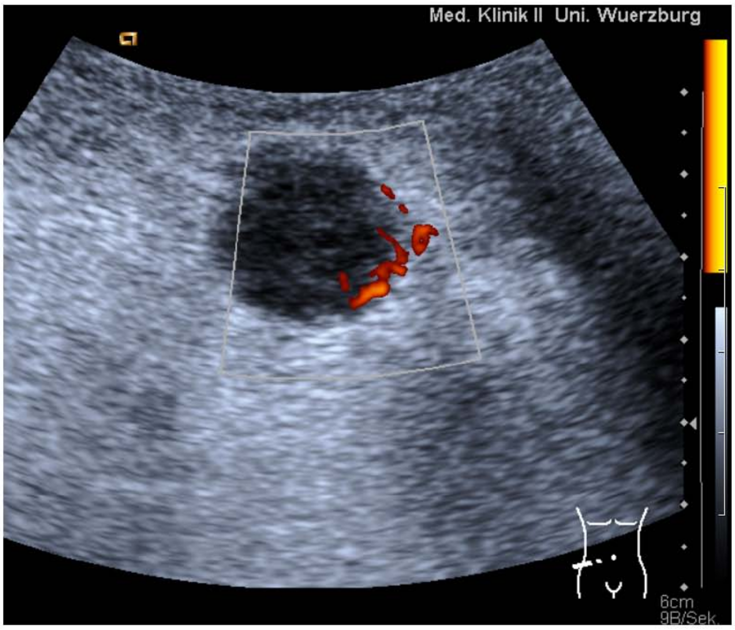

Fig. 6. Exemplary image from an ultrasound examination.

The resources can be easily described, tagged and annotated, and junior examiners can search for specific (difficult) situations using the tag cloud and/or the search functionality. Additionally, hints can be easily communicated using the annotation and/or comment function of the system. Furthermore, using the SKOS features of the system the terminology of the different organ systems and diseases can be directly utilized for an effective retrieval of resources.

4.2 Knowledge Capture for Tutoring

After the resources have been collected in the KNOWTA system, they are readily available for further processing. In order to apply the system for knowledge capture for designing and building tutoring cases, a (semi-)automatic export to the CaseTrain system is provided. Furthermore, an automatic option is given by exporting all images with their associated 'main' tag (the page title) and storing the complete list as 'long menu' questions for all images. In this case a (potentially randomly selected) subset of tagged images is obtained; for tutoring, the user needs to select a specific tag from the 'long menu' that describes the image best.

So, the images, the annotations and tags can be utilized for creating new quizzes for tutoring and teaching purposes. Additionally, the generated cases can be potentially linked to KNOWTA articles, such that CaseTrain users can get background information about the case. An exemplary CaseTrain screenshot is given in Figure 8.

Fig. 7. Exemplary tag-search for liver diagnoses (in german).

5 Conclusion

Social tagging systems can provide powerful and intuitive solutions to various knowledge management problems. In this paper, we have introduced the KNOWTA system for tutoring and general knowledge management. We discussed the different components and features of the system, and we have described its application in a real-world medical application scenarios.

So far, we have implemented a full prototype of the system: Since the results are quite promising, we aim to perform a comprehensive evaluation of the system in the sketched application scenario. Additionally, we want to utilize as much background knowledge as possible in order to exploit the semantic features at their full level. We also plan to enhance the tag recommendation feature by discovering certain user subgroups and applying their features for a better recommendation, knowledge capture, and user experience using data mining [15], text mining [16] and information extraction [17] techniques.

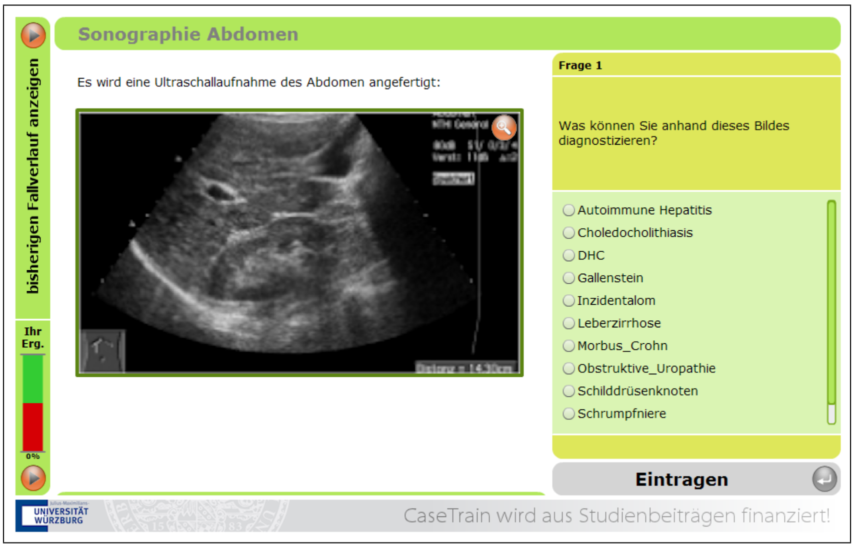

Fig. 8. Exemplary CaseTrain case.

Acknowledgements

This work has been partially supported by the German Research Council (DFG) under grant Pu 129/8-2.

References

1. Golder, S., Huberman, B.A.: Usage Patterns of Collaborative Tagging Systems. Journal of Information Science **32**(2) (April 2006) 198–208
2. Antoniou, G., Harmelen, F.v.: A Semantic Web Primer. MIT Press, Cambridge, Mass. (2004)
3. Hammond, T., Hannay, T., Lund, B., Scott, J.: Social Bookmarking Tools (I) - A General Review. D-Lib Magazine **11**(4) (April 2005)
4. Mislove, A., Koppula, H.S., Gummadi, K.P., Druschel, P., Bhattacharjee, B.: Growth of the flickr social network. In: Proceedings of the 1st ACM SIGCOMM Workshop on Social Networks (WOSN'08). (August 2008)
5. Jäschke, R., Marinho, L.B., Hotho, A., Schmidt-Thieme, L., Stumme, G.: Tag recommendations in folksonomies. In: Proc. PKDD '07. Volume 4702 of Lecture Notes in Computer Science., Berlin, Heidelberg, Springer (2007) 506–514
6. Atzmueller, M., Hörnlein, A.: Exploiting the Power of Social Tagging Systems: A Semantic Flickr Approach for Tutoring and Knowledge Management. In: Proc. 1st European Workshop on Design, Evaluation and Refinement of Intelligent Systems, Erfurt (2008)
7. Reutelshoefer, J., Haupt, F., Lemmerich, F., Baumeister, J.: An extensible semantic wiki architecturre. In: SemWiki'09: 4th Workshop on Semantic Wikis. (2009)
8. Brickley, D., Miles, A.: SKOS core vocabulary specification 2005-11-02. W3C working draft, W3C (November 2005) http://www.w3.org/TR/swbp-skos-core-spec.

9. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. 1st edn. Addison Wesley, Harlow (1999)
10. Shneiderman, B.: Direct Manipulation for Comprehensible, Predictable and Controllable User Interfaces. In: Proc. 2n Intl. Conf. on Intelligent User Interfaces. (1997) 33–39
11. Hoernlein, A., Ifland, M., Kluegl, P., Puppe, F.: CaseTrain: Design and Evaluation of a Case-Based Training-System and its Comprehensive Application in University (Konzeption und Evaluation eines fallbasierten Trainingssystems im universitätsweiten Einsatz (CaseTrain)). GMS Med Inform Biom Epidemiol **5**(1) (2009)
12. Huettig, M., Buscher, G., Menzel, T., Scheppach, W., Puppe, F., Buscher, H.P.: A Diagnostic Expert System for Structured Reports, Quality Assessment, and Training of Residents in Sonography. Medizinische Klinik **99**(3) (2004) 117–122
13. Puppe, F., Atzmueller, M., Buscher, G., Huettig, M., Lührs, H., Buscher, H.P.: Application and Evaluation of a Medical Knowledge-System in Sonography (SonoConsult). In: Proc. 18th European Conference on Artificial Intelligence (ECAI 20008). (2008) 683–687
14. Atzmueller, M., Puppe, F., Buscher, H.P.: Profiling Examiners using Intelligent Subgroup Mining. In: Proc. 10th Intl. Workshop on Intelligent Data Analysis in Medicine and Pharmacology (IDAMAP-2005), Aberdeen, Scotland (2005) 46–51
15. Atzmueller, M., Puppe, F.: A Knowledge-Intensive Approach for Semi-Automatic Causal Subgroup Discovery. In: Knowledge Discovery Enhanced with Semantic and Social Information. Springer Verlag (2009)
16. Atzmueller, M., Nalepa, G.J.: A Textual Subgroup Mining Approach for Rapid ARD+ Model Capture. In: 22nd International Florida Artificial Intelligence Research Society Conference (FLAIRS). (2009) 414–419
17. Atzmueller, M., Kluegl, P., Puppe, F.: Rule-Based Information Extraction for Structured Data Acquisition using TextMarker. In: Proc. LWA 2008 (Knowledge Discovery and Machine Learning Track), University of Wuerzburg (2008)

Engineering Expressive Knowledge with Semantic Wikis. ⋆

Joachim Baumeister[1], Grzegorz J. Nalepa[2]

[1] Institute for Computer Science
University of Würzburg, Am Hubland, 97074 Würzburg, Germany
joba@uni-wuerzburg.de
[2] Institute of Automatics,
AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Kraków, Poland
gjn@agh.edu.pl

Abstract. Semantic wikis are successfully used in various application domains. Such systems combine the flexible and agile authoring process with strong semantics of ontologies. The current state–of–the–art of systems, however, is diverse in the sense of having a common ground. Especially, the expressiveness of the knowledge representation of semantic wikis undergoes continuous improvement. In the paper, two semantic wiki implementations are discussed, that are both extending semantic wiki implementations by strong problem-solving knowledge. We compare their approaches and we aim to condense the fundamental characteristics of a strong problem-solving wiki.

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*. 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, a 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. 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.

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

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 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 wiki technology has gained importance with respect to the collaborative knowledge acquisition and engineering. The development of *semantic wikis,* such as IkeWiki [1], Semantic MediaWiki [2], and SweetWiki [3], allowed to use the Semantic Web methods and tools on top of the existing content-centered wiki solutions.

Existing semantic wikis allow for an 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, we introduce a categorization of semantic wiki functions in order to simplify the comparison of system approaches and implementations. The categorization will concentrate on the features required during knowledge engineering activities and will omit other important aspects such as data storage and scalability of the implementations. We compare the two semantic wikis *PlWiki* and *KnowWE* according to the introduced scheme.

2 Categorization of Expressive Semantic Wikis

The semantic wiki community tracks the recent developments of semantic wiki features within a matrix[3]. For example, the matrix lists features such as *editing paradigm*, *annotation mechanism*, *programming language* and *license.* For systems —extended by expressive knowledge formalizations— we see that this categorization is too broad to capture the their special capabilities. For this reason, we propose a supplementary set of feature categories in the following, which we use to describe the wiki implementations in the latter of the paper.

We see that semantic wikis are not used in a unified application context, but are often designed to match to a more specific application area. Thus, the particular systems cannot be compared in a linear order, but show advantages and disadvantages for specific applications. A new matrix can help to simplify the selection of a system for a given work task.

A. Targeted applications

Often, the systems are designed with a specific application context in mind. Possible values are:

1. Community-based ontology engineering
2. Encyclopedia-like application, e.g., Wikipedia enhancement

[3] http://semanticweb.org/wiki/Semantic_Wiki_State_Of_The_Art

3. E-learning
4. Special-purpose system for knowledge engineering tasks (closed communities)
5. Other (please describe your context)

B. Underlying knowledge representation

The semantic wiki represents the developed knowledge in a particular format. This sub-area specifies the type of structure, how the atomic concepts of the knowledge base are represented, it names the used knowledge representation language, and discusses if/how additional sources of knowledge are connected.

1. **Subject granularity:** What is the level of detail of the atomic concepts represented in the knowledge base. This also corresponds to the main structuring paradigm of the system; i.e., most wikis use the single wiki pages as the underlying structuring paradigm, where the particular concepts are represented by single wiki pages. Possible values are:
   (a) One concept/property for each wiki page, (b) Multiple concepts/properties for one wiki page, (c) Other—please specify
2. **Knowledge representation language:** This sub-area names the expressivity of the underlying knowledge representation. Possible values are:
   (a) RDF(S), (b) OWL, (c) Other/combination—please specify
3. **Additional knowledge sources:** In addition to standard ontological knowledge, often a special type of additional knowledge is attached, for example, rules and domain models. Possible values are:
   (a) SWRL rules, (b) Prolog rules, (c) Model-based knowledge, (d) Text in controlled language, (e) Other—please specify

C. UI for knowledge capture and sharing

In practical applications the user interface for the development and use of the represented knowledge is of prime importance. Here, most of the current semantic wiki implementations strongly differ and show emphasis on their chosen application context.

1. **Knowledge editing paradigm:** How is the knowledge acquired and maintained within the wiki system? Possible values ares:
   (a) Inline text markup: The knowledge entered and maintained in combination with the usual textual edit pane of the wiki. The wiki offers special purpose markup to define the knowledge entities.
   (b) Semantic forms/visual editors: Ontological concepts and more expressive knowledge is defined using customized forms and graphical editors.
   (c) Multiple (combination of above approaches)
   (d) Other—please specify
2. **Semantic search/knowledge retrieval:** One prominent application of knowledge use is the integration of semantic search facilities. Of interest is the applied approach of integrating the search into the wiki as well as the language to formulate the queries.

(a) Used query language: SPARQL, Datalog-like queries, other—please specify.
   (b) Query integration: Special-purpose forms, queries embedded into the wiki article, interview questionnaires, other—please specify.
   (c) Further capabilities of knowledge use: Generated knowledge-based interviews, Prolog queries, other—please specify.
3. **Semantic navigation:** The semantic annotations of the wiki articles allows for an improved navigation through the contents of the system. Some systems provide a simple extension of semantic navigation by extending the standard link structure, whereas other implementations (additionally) facilitate semantic search by generated fact sheets interlinking connected articles.
   (a) Extended links within wiki article
   (b) Generation of fact sheets
   (c) Other—please specify

D. Connectivity

This sub-area specifies the capabilities of the wiki with respect to the import and export of knowledge from and to external sources.

1. **Import facilities:** Is it possible to import external knowledge sources, and if yes then what type of sources are supported? Possible values are:
   (a) RFS(S)/OWL, (b) SWRL, (c) RIF, (d) Proprietary knowledge sources—please specify, (e) none
2. **Export facilities:** Is it possible to export the knowledge base of the wiki to an external storage?
   (a) RDF(S)/OWL, (b) SWRL, (c) RIF, (d) Proprietary knowledge sources—please specify, (e) none

E. Extensibility

Due to the fast development of new technologies and application areas, the extensibility of existing wiki systems by new features is very important. This sub-area captures the type of extensibility and lists a number of already existing extensions/modifications of the wiki.

1. Extension mechanism:
   (a) Plug-in mechanism, (b) Code-in mechanism, (c) none, (d) other—please specify
2. Existing extensions/modifications: Please list running extensions of the wiki system.

With the proposed categorization we try to capture the conceptual properties of semantic wikis. Please note that we tried to not include technical details of the implementation of the wiki systems, such as data storage (databases vs. text) and the programming language.

3 The Semantic Wiki KnowWE

We first summarize the features of KnowWE according to the introduced feature matrix and then we introduce the system in more detail.

3.1 KnowWE in the Feature Matrix

We summarize the KnowWE features according to the introduced feature matrix:

A. Targeted applications	
4) Special-purpose system for knowledge engineering tasks	
B. Underlying knowledge representation	
1) Subject granularity	a) One concept/property for each wiki page
2) Knowledge repr. lang.	c) Combination (OWL and Special-purpose problem-solving knowledge)
3) Add. knowledge sources	e) Other: heuristic rules, decision trees, fault models
C. UI for knowledge capture and sharing	
1) Editing paradigm:	a) Inline text markup, b) visual editors (for d3web extension)
2) Search/Retrieval	a) SPARQL, b) Queries embedded into the wiki article, interview questionnaires, c) Further capabilities: Generated knowledge-based interviews
3) Semantic navigation	a) Extended links within wiki article, b) Generation of fact sheets, c) Other: Knowledge-based interview
D. Connectivity	
1) Import facilities	a) OWL, d) Proprietary: text (KnOffice)
2) Export facilities	a) OWL, d) Proprietary: text (KnOffice), xml (d3web)
E. Extensibility	
1) Extension mechanism	(a) Plug-in mechanism
2) Extensions mods	d3web plugin, semantic tagging plugin Hermes plugin

3.2 KnowWE in a Nutshell

In this section, we introduce the general features of KnowWE by using a simple example application for diagnosing car faults. The basic idea is, that possible causes for a car fault —the solutions of the problem— are represented by corresponding wiki articles. The wiki contains, for instance, articles about *empty battery*, *clogged air filter*, and *bad ignition timing*.

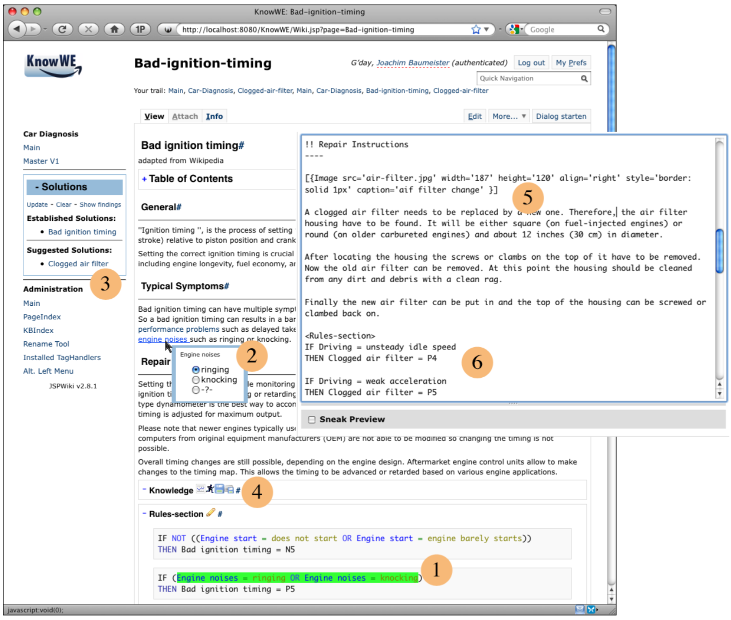

Fig. 1. A wiki article describing the solution *bad ignition timing* in the context of a car diagnosis application (1–4), and the edit panel of another wiki page describing the solution *Clogged air filter* (5–6).

Inline Editing of Text and Knowledge In Figure 1, a page of the wiki is shown, describing the solution *bad ignition timing*. Besides standard text describing the problem in more detail, also explicit problem-solving knowledge is included on the page. In Figure 1-(1) two heuristic rules of the rule base are displayed, that describe some derivation knowledge of the solution. The first rule states, that the solution *Bad ignition timing* will receive a negative score, if the user enters for the symptom *engine start* that it neither *does not start* nor *barely starts*. The second rule states a positive score, if the *engine noise* was observed by the user as *ringing* or *knocking*. Besides the representation of rules, we also allow for the inclusion of model-based knowledge and decision trees, see [4] for more details on the implemented knowledge connectors.

We see, that the derivation knowledge for a solution is locally defined and maintained together with the corresponding article of the solution; see the right top of Figure 1 for an example edit panel of the wiki, where a rule base (Figure 1-6) is edited inline. This allows for a simplified update of informal (e.g., text) *and* explicit knowledge (e.g., rules) about one entity.

Although, the wiki is mainly used as a tool for knowledge engineering, it also provides interfaces for interactive problem-solving. We give an example of the problem-solving process in the following: Some parts of the text are related to concepts of the knowledge base, and thus have a meaning for the problem-solving process. Specific semantic annotations relate these text parts with the concepts. In the view mode of the wiki the user is able to click on the annotated text and can enter findings based on the corresponding concept. We call this approach *inline answers* for problem-solving in wikis. The text phrase "engine noises" within the text was annotated by the corresponding concept *Engine noises* available in the knowledge base, see Figure 1-(2). In the given example, the value *knocking* for the concept *Engine noises* was entered by the user. In the solutions pane of the wiki — Figure 1-(3) — we see that the solution *Bad ignition timing* is derived with a high score, whereas the alternative solution *Clogged air filter* was also derived and is considered as a possible solution. Both solutions were derived on the basis of this finding and previously entered findings. By clicking on the solution *Clogged air filter* in the solutions pane, we quickly can navigate to the wiki article describing the corresponding article.

Alternatively, the user is able to download an executable version of the knowledge base by clicking the download button, see Figure 1-(4). This way, the knowledge bases can be developed using the wiki and later are exported to an external application when required.

Semantic Annotations In KnowWE, semantic annotations are defined inline with the wiki text. The markup for those annotations was inspired by the syntax of Semantic MediaWiki [2], and ontological concepts can be simply linked by the definition of ontological properties. The general syntax of the markup connects a text phrase of the wiki text with a concept using an ontological property.

```
[Bad ignition timing is a technical problem
 <=> subClassOf:: TechnicalProblem] that can be solved ...
```

In the example shown above, the text phrase "Bad ignition ... problem" is annotated, stating, that the concept represented by this article is a `subClassOf` the concept `TechnicalProblem`. The annotation itself states, that the annotated text phrase documents/justifies the given relation.

By this type of annotations many useful ontological relations can be defined inline the wiki text. All annotations are represented in the application ontology and can be queried using a SPARQL. Queries are embedded into the wiki text, and the results of the queries are shown in the view mode of the article.

3.3 Applications of KnowWE

KnowWE is typically used together with the d3web plugin for building knowledge-based applications. The system is currently used in a number of (industrial and academic) projects, ranging from simple recommender systems to complex decision-support systems for technical and medical devices. For example, KnowWE provides a technical platform to support a scientific community in the biological domain in the context of the *BIOLOG Wissen*[4] project. BIOLOG Wissen serves as a web-based application for the collaborative construction and use of a decision-support system for landscape diversity. It aims to integrate knowledge on causal dependencies of stakeholders, relevant statistical data, and multimedia content. We refer the interested reader to [5,6] for more details. In another recent project, KnowWE is extended by diagnostic workflow knowledge in the context of the CliWE project[5]. By this extension, the wiki will be used to collaboratively develop clinical guidelines, that are integrated as compiled knowledge bases into next-generation medical devices. A first prototype of this extension is reported in Hatko et al. [7].

4 The Semantic Wiki PlWiki

The PlWiki [8] features are briefly introduced first. Then, the basic architectural assumptions are given, and specific knowledge representation aspects presented.

4.1 PlWiki in the Feature Matrix

The features of PlWiki according to the introduced feature matrix are:

A. Targeted applications	
4) Special-purpose system for knowledge engineering tasks	
B. Underlying knowledge representation	
1) Subject granularity	b) Multiple concepts/properties for one wiki page
2) Knowledge repr. lang.	c) Other/combination (Prolog low-level, OWL higher-level)
3) Add. knowledge sources	b) Prolog rules
C. UI for knowledge capture and sharing	
1) Editing paradigm:	a) Inline text markup, d) Other: Prolog editing support
2) Search/Retrieval	a) Prolog queries, b) Queries embedded into the wiki article, c) Further capabilities: Prolog predicates for knowledge processing
3) Semantic navigation	a) Extended links within wiki article, b) Generation of fact sheets
D. Connectivity	
1) Import facilities	a) OWL (planned) d) Proprietary: SMW knowledge format, Prolog, XTT
2) Export facilities	a) OWL (planned) d) Proprietary: Prolog
E. Extensibility	
1) Extension mechanism	(a) Plug-in mechanism, (b) Code-in mechanism
2) Extensions mods	Custom Prolog extensions, SWI Semantic Layer

[4] BIOLOG is funded by the German Federal Ministry of Education and Research from 2007-2009 (final funding phase).

[5] CliWE (Clinical Wiki Environment) is funded by Drägerwerk, Germany and runs from 2009-2012.

4.2 PlWiki in a Nutshell

System Architecture The main objective of the PlWiki design is to deliver a generic and flexible knowledge engineering solution [8]. 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. 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 [9].

DokuWiki provides a flexible plugin system, providing five kinds of plugins (see `www.dokuwiki.org/devel:plugins`). The current version of PlWiki implements both the *Syntax* and *Renderer* plugin 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.

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

```
<pl> capital(germany,berlin). country(germany). country(poland). </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 specify a given *scope* of the query (in terms of 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. An example of editing session with PlWiki can be observed in Fig. 2.

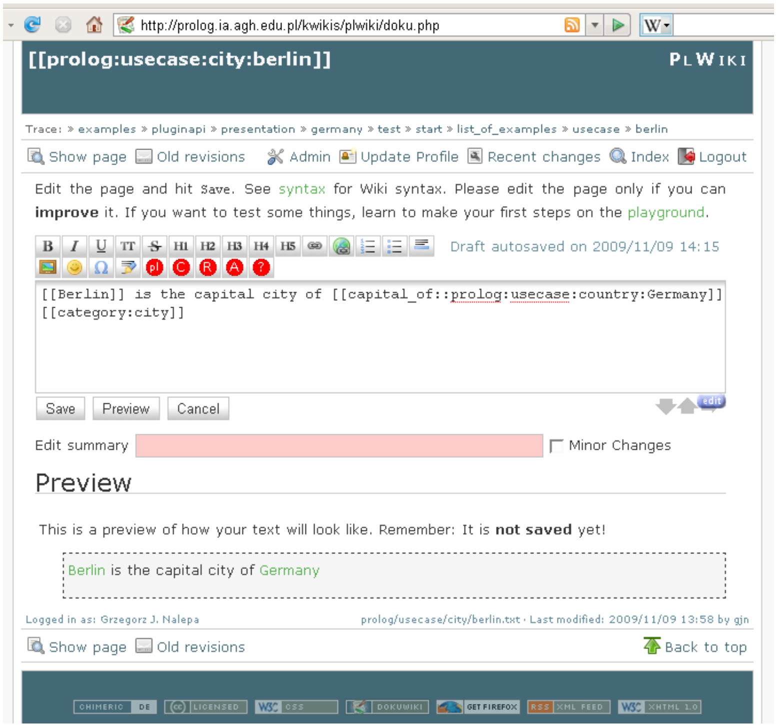

Fig. 2. PlWiki editing session

Semantic Representation Layer Besides the generic Prolog-based knowledge representation features based on pure Prolog clauses, typical semantic wiki features are supported. Semantic Media Wiki (SMW) [10], 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 clauses could also be introduced by using the PlWiki `<pl></pl>` tags. Examples are shown below, with the SMW syntax given first, and the corresponding Prolog representation after it.

```
Berlin is the capital city of [[capital_of::Germany]] [[category:city]]
  wiki_category('City','Berlin').
  wiki_property(capital_of,subject_page_name,'Germany').
Germany is a country in central Europe.
[[category:country]] [[location:=Central Europe]]
  wiki_category('Country,'Germany').
  wiki_attribut(page_uri,location,'Central_Europe').
```

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 Prolog goal:

```
{{#ask: [[category:city]] [[capital of::Germany}}
  wiki_category('Cities',Page),
  wiki_property(capital_of,Page,'Germany'), 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('Berlin',capital_of,'Germany')`.

Using wiki knowledge it is possible to define rules, e.g.: "Nordic country is a country with `location` set to `Northern Europe`" is in Prolog:

```
<pl cache="true"> nordic_country(X) :-
                    wiki_category(X,'country'),
                    wiki_attribute(X,'location','Northern Europe'). </pl>
```

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

4.3 Applications of PlWiki

PlWiki is in an experimental development phase. Current applications include special knowledge engineering tasks, including basic rule-based reasoning tasks in the wiki, and teaching knowledge engineering classes. Future applications are planned, including dedicated knowledge intensive closed community portals. System development will focus on flexible user interfaces supporting complex knowledge representation features.

5 Conclusions

Semantic wikis are successful examples of semantic applications and are widely used in academia and industry. From the technological viewpoint, however, we currently see no common baseline for the expressiveness and functionality of semantic wikis. In contrast, the available and active semantic wiki implementations strongly differ in their characteristics. In this paper, we introduced a catalog of functions with which systems can be compared and evaluated. We emphasized the functionality of wikis with respect to knowledge engineering activities, thus excluding certainly important aspects such as editing paradigm, data storage, and scalability. The semantic wiki implementations KnowWE and PlWiki were presented according to the introduced catalog.

This work is an initial attempt in this research direction. In the future, we are planning to undertake an extensive study not only incorporating two systems, but comparing all available and active semantic wiki implementations within a feature matrix. The studies would we also aimed at providing benchmark case studies implemented using different wiki implementations which would allow for a synthetic comparison and feature analysis.

References

1. Schaffert, S., Gruber, A., Westenthaler, R.: A semantic wiki for collaborative knowledge formation. In: Proceedings of SEMANTICS 2005 Conference, Trauner Verlag (2006)
2. 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
3. Buffa, M., Gandon, F., Ereteo, G., Sander, P., Faron, C.: SweetWiki: A semantic wiki. Web Semantics **8**(1) (2008) 84–97
4. 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
5. Nadrowski, K., Baumeister, J., Wolters, V.: LaDy: Knowledge Wiki zur kollaborativen und wissensbasierten Entscheidungshilfe zu Umweltveränderung und Biodiversität. Naturschutz und Biologische Vielfalt **60** (2008) 171–176
6. Baumeister, J., Reutelshoefer, J., Haupt, F., Nadrowski, K.: Capture and refactoring in knowledge wikis – coping with the knowledge soup. In: SCOOP'08: Proceedings of 2nd Workshop on Scientific Communities of Practice, Bremen, Germany (2008)
7. Hatko, R., Baumeister, J., Puppe, F.: Diaflux: Diagnostic flows in wikis. In: FGWM'09: Proceedings of German Workshop of Knowledge and Experience Management (at LWA'09). (2009)
8. Nalepa, G.J.: Plwiki – a generic semantic wiki architecture. In Ngoc Thanh Nguyen, Ryszard Kowalczyk, S.M.C., ed.: 1st International Conference on Computational Collective Intelligence - Semantic Web, Social Networks & Multiagent Systems. Volume LNAI 5796., Springer (2009) 345–356
9. Nalepa, G.J., Ligęza, A.: Hekate methodology, hybrid engineering of intelligent systems. International Journal of Applied Mathematics and Computer Science (2009) accepted for publication.
10. Krötzsch, M., Vrandecić, D., Völkel, M., Haller, H., Studer, R.: Semantic Wikipedia. Web Semantics **5**(4) (2007) 251–261

Experiences in Applying Artificial Intelligence within SIARAS Project

Sławomir Nowaczyk

Katedra Automatyki,
Akademia Górniczo-Hutnicza,
Slawomir.Nowaczyk@agh.edu.pl

Abstract. In this paper we present our experience in designing and building knowledge base for the SIARAS project. This project aimed to produce a tool, called Skill Server, supporting engineers during reconfiguration of manufacturing systems. We mainly focus on the process of developing knowledge representation used by Skill Server and discuss some lessons we have learned about applying Artificial Intelligence concepts in practical applications within high-tech industry. We discuss several types of knowledge that Skill Server needs to have access to, and how different KR solutions can be integrated together in a coherent system.

1 Introduction

In this paper we present our experience in designing and building knowledge base for the SIARAS project, focusing especially on how the approaches we used evolved as we gained more and more insight into the actual requirements of the system.

SIARAS is an acronym of an EU-funded (FP6 - 017146) STREP-project "Skill-Based Inspection and Assembly for Reconfigurable Automation Systems," running in the years 2006–2008. Its general aim was to support end users and engineers of manufacturing systems, including robotic ones, and to make production engineering easier (and thus cheaper) in several common circumstances.

The primary goal of the project was to build an intelligent system, provisionally called *Skill Server*, that would be capable of supporting automatic and semi-automatic reconfiguration of manufacturing processes in response to changing requirements. The main issue during the design phase was to merge two, somewhat opposed, views on the reconfiguration process: the top-down, AI-based approach and the bottom-up, engineering one.

The top-down approach describes the reconfiguration as a (re)planning problem. We are given a new task, usually expressed as a goal condition, possibly being a modification of an earlier, correct one. Given a set of skills available in the system, understood as a description of the operations that might be performed by the devices available to the user, we are to find such a sequence of operations that will ensure that the task is correctly executed, i.e. to find a plan that achieves the goal.

In the bottom-up approach, the Skill Server is used only for reconfiguration of an existing, correct, properly modelled production line. The system is not expected to propose novel solutions, nor to search for alternative ways of implementing the process. In particular, one should expect a detailed description of the task: what is produced and how (i.e. what are the steps of the process). Moreover, for each step, it should be clear how does it contribute to the goal. On the other side, available devices must be described in terms of operations they are able to perform (skills) and conditions under which they can operate. Skill Server needs to map task into skills and parametrise them appropriately.

It is rather obvious that the top-down AI approach is both computationally infeasible and impossible to model sufficiently well, while the bottom-up reparametrisation approach lacks generality and risks ending up as a database of *previously used* parameter settings for a number of devices in a number of scenarios. The main issue with this approach is guaranteeing scalability and extendibility to new domains or to new kinds of devices. There is a risk of limiting the approach to the previously considered cases and very similar ones only, thus precluding a more open-ended solution.

Taking this into account, we have settled for a layered approach, with reconfiguration level at the bottom and (re)planning level on top of it.

2 Knowledge Representation

Initially, we have identified several types of knowledge Skill Server will use: skills, devices, tasks, workpieces and environment. Most of them can be specified on, at least, two levels of abstraction: simplified, generic descriptions (like a universal "pickup skill") and instantiated ones (the operation of gripper G_1 picking the windshield W_1 in factory F, at time point t_n and position p_m). Throughout the project, however, we have been continuously investigating possibilities of introducing additional, intermediate levels of abstraction in between.

Nevertheless, in addition to the symbolic knowledge, there exist a number of domain-specific or device-specific procedures for calculating various aspects (e.g. trajectory planner, device calibration and reparametrisation procedures, etc.) which, in many contexts, should also be treated as knowledge. Despite several attempts, however, we have not managed to find an acceptable and useful way to put them into any kind of even semi-formal framework. This would be a very interesting research project in itself, but our experience shows that this knowledge is simply too diverse to formalise in the timeframe we had. Therefore, Skill Server considers such procedures as, essentially, black boxes.

The overall structure of the Skill Server is shown in Figures 1 and 2. They depict, from two different points of view, how the "core" Skill Server is integrated with various knowledge, reasoning and user interface modules and what kind of information sources it has at its disposal.

In effect, we have devoted most of our efforts to designing symbolic knowledge representation in a way that would be intuitive and useful in practice to our industrial partners (who, in general, have no background in AI) and, at the

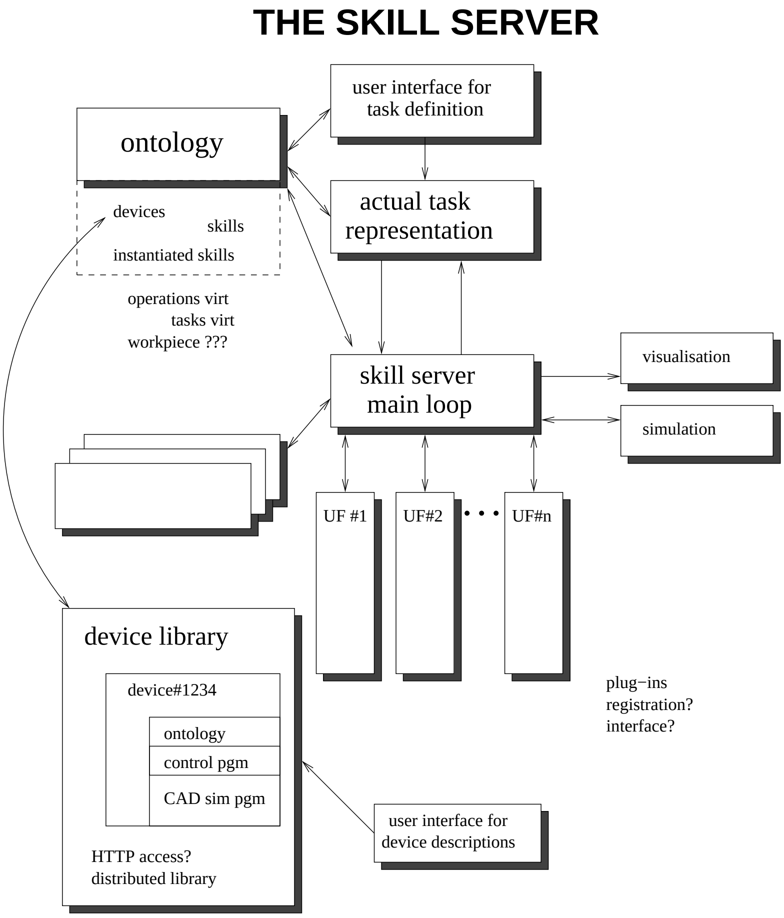

Fig. 1. Skill Server structure

same time, allow Skill Server to perform the reasoning tasks required. It turned out to be a surprisingly difficult task to agree upon a knowledge representation formalism within the consortium, however. In fact, while we (from the academia side) were already aware that there are costs associated with formal approaches to KR, we have not anticipated how difficult it will be to convince industrial partners that the benefits outweigh those costs.

In principle, the initial response was that any representation more advanced than *attribute-value* pairs was deemed as unnecessarily complex and still not expressive enough. Basically, the expectations seemed to be that we should use attribute-value pairs wherever appropriate, and full expressiveness of programming languages everywhere else. Finally, the consortium have eventually managed to agree upon using *Ontology* as a default knowledge representation formalism, provided we create a simple and intuitive one, not use existing.

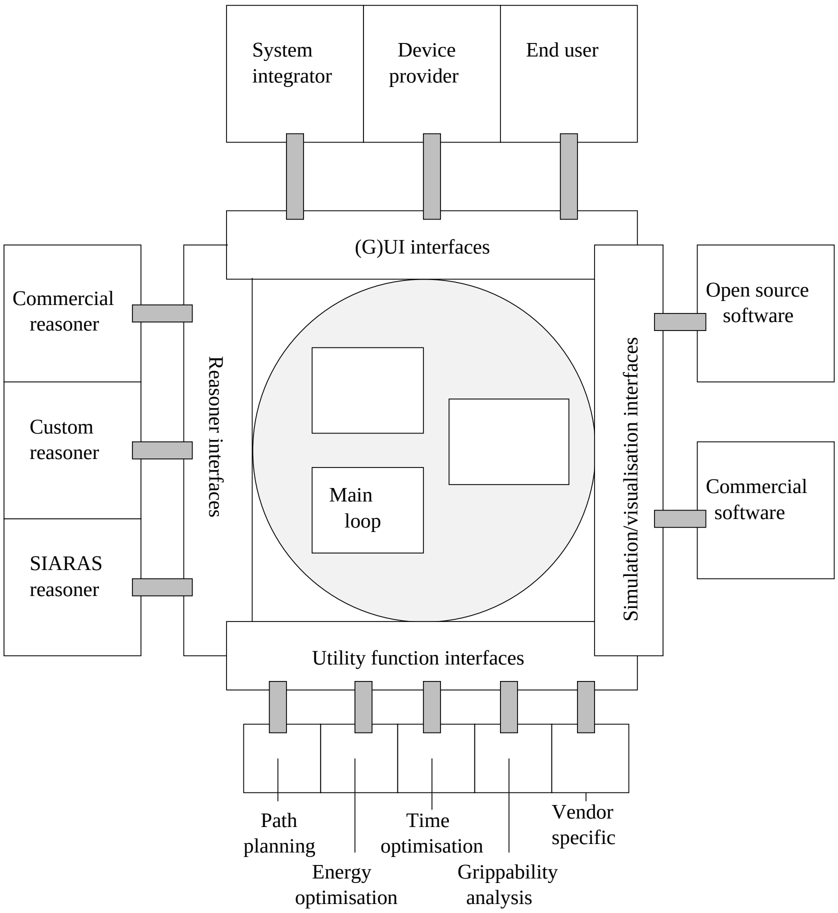

Fig. 2. Skill Server structure

One of the issues that caused the most discussions among the project partners was the contents and structure of the ontology. The latter question is mostly technical and amounts to fixing a representation which would make typical uses of the knowledge stored in the ontology as easy as possible. Still, during discussions, those two aspects seemed to be confused very often.

However, the former issue, i.e. what should the contents of the ontology be, was of paramount importance. There was little doubt that it should contain knowledge about skills and about devices providing them. What was not so obvious was the status of tasks: whether they should be present in the ontology as a separate entities.

There are strong arguments both for and against such solution. As tasks are one of the major concepts present in the basic idea of the Skill Server, they need to be included in the ontology (which serves as Skill Server's primary vocabulary) as well. Without their existence, no reasoning about them would be possible and thus the Skill Server would not be able to perform some of its basic functions.

However, on the most basic level, tasks necessarily have to correspond directly, on the one-to-one basis, with the skills. Otherwise there would be no possibility for the skill server to perform any kind of matching between them. Therefore, it is probably unwise to exactly duplicate the hierarchy of skills with a corresponding hierarchy of tasks. Moreover, the "lifetime" of tasks is also significantly different than the rest of information within the Ontology: skills and devices are almost eternal, in a sense that they can be input into the knowledge base once and are useful for everybody any time in the future. Tasks, on the other hand, are defined within a very specific context and usually only useful within a particular factory shop and for a limited time.

The ontology contains knowledge about (abstract) skills and about devices. This is an area for which ontology is well-suited, so it is both easy and efficient to specify that, for example, a concept of "vacuum gripper" is a subconcept of "gripper", which in turn is a subconcept of "device". In a similar manner, a "vacuum-pickup" skill is a subconcept of "pickup" skill. Even though the expressive power of an ontology goes much further, we did in fact get most mileage out of it by treating it as little more than a glorified taxonomy. This part was considered sufficiently easy and sufficiently useful by all partners within a consortium to make it universally accepted. In fact, in some regards the ontology proved *too successful*, and there has been significant pressure to put virtually *all* knowledge there, even the kinds for which it is clearly inappropriate.

A useful feature of the ontology was that it allowed us to specify properties of each skill and device, with natural inheritance rules. Thus we were able to associate "mass" property with the concept of "device" and "number of fingers" with "finger gripper", to specify that "mass" can be expressed in "grams" or "pounds", and that "max image resolution" makes sense only for cameras, and not for robots. What we did not manage to do is to find a way to express the *purpose* of those properties in a way that would be accessible to the non-symbolic parts of knowledge, such as calibration procedure or specialised domain-dependent algorithms. However, this has more to do with how unstructured this procedural knowledge turned out to be and less with the ontology itself.

Another issue we have been debating many times within the consortium but never managed to reach a satisfactory conclusion was the concept of *compound devices*. For example, when talking about robots, it often appears useful to specify that a *robot* uses a *gripper* to perform some *skill*. It only seems natural to express that robot-with-a-gripper can perform more (or rather different) actions than robot alone. However, doing it properly turned out to be outside the representational power of an ontology, at least in the form we had it in the project.

3 Ontology Structure

We have decided to center knowledge representation around the concepts of devices (physical objects provided by their manufacturers) and skills (operations that can be performed). Task descriptions exist only during problem solving sessions, as dynamic structures, specific to a particular case. They can be seen as (arguably, quite complex) combinations of skills and parameters and therefore there is no need to have them explicit in the vocabulary.

The static part of the knowledge is represented in an ontology: a data structure storing all the necessary relations between the terms used. While ontologies are often used for classification purposes, in our case the classification is done when objects (skills or devices) are introduced in the structure. The main use of the ontology is to allow reasoning about skills matching particular tasks and about devices being suitable for particular operations, as well as to standardise the nomenclature used and the relationships of different concepts.

For the prototype of Skill Server, we have chosen the open source tool Protégé for ontology creation and manipulation, together with reasoners such as Racer[1], Fact++[2], Algernon[3] and Pellet[4]. If Skill Server is ever to enter production stage, a specialised user interface will need to be developed, since general purpose tools we such as those are not appropriate for its intended end users. They worked fine within the SIARAS project, however.

The ontology contains three hierarchies: Devices, Skills and Properties. The Device hierarchy specifies what types of devices do we intend Skill Server to reason about. We have provided device manufacturers with a rudimentary interface for introducing their products: they specify where in the hierarchy should a particular device reside and, depending on that, they are asked to fill in values of appropriate properties. Those devices are, at least conceptually, leaves (instances) of the Device hierarchy. However, since we expect no centralised repository of all devices to be available and assume that data will be organised in a distributed manner (for example, downloadable from device manufacturers' web pages), we are actually storing all the devices in an SQL database.

The second hierarchy is the Skill hierarchy, which mainly aims at providing a common vocabulary between the device manufacturers (who specify which skills does a device offer) and the end users or systems integrators (who specify what effect do they want to achieve, or which task do they want performed). The Skill hierarchy is supposed to be a common ground where the capabilities offered by device manufacturers can be mapped into the requirements of the users. It performed satisfactorily during the project, including the demonstration stage, but it has by now became clear that we are missing a common understanding of an appropriate abstraction level for skills. For example, there was much discussion on whether there should be a "drill" skill, or if it should rather be modelled as a sequence of "start drill rotation", "move", "stop drill rotation" operations. In the end we have gone with the former approach, but we are not convinced this is the best solution and would rather have a setup where different abstraction levels could be used as appropriate.

There has been much debate about the need for complex (or composite) skills, but we do not think it is appropriate to represent them directly in an ontology. It would make more sense to provide a more expressive way of building them, using a language like Finite State Machines or Sequential Function Charts. The skills in the ontology should be simple enough that it is possible to reason about their constraints and effects of their application.

Finally, the Properties hierarchy forms a link between Devices and Skills. It is, in a sense, the first – simplest – way of specifying the dependencies and instantiating parameters (for example, that a particular gripper has a certain maximum weight limit).

Throughout the project we have kept our initial assumption that all the hierarchies will be defined by the SIARAS consortium and remain fixed, as the meaning of all the concepts used needs to be encoded into the reasoning mechanisms of the Skill Server. However, it has become apparent that this assumption is a very constraining one, so it would be good to investigate ways to lift it.

4 Knowledge Representation outside Ontology

The major part of knowledge representation we have decided to store outside the ontology are the tasks. Tasks can be thought of as *generalisations* of skills along (at least) two axes: first, they are time-ordered (or partially parallelised) sequences of skills (or rather of skill applications): for example, a "relocate" task can be seen as a sequence of "pickup", "move" and "putdown" skills. Second, it makes sense to have hierarchy of tasks, with "windshield fitting" task decomposing into "find windshield", "position windshield" and "fix windshield" subtasks.

In addition, tasks constitute means of achieving a particular goal, and Skill Server needs to have a representation of that goal, in particular to reason about rationale for each operation and about motivations why a particular device and particular parameters were chosen. At the very least it requires a set of criteria which distinguish acceptable execution of this task from unacceptable ones, and possibly ways to *compare* two solutions and determine when one of them is better than the other.

The bottom level of tasks hierarchy are *operations*, which are a kind of associations of skills, devices, workpieces, factory floor positions and time-line constraints. Where tasks are especially useful is on a higher level, when defining a concrete production process that will be the subject of reconfiguration. Therefore the ontology does not need them, or putting it differently, the only tasks that are available in the ontology are those that skills refer to.

We have chosen Sequential Function Charts[5] to represent tasks, since it allows us to specify temporal ordering of operations in an intuitive and yet flexible way. We have used the open source tool JGrafchart for our prototype.

Finally, we have implemented the core reasoning in Python programming language, "gluing" together the knowledge from ontology, SFCs and device-dependent procedures provided in the form of plugins.

In our approach the ontology is used for reasoning about skills matching particular tasks (after some initial re/parametrisation) and devices offering those skills under certain conditions. A pure ontology may be used for retrieval, matching and simple classification, while other forms of reasoning, like planning, optimisations, consistency checks, etc., need to be done by reasoners, either general-purpose ones, or specialised for the task. The generic tools that have been used by in the project (Racer, Fact++, Algernon and Pellet) differ in their reasoning power and efficiency, being able to handle either a restricted Description Logic language [6] (like OWL-DL offered by Protégé) in an efficient manner [7], or a full OWL [8] representation, but using exponential search algorithms. The user has the possibility of choosing a different reasoner depending on the question asked, thus achieving flexibility and adaptability of the reasoning part of the system.

We have also developed tools for storing and retrieving knowledge in appropriate data structures, so that on one hand the ontology can be easily extended by the system providers, while on the other hand it may benefit from distributedness, letting some parts be completed and stored at the device manufacturer's site. Yet another set of requirements is put on the reasoning process by the list of optimisation tasks that may be requested by the user. Due to their computational complexity, and to their specificity to particular devices, they cannot be implemented in a general-purpose manner but rather require their specific reasoning blocks fitting the structure of the server.

5 Related Work

The research on knowledge representation has been extensively documented, both in general textbooks on artificial intelligence and in numerous books devoted solely to this domain. One of the recent ones, and a very good overview of the field, is by Brachman and Levesque [9].

The work that originated discussions over semantic web and, in particular, on ontologies, has an extensive library of published documents available for example at the W3Consortium's Semantic Web site [10]. In particular, the specifications of the most popular KR formalisms, like OWL [8] or DAML-OIL [11], together with available tools for using those formalisms, can be found there.

Production planning is usually considered in AI to be a part of the automatic planning domain. However, besides the classical manufacturability analysis, reported for example in [12], there is in principle no documented research on using ontologies in automated production planning. However, there is an extensive research aimed at supporting the engineering activities in production design by providing modelling languages and tools allowing formal, automatic analysis of the discussed process. Quite naturally, most of those formalisms and tools are heavily domain-dependent, with a small number of exceptions explicitly stating goal of being general-purpose tools. We may name here the *Sensor Modelling Language*, or Sensor ML for short, offering a rich sensor ontology (see `http://www.sensorml.org` for an extensive documentation). A dual enterprise, not exactly fitting our point of view either, is the *unified robot modelling language*, (URML), from the University of Karlsruhe, as URML does not provide representation facilities for the dynamic aspects of robot performance.

Finally, an important attempt to formalise the language for speaking about production processes has been done at NIST which created the Process Specification Language [13]. The language, and some of the associated tools, are serving as a reference point for the ontology developed within SIARAS.

6 Conclusions

In this paper we discuss the knowledge representation we use in the EU-funded project SIARAS, and present the most interesting points where real-life considerations clashed with our initial assumptions. We hope this overview will be helpful to other people designing knowledge bases for industry applications and will allow them to avoid some of the pitfalls we have encountered.

The main point of our discussion was the use of Ontology and what are the benefits and costs associated with representing knowledge in such structured way. We have discussed situations where ontology turned out to be a rather cumbersome tool, and also those where it proved to be very helpful. We wish we were able to offer some kind of evaluation of the approach we have chosen and of the representation we have ended up with, but unfortunately we do not have anything more concrete than our own reflections.

References

1. Haarslev, V., Möller, R.: Racer: A core inference engine for the semantic web (2002)
2. Horrocks, I.: FaCT and iFaCT. In: Proceedings of the Description Logics Workshop, DL'99. (1999)
3. Stoica, F., Pah, I.: Intelligent agents in ontology-based applications. In: 12th WSEAS International Conference on Computers. (2008) 274–279
4. Sirin, E., Parsia, B.: Pellet: An OWL DL reasoner. In: Description Logics. Volume 104 of CEUR Workshop Proceedings. (2004)
5. Fernández, J.L., Sanz, R., Domonte, E.P., Alonso, C.: Using hierarchical binary petri nets to build robust mobile robot applications: Robograph. In: International Conference on Robotics and Automation. (2008) 1372–1377
6. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F., eds.: The Description Logic Handbook. Cambridge University Press (2003)
7. Buchheit, M., Donini, F.M., Schaerf, A.: Decidable reasoning in terminological knowledge representation systems. Journal of Artificial Intelligence Research **1**(1) (1993) 109–138
8. W3C: Semantic web (2003)
9. Brachman, R.J., Levesque, H.J.: Knowledge Representation and Reasoning. Morgan Kaufmann (2004)
10. W3C: Semantic web (2001)
11. Fensel, D., Horrocks, I., van Harmelen, F., McGuinness, D.L., Patel-Schneider, P.F.: Oil: An ontology infrastructure for the semantic web. IEEE Intelligent Systems **16**(2) (2001)
12. Ghallab, M., Nau, D., Traverso, P.: Automated Planning, Theory and Practice. Morgan Kaufmann (2004)
13. Schlenoff, C., Gruninger, M., Tissot, F., Valois, J., Lubell, J., Lee, J.: The process specification language (PSL): Overview and version 1.0 specification. Technical report, National Institute of Standards and Technology (NIST) (2000)

Evaluation Needs of Webble Technologies in an E-Learning Laboratory Case Study

Jun Fujima, Imke Hoppe & Klaus P. Jantke

Fraunhofer IDMT, Children's Media Dept.,
Hirschlachufer 7, 99084 Erfurt, Germany
{jun.fujima|imke.hoppe|klaus.jantke}@idmt.fraunhofer.de

Abstract. Webble Technology is an advanced current form of Meme Media on the Web. The authors use Webbles for the implementation of Web-based interactive laboratories. There arises a particular question for the perception of added values which result from peculiarities of Meme Media technologies. This should not be confused with the question for the laboratories' usability. The focus of the present investigations is on the perception and, perhaps, appreciation of implementing ideas of memetics by different groups of users.

1 Introduction

When new technologies enter an application domain, technology providers are always assuming the invention's beauty and success. However, the proof of the pudding is the eating of the pudding.

The authors of the present paper are enganged in a comprehensive endeavor of introducing Meme Media technologies [1] into Web-based applications aiming at learning support at school. The technology of choice is "Webble".

The acronym Webble [http://www.meme.hokudai.ac.jp/WebbleWorldPortal/] abbreviates Web Pebbles, where "Pebble" is short for "Pad Enhanced Building Block Lifelike Entity". In this descriptional phrase, the term "Pad" again is short for "IntelligentPad" according to [1].

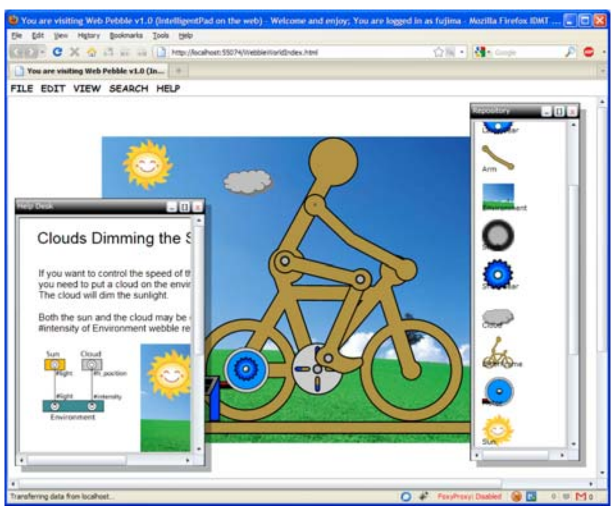

Fig. 1. Screenshot of the Webble-Based Solar Biker

One might ruoghly understand IntelligentPad as a middleware having a number of quite desirable features. Those who are more ambitious understand the IntelligentPad approach as a way to implement Meme-like building blocks intended to enable knowledge representation and evolution. The concept "Meme", being intended to resemble the words memory and gene, was coined by Richard Dawkins [2]. Yuzuru Tanaka took up the challenge to carry over Richard Dawkins' ideas to software technology [1]. Susan Blackmore is providing a general perspective at the reach of those ideas and approaches [3].

The authors are using Webble technologies for the implementation of a series of interactive laboratories of which the so-called Solar Biker is a prototypical example. The key question is to what extent the quality of the new technology is accepted.

2 The Solar Biker Laboratory

The authors' e-learning Solar Biker project has been inspired by some real toy kit originally developed for educational purposes by Peter Thron et al. in Ilmenau, Germany, but nowadays available as a commercial product [4]. The authors' project has no immediate commercial goals and, thus, does not interfere with the source of inspiration.

Figure 1 is intended to give an impression of the current state of implementation.

The Solar Biker Laboratory is an interactive playground on the Internet. The authors are aiming at a series of similar Web laboratories providing useful content for the playful acquisition of knowledge at school. Questions concerning access to and administration of those laboratories are beyond the limits of the present paper. The focus of the present investigations is on the *evaluation* of the *impact* of the *novel technology*.

For this purpose, this chapter is providing a sketch of what is in the Webble technology seen from the perspective of the Solar Biker educational application.

In particular applications such as the present one, the Webble technology is providing a repository of building blocks as illustrated in figure 2. The building blocks are pads.

Fig. 2. Repository

Every pad has its model view controler architecture. What a pad does and how it looks is defined internally in so-called slots. By setting a slot value, you can modify both the appearance and the functionality of a pad. This includes the position of a pad.

From the user's point of view, pads may be used very much like Lego building blocks. You take them and plug one on top of the other. In the example shown in figure 1, the sun is plugged on top of the environment.

Plugging pads together establishes connections between particular slots of these pads. The data flow between slots allows for a coordinated functionality of the individual components of a composite pad.

For illustration, the solar biker on display in figure 1 will be driven by some energy provided by the solar cell in the left lower corner. This means that the solar cell is providing some slot value to the moving pads of the solar biker. When a cloud is plugged onto the environment pad, this cloud pad is sending some slot value to the environment pad. This value is used to modify, in fact to diminish, the value of the energy slot delivered by the solar cell pad. How much the cloud pad is dimming the solar cell energy value depends on the position of the cloud pad on the environment pad. In such a way, direct manipulation of the pads on the playground dtermines the behavior of the composite pad under construction.

In an expert mode, all details are accessible to the human manipulating pads. One may freely choose pads, modify them and plug them together with other pads. The result is always a composite pad the structure of which may be seen as a tree.

For educational purposes, the freedom of access has to be controled according to the knowledge and skills of the learners. Didactic intentions such as a certain degree of exploration in learning may be a further source of access regulation.

Technology may be modified to support playful learning and construction. One of the key features is automatic plug-in. When one pad is moved over another pad which it fits, the right slots are connected automatically. The whole construction process is just a playful drag and drop.

3 The Meme Media Peculiarities

The *Meme* concept has been introduced and the term has been coined with high ideological ambitions [2, 3]. There is–roughly speaking–the idea of non-biological evolution as observable in areas such as fashion, architecture and, perhaps, religion.

Yuzuru Tanaka has been excited by the idea to foster evolution of ideas by means of information and communication technologies [1]. In response to the challenge, he took the initiative to develop and implement digital meme concepts. Tanaka coined the term *Meme Media* and developed what was then called *Meme Media technologies* in its prominent variants of *IntelligentPad* and *IntelligentBox*.

Webble technologies [5] are a modern Internet-enabled version of IntelligentPad. The way from conventional IntelligentPad to current Webbles may be studied on the basis of a few representative IntelligentPad publications such as [6], [7], [8], [9], [10], [11], and [12].

IntelligentPad, in general, and Webble technology, in particular, is more than just another middleware for the implementation of Web services. A short illustration from the Solar Biker application shall clarify the pecularities under consideration.

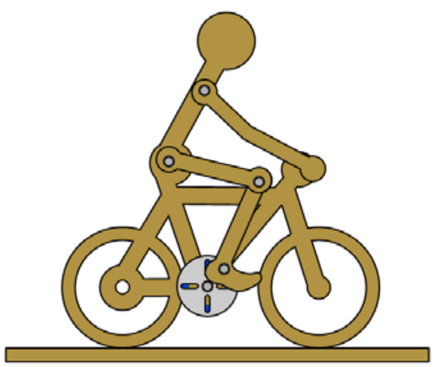

Fig. 3. A Basic Solar Biker Construct

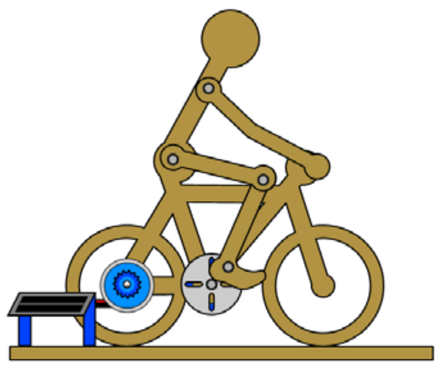

Fig. 4. Photo-Voltaic Propulsion Added

All components used in one of the constructions on display in the figures 3 and 4 are pads having their own individual model-view-controler architecture. They are able to perform their corresponding functionality without embedding into a larger machinery. They run in any browser equipped with a Silverlight plug-in.

Figure 4 shows an extended variant of the construct in figure 3. Some photo-voltaic cell has been connected to an electrical engine which is attached to a gear fixed to the rear wheel of the bike.

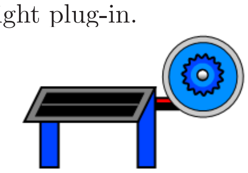

Fig. 5. Solar Engine

The Solar Biker torso shown in figure 4 is a composite pad. Learners may peel off any component of this composite pad. Readers may think of pads as trees in a mathematical sense–see figure 6 for illustration–such that components are subtrees. Those subtrees decribe components such as the propulsion unit on display in figure 5.

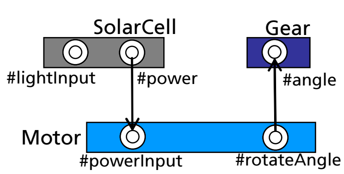

Fig. 6. The Propulsion Pad

These peculiarities sketched above allow for a variety of manipulations which are relevant to teaching and learning. They are particularly important to didactics relying on communication and cooperation. Because every functional component of a composite pad preserves its functionalities after being peeled off, learners may, e.g., exchange unfinished constructs or engage in shared construction projects.

The key interest of the authors is–beyond the standard issues of usabilty–how much the peculiarities of Webble technolgy are perceived by human teachers and learners and to what extent they are appreciated and useful in learning.

4 Solar Biker Lab Evaluation

As said above, the focus of the present investigations is on the *evaluation* of the *impact* of the *novel technology*.

There will be usability tests and the like as usual. But these questions are not within the focus of the present paper. Instead, the authors want to investigate to what extent and with what impact users are perceiving the peculiarities of the Webble technology.

There is a need to distinguish two basic categories of users. First, there are those setting up e-learning contents for the intended learner audience. Second, there are the learners by themselves.

4.1 The Media and Technology Perception by Authors

Those who set up content of interactive laboratories such as the Solar Biker may be teachers, tutors, content providers or software specialists. Due to the mediatization of our contemporary society [13], their affinity to information and communication technologies is continuously changing. Nevertheless, we still have to assume a low familiarity with digital media technology among teachers.

The authors do not expect to have access to a larger community of authors. Thus, they are planning for systematic qualitative investigations [14] of way in which authors cope with the novel potentials of the technology.

4.2 The Media and Technology Perception by Learners

In contrast, learners are–on the average–persons who are, at least to some extent, familiar with information and communication technologies. They have to be kept from dealing with too many technological detail in addition to the main learning task and from being bothered with technicalities of handling the software. Learners playing with the digital Solar Biker toy kit shall learn about solar energy and photo-voltaic technology. The should be encouraged to play with the system and to explore, for illustration, the effects of putting more clouds to the sky.

Both for quantitative and qualitative approaches [15], there is a large variety of settings.

Shall we compare two groups of learners where the one group has the physical toy kit [4] available whereas the other one uses the virtual laboratory over the Internet? What are the details we are looking at and asking for?

4.3 Survey of the Evaluation Approach

To avoid confusions and inappropriate expectations, the authors are not going to present evaluation results within this paper. Instead, the authors are planning (i) to present the Solar Biker laboratory under consideration and then, based on the DERIS audience's impression of and knowledge about the system, (ii) to discuss their intended evaluation approach. The authors' hope is for guidance and advice.

In summary, we need to investigate the acceptance of the Solar Biker with regard to the two main target groups: teachers and learners. Some acceptance factors will probably affect both of the groups–e.g. if students learn something about solar technology when using the Solar Biker. However, in some aspects the expectations towards the Solar Biker may differ–maybe it is less important for the teachers, that their students have fun while using the Solar Biker. Therefore two main research questions are outlined, which take the different acceptance factors of the two groups into account. These questions will lead us through the whole evaluation process. The main questions are completed through some more focussed questions.

Research question for the group of learners

- How do young people between 12 and 16 years with different education levels experience the Solar Biker as playful learning tool?
- Which learning methods and scenarios encourage the usage of the solar biker, e.g. how should the introduction be like? What exactly can be learned with the Solar Biker? What are the entertaining factors of the Solar Biker?

Learning about solar technology includes some factual knowledge, e.g. to know which components are necessary to build up a Solar Biker, skills, e.g. like being able to put the components together, as well as motivational aspects, e.g. if students would like to know more about solar technology. Two learning scenarios will be tested: guided learning [16] and discovery learning [17]. A guided learning scenario is defined by instructions of a teacher, e.g. ‘Please build up a Solar Biker, and use the environment pad first.’ The discovery learning scenario is also characterised by an instruction, but a more open one, e.g. ‘You have 20 minutes time to use the Solar Biker as you like’. Entertaining factors will be operationalized as feelings of control, emotional pleasantness (positive feelings like fun) and sovereignty [18]. We will focus on the students’ age between 12-16 years. In Germany it can be expected that students from the seventh grade onwards will have basic knowledge about chemical and physical foundations of solar technology. Solar technology is not a regular topic in school until the tenth grade.

Research question for the group of teachers

- How do school teachers experience the Solar Biker concerning the learning process of their students and the potential of the system to encourage teaching?
- In which learning scenarios can the Solar Biker be used? Which target groups can be reached? What can be learned with the Solar Biker?

In this target group we will focus on the same learning methods, namely guided learning and discovery learning as two possible use cases. We would like to know from the teachers which age the students need to have to deal with the Solar Biker appropriately and how they assess the learning effect.

4.4 Research Design

We would like to continue enhancing the system of the Solar Biker. To be able to influence the design of the Solar Biker constantly by the results of the evaluation, we choose a formative evaluation as our basic research design [19, 15]. The first step will be to ensure that the usability of the system is given and to explore for which age the Solar Biker is most appropriate. As a second step we will conduct a qualitative group discussion with teachers from schools with different education levels, ranging from high school level (’Gymnasium’) to secondary modern school (’Hauptschule’), for which school system the Solar Biker is an appropriate learning tool and in which class levels it can be used best. The results of the group discussion will be integrated in the third step of the evaluation process, a quasi-experimental setting. Hereby it will be compared how the Solar Biker is used in two contrasting education levels. The two different learning methods will be compared within that setting as well. The main focus will be on the perception of the Solar Biker as a new possibility to deal with digital content. Additionally the help function will be evaluated as well as possibilities of collaborative learning (e.g. to send a running biker per mail to a friend). As a last step the results of the experimental study will be discussed again with the teachers in a group discussion to find out how teachers assess the learning progress of their students.

Usability Test A good usability means that the interaction between the user and the computer is experienced as efficient, effective and satisfying. The concept focuses how targets and tasks can be fulfilled and solved through the use of the system. Hence a usability test will detect the main problems within the user-system-interaction as a basis of our evaluation [20].

a) The first usability test will be conducted with a sample of students ranging from 12-16 years. There should be a good balance between students with a very high computer literacy and students with a very low computer literacy. Because the use of the Solar Biker is maybe influenced by knowledge about solar technology and skills in subjects like physics and chemistry, we will compare students with a higher and a lower education level.
b) The other usability test with the teachers from natural and technical sciences will also take the different level of computer literacy into account. In that stage of the evaluation it is necessary to consider the teachers' subjects, because the usability is maybe, as said before, influenced by the knowledge about solar technology, nonetheless the Solar Biker is an interdisciplinary object.

The participants will get the general task to complete the Solar Biker, but there will be also some special tasks to fulfil, to outline the weak spots, e.g. what happens, if somebody build up a completely wrong constellation and how the help system can assist in these situations. The empirical method used here is an observation, which will use both structured and open criteria to describe the behaviour of the participants. The test will end with a very short interview including some open questions to detect weak spots which the participants would like to remark. Here, too, we will ask in detail how the help function is perceived–a good result would be, if the participants receive the help function as a very individual helping hand and personal assistance.

The concept of usability is not able to explain the usage and experience of the system in general, e.g. if it is fun to use the Solar Biker and if the uniqueness of the technology is experienced as an added value. These aspects will be clarified through the next steps of the evaluation. But before the results of the usability test will end in a revise of the Solar Biker.

Group Discussion I The participants in the group discussion [21] will range from the lower to the higher education levels. The teachers will get an introduction into the Solar Biker system. Afterwards they will be asked for which age and class level the Solar Biker is useful and if they think that they would use it in their schools and why. Furthermore the innovative potential of the Solar Biker in particular and interactive laboratories modelled with Webble technology in general will be discussed.

Quasi-Experimental Setting The quasi-experimental setting [19] will be conducted in two schools with a different education level. Depending on the results of the group discussion two different class levels will also be integrated. The effort of doing research in three different school levels would probably not be appropriate, but this decision does very much depend on the results of the group discussion with the teachers. For the moment we assume surveying and comparing two different school systems. The sample will take two classes out of these two different school levels, so that we will have a sample of four classes with approximately 80 students. These four classes will be surveyed, two out of every school level, from which one will use the guided learning scenario, the other one the discovery learning approach. To use the Solar Biker within a guided learning scenario will be operationalized by using the Solar Biker in a computer lap during a school lesson. The other class will use the Solar Biker as part of homework, without a specific task but a minimum of time to spend. The teachers will be provided with a standardised introduction to the topic 'solar technology' (e.g. a movie) and a standardised instruction. Hereby the influence of the teacher on the learning result will be reduced.

Before the quasi-experimental study there is a high need to conduct qualitative interviews as pre-test to clarify some key dimensions of the usage, e.g. what specific factors influence the acceptance of the Solar Biker (e.g. '*What do you like about the Solar Biker?*') and how students evaluate the unique character of the system. Therefore some scenarios will be given, like '*Imagine you could send the biker as mail to your friends; would you do so? Why?*' or '*Imagine you could put the biker on different countries on a map, e.g. Australia and Sweden; would that be interesting?*'

Students will get a standardised questionnaire before they use the solar biker, e.g. to ask for the different computer literacy levels and their general usage of interactive media. The knowledge about solar technology and the interest in chemistry, physics etc will be asked as well, finalising the first questionnaire with some socio-economical variables, e.g. sex. The second questionnaire will be given to the students after they used the system, contending closed and open questions. Within that questionnaire items concerning the use of the solar biker as a playful learning tool will be in the focus of interest, which means to integrated dimensions of knowledge (e.g. 'Which components are needed for a solar biker?'), emotional (e.g. 'I don't like to use the internet for learning purposes') and cognitive (e.g. 'The internet offers good possibilities to learn') attitudes, emotional (e.g. 'To handle the solar biker is pleasant to me') and cognitive (e.g. 'The solar biker explains solar technology in an understandable way') opinions and intentions (e.g. 'I would like to use applications like the Solar Biker in school more often').

The main focus within these dimensions will be on the perception of the Solar Biker as a unique possibility to deal with digital content, therefore it will be asked, as how unique, new and novel it is valued (by means of an semantic differential) and which features are liked the most, e.g. to send objects per mail, to get individual help, to explore the possibilities how to stick components together etc.

Group Discussion II As a finishing method again a group discussion with the teachers is planned, so that the teachers can assess the learning process and the usage of the Solar Biker in school. At least it will be discussed, which benefits are perceived (e.g. to collaborate with their students, to monitor their learning progress, that students get individual assistance). Therefore a scenario will be outlined to extend the discussion towards the possibility having a complete interactive laboratory, so that the direction of further software development will integrate the perspective as teachers and learners at one time.

Outlook An additional evaluation could ask for the different perception and usage of a 'real-life' Solar Biker in comparison to the digital one. The comparative study would have to look carefully for possibilities to combine the use of 'real-life' tool kits and digital ones in school. We decided to conduct the basic evaluation first, because we need to know about the best learning scenarios for the solar biker before comparing it to a well established learning method.

5 Acknowledgements

This first version of this manuscript has been submitted when the first and the third author have been visiting the Meme Media Lab. at the Hokkaido University Sapporo, Japan, the place where the technology under consideration has its origins.

The authors are particularly greatful to Yuzuru Tanaka for encouraging their publication activities and to Micke Kuwahara for providing Webble World [5].

The present research and development has been partially supported by the Thuringian Ministry for Culture within the project iCycle under contract PE-004-2-1.

References

1. Tanaka, Y.: Meme Media and Meme Market Architectures: Knowledge Media for Editing, Distributing, and Managing Intellectual Resources. IEEE Press & Wiley-Interscience (2003)
2. Dawkins, R.: The Selfish Gene. Oxford University Press (1976)
3. Blackmore, S.: The Meme Machine. Oxford University Press (1999)
4. Woll Solar Systeme. http://shop.solaranlage.com/shop/solarbiker-p-267.html [latest access on Nov. 9, 2009] (2009)
5. Kuwahara, M.: http://www.meme.hokudai.ac.jp/webbleworldportal/ [last access on Sept. 20, 2009] (2009)
6. Tanaka, Y., Imataki, T.: IntelligentPad: A hypermedia system allowing functional compositions of active media objects through direct manipulations. In: Proceedings of the IFIP 11th World Computer Congress. (1989) 541–546
7. Tanaka, Y.: Intelligentpad as meme media and its application to multimedia databases. Information & Software Technology **38** (1996) 201–211
8. Tanaka, Y.: Meme media and a world-wide meme pool. In: MULTIMEDIA '96: Proceedings of the fourth ACM international conference on Multimedia, New York, NY, USA, ACM (1996) 175–186
9. Tanaka, Y., Fujima, J., Sugibuchi, T.: Meme media and meme pools for re-editing and redistributing intellectual assets. In: Revised Papers from the nternational Workshops OHS-7, SC-3, and AH-3 on Hypermedia: Openness, Structural Awareness, and Adaptivity, London, UK, Springer-Verlag (2002) 28–46
10. Ito, K., Tanaka, Y.: A visual environment for dynamic web application composition. In: HYPERTEXT '03: Proceedings of the fourteenth ACM conference on Hypertext and hypermedia, New York, NY, USA, ACM (2003) 184–193
11. Tanaka, Y., Ito, K., Fujima, J.: Meme media for clipping and combining web resources. World Wide Web **9** (2006) 117–142
12. Tanaka, Y., Fujima, J., Kuwahara, M.: Meme media and knowledge federation. In: KI '08: Proceedings of the 31st annual German conference on Advances in Artificial Intelligence, Berlin, Heidelberg, Springer-Verlag (2008) 2–21
13. Krotz, F.: Mediatisierung: Fallstudien zum Wandel von Kommunikation. VS (2009)
14. Kuckartz, U.: Qualitative Datenanalyse: computergestützt. Methodische Hintergründe und Beispiele aus der Forschungspraxis. VS Verlag (2007)
15. Tisdal, K.M., B., D.J., Gallagher, M.: Researching with Children and Young People: Research Design, Methods and Analysis. SAGE Publications Ltd. (2009)
16. Kirschner, P.A.S.J.u.C.R.E.: Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. Educational Psychologist **41** (2006) 75–86
17. Loyens, S.M.M., Gijbels, D.: Understanding the effects of constructivist learning environments: Introducing a multi-directional approach. Instructional Science **36** (2008) 351–357
18. Früh, W., Schulze, A.K., Wünsch, C.: Unterhaltung durch das Fernsehen: Eine molare Theorie. UVK Verl.-Ges., Konstanz (2002)
19. Bortz, J., Döring, N.: Forschungsmethoden und Evaluation: Für Human- und Sozialwissenschaftler; mit 87 Tabellen. 4., überarb. aufl., nachdr. edn. Springer-Medizin-Verl., Heidelberg (2009)
20. Rubin, J.: Handbook of usability testing: How to plan, design, and conduct effective tests. Wiley, New York (1994)
21. Lamnek, S.: Gruppendiskussion: Theorie und Praxis. 2., überarb. u. erw. aufl. edn. UTB, Stuttgart (2005)

This article was processed using the LaTeX macro package with LLNCS style

Can Rule-Based Mashups play a role in the Cloud?
A Position Paper

Adrian Giurca

Brandenburgische Technische Universität, Germany
Giurca@tu-cottbus.de

Abstract. In the last time the cloud computing stays more and more in the attention of the major IT players. Large companies start moving their businesses towards such an architectural approach. Our proposal is to investigate the potential use of the rule-based mashups to perform Enterprise 2.0 implementations in the cloud. We argue that some of the issues of modeling and executing mashups on the cloud can be addressed by using intelligent, rule-based, mashups and derive some open research questions. We look towards other researchers' feedback including ones which are interested to join our initiative.

1 Basics on Cloud Computing

In a report from June 2008, Gartner defines cloud computing as "a style of computing where massively scalable IT-related capabilities are provided *as a service* using Internet technologies to multiple external customers." *"During the past 15 years, a continuing trend toward IT industrialization has grown in popularity as IT services delivered via hardware, software and people are becoming repeatable and usable by a wide range of customers and service providers,"* said Daryl Plummer, managing vice president and Gartner Fellow. *"This is due, in part to the commoditization and standardization of technologies, in part to virtualization and the rise of service-oriented software architectures, and most importantly, to the dramatic growth in popularity of the Internet."*

Cloud computing is a computing paradigm in which business processes are assigned to a combination of *data*, *software* and *services* available over a network - all of them known as "the cloud" (see Figure 1). Cloud computing enables users and developers to utilize already existent services without any knowledge about the technology infrastructure that supports them.

The basic idea is that instead of writing your own home page or blog or e-commerce web site, and running that on someone else's servers, you write a software application instead. This idea has been pioneered on the web by Amazon Web Services and Salesforce.com, though similar approaches have been used before.

The cloud computing architecture comprises three levels:

1. *Infrastructure as a Service* (IaaS) - offering storage data, network capacity, computing power, and other resources for which you have to pay only the actual resources used. This is the case for the infrastructure offered by IBM Cloud, Amazon Web Services or Apple mobile.me.
2. *Platform as a Service* (PaaS) - where developers create, test and execute out their own applications on the runtime environment provided by the cloud. The runtime can be sold to cloud customers. Significant examples are Google AppEngine Microsoft Azure Services Platform, CogHead (bought by SAP in February 2009), Bungee Labs or Quickbase.
3. Software as a Service (SaaS) - the provider operates a variety of applications in the cloud. They are used by many customers over the Web and only end-user services have to be offered or sold. Significant providers are Oracle OnDemand, Microsoft Office Live Salesforce.com, Zoho or Animoto.

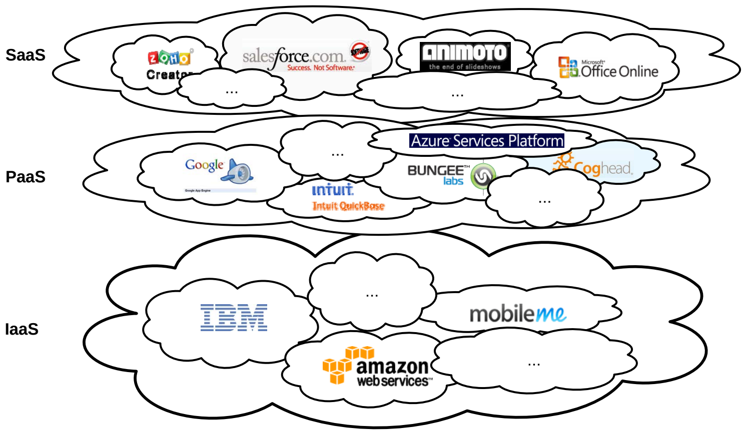

Fig. 1. The Cloud

Cloud computing was started and is developed mainly in enterprises while universities start joining the enterprise research teams just recently. The National Science Foundation announced it had awarded $5 million in grants to fourteen universities as part of its Cluster Exploratory (CLuE) program. The universities receiving money include Carnegie-Mellon, Florida International, MIT, University of Wisconsin, Yale, Purdue, UC-Irvine, UC-San Diego and the San Diego Supercomputer Center, UC-Santa Barbara, University of Maryland-College Park, University of Massachusetts-Amherst, University of Virginia, University of Washington, University of Utah. Corporations include IBM and Google on their "Cloud Computing University Initiative", which serves as a type of spear-head group for the project's goals and focuses results on industry-oriented needs. [1].

[1] See IBM press release, April 23, 2009.

1.1 Enterprise 2.0

Enterprise 2.0 or *Web 2.0 in the Enterprise* was introduced in [11, MacAfee, 2006] to describe how the use of Web 2.0 techniques within an organization can improve productivity and efficiency.

Adopting Web 2.0 techniques allows information workers to control their own experiences with simplified support guidance from IT, and, consequently, create for themselves a more intuitive and efficient work environment.

However, the Enterprise 2.0 scenario remains connected to the main thread of Enterprise Architecture Planning (EAP). In early 90's [19, Spewak and Hill, 1993] defined Enterprise Architecture Planning (EAP) as *"the process of defining architectures for the use of information in support of the business and the plan for implementing those architectures"*.

Spewak's approach to EAP sees four sequential modeling layers:

Q1: *What is my business mission?*
Q2: *What data is required to satisfy the mission?*
Q3: *What application I have to build/use using that data?*
Q4: *What is the technology to implement my applications?*

In an Enterprise 2.0 scenario, while the first layer is much more devoted to the business of the enterprise, the next three are clearly related to the IT landscape.

But how can we use these layers on the cloud?

2 Intelligent Mashups

One can start using the cloud at any of its levels. However, despite the case that most of the service providers in the cloud deliver open APIs, another perspective of services aggregation on the Web is offered by mashups. Basically, a mashup is a Web application that combines data or functionality of one or more services into a single integrated application. In its early stage this concept has been seen just like any other software engineering approach, but in the last time it started to receive attention also from the academia's side (see for example, [12, Morbidoni et al., 2007], [1, Abiteboul et al., 2008], [7, Jarrar and Dikaiakos, 2008]). Artificial intelligence techniques are now applied on a large scale in enterprise applications. Nowadays, in many cases, businesses behavior is expressed naturally through business rules(see [16, Ross, 1997], [17, Ross, 2003]).

2.1 A Basic Mashup Classification

We are not aware of a well established mashup classification, therefore this section will provide a basic one following the aspect of data processing and presentation and the aspect of content processing.

Considering the aspect of data processing and presentation we see the following kinds of mashups:

1. *Data-centric.* Such applications use two or more services to create an integration point towards a business process goal. Usually the used services provide information feeds. Such applications does not focus on any presentation layer (i.e. they may not provide any presentation too). The main activities are related to automating data extraction, data migration and data integration by consuming SOA services.
2. *Presentation-centric.* Application related to presentation of some data. It takes two different resources to create something which is more useful than the standard sum of its parent parts. This business value should be seen on the user's screen. Presentation-centric mashups could also be intra-enterprise (e.g. representing sales with a graphical enterprise logistic system).

According with the content they process we have:

1. *Republishing HTML.* This is an old technique used for a long time: the application retrieves HTML content from specific web sites and then re-publishes a customized content.
2. *Re-syndication.* The simplest form of mashup is taking RSS feeds [18], and either combining it with another feed or embedding it in another location. There are many ways of doing this.
3. *Customized Search.* Nowadays, all search engines offer public APIs, therefore building a customized search engine is no longer a difficult task. Such customized engines are particularly interesting on top of the data provided by social networks or search engines that are customized by using the users' profiles in the social network.
4. *Personalized Portal.* Such application uses available services to define a "custom page" where the user finds its needs. Usually, such a page interacts with the user and is dynamically changed according with the user's profile.
5. *Business Mashup.* They use the enterprise platform to enrich the collaborative action among businesses, between employees, as well as between the business and its customers.

2.2 Rule-Based Intelligent Mashups

When we refer to the term *intelligent mashup* we understand *a mashup enriched with reasoning capabilities usually provided by an inference engine.* In computer science, and specifically the branches of knowledge engineering and artificial intelligence, see an inference engine as a computer application that tries to derive answers from a knowledge base. Our approach on rule-based intelligent mashups focuses on data integration into a single presentation and allows for artificial reasoning and collaborative action among businesses and users.

This approach uses the JSON Rules language introduced in one of our previous works ([[5, Giurca and Pascalau, 2008]]) with the goal to empower Web 2.0 applications, particularly mashups, with rule-based inference capabilities. At the modeling layer we will use an UML based modeling language, designed in [20, Wagner at al., 2006] which was successfully used to model business rules (see [14, Pascalau and Giurca, 2008]).

We argue more to using rules for mashup modeling since the cloud is service-oriented and rules showed their capabilities to model Web Services (see for example, [4, Giurca et al.], [10, Lukichev et al., 2007], [15, Ribaric et al., 2009])

2.3 Benefits and Drawbacks of Rule-Based Mashups

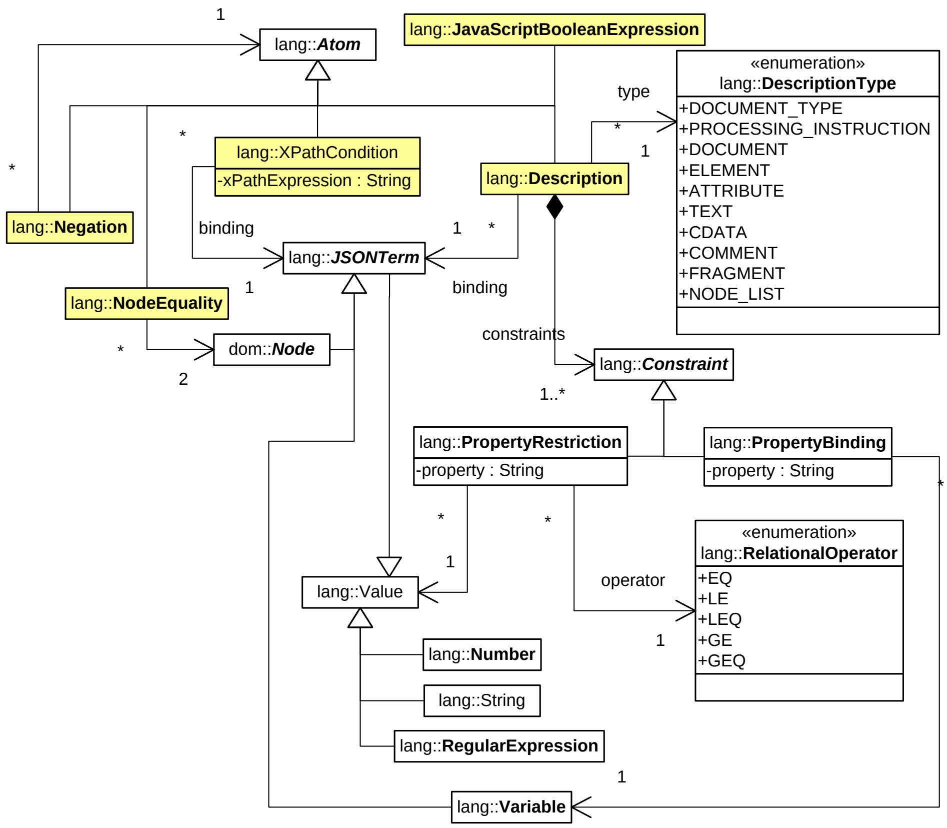

Fig. 2. Basic JSON Rules Conditions

JSON Rules are JavaScript-based reaction rules triggered by DOM Events. Using Ajax technologies, they are able to handle any XML-based format such as Atom [13], [6], RSS 2.0[18], and RDF [9, Klyne and Caroll, 2004]. The language uses a condition language similar with other rule systems and employs any JavaScript function call as actions. The syntax was influenced by the JSON Notation [2, Crockford, 2006] a well known notation to express JavaScript objects.

JSON rules operates on a specific knowledge base with facts obtained from the content. Rules conditions are based on atoms defined on top of Document Object Model (DOM). Figure 2 depicts the UML model of supported atoms. While the metamodel is large we can provide straightforward examples for such conditions. For example, considering an Atom entry (Google) such as:

```
<entry gd:etag="DUEMQno6fyp7ImA9WxVaF08">
  <id>tag:blogger.com,1999:blog-10861780.post-6717232825138410541</id>
  <published>2009-04-13T17:40:00.000-07:00</published>
  <updated>
   2009-04-14T09:48:03.417-07:00
  </updated>
  <app:edited xmlns:app="http://www.w3.org/2007/app">
   2009-04-14T09:48:03.417-07:00
  </app:edited>
  <category scheme="http://www.blogger.com/atom/ns#" term="accessibility"/>
  <title>An ARIA for Google Moderator</title>
  <content type="html">
   ...
  </content>
  <link rel="edit" type="application/atom+xml"
    href="http://www.blogger.com/feeds/..."/>
  <author>
   <name>A Googler</name>
   <email>noreply@blogger.com</email>
  </author>
  <feedburner:origLink>
   http://googleblog.blogspot.com/2009/04/aria-for-google-moderator.html
  </feedburner:origLink>
 </entry>
```

Our rules can handle various conditions such as:

```
// Variable $C is bound to all <category> elements
// in the DOM  and $V retrieve the value of the attribute term i.e.
// ?V == 'accessibility' for the above excerpt
$C:category($V:attributes['term'])

// check if the content of <email> element is a valid email address
// if so, bound the element  to variable $E
$E: email(nodeValue=="match(^[0-9]{4}-(((0[13578]|(10|12))-(0[1-9]|[1-2][0-9]
|3[0-1]))|(02-(0[1-9]|[1-2][0-9]))|((0[469]|11)-(0[1-9]
|[1-2][0-9]|30)))$)")

// check if the node bound to the variable $X is in the
// node list obtained by evaluating the corresponding xPath expression
"$X in "entry//category[@term='accessibility']"
```

A simple analysis on JSON Rules proves that they provide a platform which is able to deal with all kinds of mashups discussed above. However there are drawbacks too. The Table 1 shows rule-based mashups benefits and drawbacks:

Feature	Status	Comments
Easy Modeling	$\Uparrow$	See [20, Wagner at al., 2006]
Support for any XML data formats	$\Uparrow$	Induced by using DOM
Allows re-syndication	$\Uparrow$	DOM combination
Search based on public Web services	$\Uparrow$	via Ajax
Service aggregation	$\Uparrow$	Using ECA rules. See for example, [4, Giurca et al., 2006]
Side effects	$\Uparrow$	Rule actions consist of any JS function call
Any presentation layer	$\Uparrow$	Induced by CSS
Support for public and private mashups	$\Uparrow$	
Declarative programming	$\Uparrow$ / $\Downarrow$	Some people like rules some others not. Newcomers should learn basics about rules.
Speed	?	The engine runs in the browser. Viability tests should be performed. Some JS frameworks are quite fast.
Stability	$\Uparrow$ / $\Downarrow$	As much as any other JavaScript-based application
Security	$\Uparrow$ / $\Downarrow$	As much as any other JavaScript-based application

Table 1. Rule-Based Mashups Benefits and Drawbacks

3 Using Rule-based Mashups on the Cloud

[3, Foster and Tuecke, 2005] analyzes terms such as *software as a service*, *software on demand*, *adaptive enterprise* and *mashups* and concludes that they are overlapping to many extents. We share this view and consider mashups at the SaaS level in the cloud. As a consequence, intelligent mashups should be able to use the PaaS layer as well as resources available in its layer.

Coming back to Spewak approach (Section 1.1) we argue that intelligent mashups provide solutions to the mainstream questions i.e.

Answering Q1: Intelligent mashups act both at SaaS and PaaS level therefore they are able to handle various businesses. Using SalesForce services (CRM) or Zoho Services (e.g. invoicing) put the mashup in the center of our business.

Answering Q2: Intelligent mashups handle all kind of XML content. content as data was proved as being enough in a large number of business applications. Such content is delivered by various platforms from the PaaS level in the cloud (e.g. Enables teams, divisions, partners and vendors to work together effectively by accessing and sharing centralized information.).

Answering Q3: The tendency to use the browser as a client are already for a long time and they are very successful. Google (e.g. Google Apps) is one of the notable leaders and Amazon Web Services is the most powerful e-commerce service-based infrastructure.

Answering Q4: Running mashups in the browser entitle us to believe that a JavaScript engine is the most suitable one.

4 Conclusion and Research Opportunities

We have argued that some of the issues of modeling and executing mashups on the cloud can be addressed by using intelligent mashups based on existing literature, our experience, and an observational case study. Future WWW programming will be strongly oriented on the cloud since it provides various data sources (from the main data creators) and a complex infrastructure using all kinds of services, together with a powerful level of application build on top of strongly established platforms. Inside of this *"cloud of data, infrastructure, technologies and services"*, some issues remain open and offer research opportunities:

Business Level: *How can intelligent mashups handle the legal agreements of using cloud resources?* ENISA (the European Network and Information Security Agency) is conducting a security risk assessment on cloud computing. For the Cloud Risk Assessment, they focus on scenarios including: (a) A user perspective on Cloud Computing (i.e. Small and Medium Enterprises), and (b) Cloud Computing in a eGovernment environment (i.e. national health service). While we find the most of mashups applications related to the first scenario, the second one is also possible. We see necessary to define standard legal agreements for publicly offered data and services. It is to be investigated if legal policies can be exchanged here as in B2B solutions.

Business Level and Technological Level: *How the cloud can guarantee the integrity of data and services offered to mashups?* It is clear that this is one of the key issues on the cloud. If your data is not available to you, for whatever reason, then it is no good for your mashup. Therefore we probably should investigate exception mechanisms. In addition, many service providers on the cloud will provide at least one back up resource, maybe more. Any subscriber should check what provisions are made and choose the data provider accordingly. Finally we see interoperability issues: For example using the CRM from Salesforce.com, you may be limited to its data proprietary format. If you want to move to another service implies how would you get your data back? This shows the necessity for a standard on data interoperability in the cloud (May be OWL 2?).

Conceptual level: *What is the conceptual model of a mashup?* Recently, [8, Jarrar and Dikaiakos, 2009] defines a data mashup language for the "Data Web". By contrary, Google deprecates its mashup editor which was XML based. Therefore we consider that researching a conceptual model of mashups will improve chances to obtain a better definition of this paradigm and will contribute to the cloud computing mainstream too.

Tools Level: *Do we need development tools to model, build and debug intelligent mashups?* The actual mashup market includes many visual tools such as Yahoo Pipes, DERI Pipes, Intel Mash Maker, Microsoft Popfly, or IBM Mashup center.

Therefore we consider that such tools are welcome for intelligent mashups too.

We are looking for groups that may want do cooperative work on these topics but also to collaborate for defining and implementing complex intelligent mashups scenarios. Economic, social, and game production are just some of the domains we are interested in.

References

1. S. Abiteboul, O. Greenshpan, and T. Milo. Modeling the mashup space. In *WIDM'08: Proceeding of the 10th ACM workshop on Web information and data management*, pages 87–94, 2008.
2. D. Crockford. The application/json Media Type for JavaScript Object Notation (JSON). `http://tools.ietf.org/html/rfc4627`, July 2006.
3. I. Foster and S. Tuecke. Describing the Elephant: The Different Faces of IT as Service . *Enterprise Distributed Computing*, 3(6):26–34, July/August 2005.
4. A. Giurca, S. Lukichev, and G. Wagner. Modeling Web Services with URML. In *Proceedings of SBPM'2006*, Budva, Montenegro, 11 May 2006.
5. A. Giurca and E. Pascalau. JSON Rules. In *Proceedings of the Proceedings of 4th Knowledge Engineering and Software Engineering, KESE 2008, collocated with KI 2008*, volume 425, pages 7–18. CEUR Workshop Proceedings, 2008.
6. J. Gregorio and B. de hOra. Atom Publishing Format (RFC5023). `http://tools.ietf.org/html/rfc5023`, 2007.
7. M. Jarrar and M. D. Dikaiakos. Mashql: a query-by-diagram topping sparql. In *ONISW '08: Proceeding of the 2nd international workshop on Ontologies and nformation systems for the semantic web*, pages 89–96, New York, NY, USA, 2008. ACM.
8. M. Jarrar and M. D. Dikaiakos. A Data Mashup Language for the Data Web. In *Proceedings of teh workshop Linked Data on the Web (LDOW2009)*, 2009.
9. G. Klyne and J.J. Caroll. Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation, February 2004. `http://www.w3.org/TR/rdf-concepts/`.
10. S. Lukichev, A. Giurca, G. Wagner, D. Gasevic, and M. Ribaric. Using uml-based rules for web services modeling. In V. Oria, A. Elmagarmid, F. Lochovsky, and Y. Saygin, editors, *Proceedings of the Second International Workshop on Services Engineering*, pages 290–298, Istanbul, Turkey, 16 April 2007. IEEE Computer Society.

11. A. McAfee. Enterprise 2.0: The Dawn of Emergent Collaboration. *MIT Sloan Management Review*, 47(3):21–28, March-May 2006.
12. C. Morbidoni, A. Polleres, G. Tummarello, and D. Le Phuoc. Semantic web pipes. Technical report, DERI - DIGITAL ENTERPRISE RESEARCH INSTITUTE, November 2007.
13. M. Nottingham and R. Sayre. Atom Publishing Format (RFC4287). `http://tools.ietf.org/html/rfc4287`, 2005.
14. Emilian Pascalau and Adrian Giurca. Can URML model successfully Drools rules? In *Proceedings of 2nd East European Workshop on Rule-Based Applications (RuleApps2008)*, volume 428, Patras, Greece, July 21-22 2008. CEUR Workshop Proceedings.
15. M. Ribaric, S. Sheidaei, D. Gasevic, M. Milanovic, A. Giurca, S. Lukichev, and G. Wagner. *Modeling of Web Services using URML, R2ML and model transformations.* IGI Publishing, 2009.
16. R. G. Ross. *The Business Rule Book: Classifying, Defining and Modeling Rules.* Database Research Group, Inc., Boston (MA), 2nd edition edition, 1997.
17. R. G. Ross. *Principles Of Business Rule Approach* . Addison-Wesley Professional, 2003.
18. RSS. RSS 2.0 Specification, version 2.0.11. `http://www.rssboard.org/rss-specification`, March 2009.
19. S.H. Spewak and S. C. Hill. *Enterprise architecture planning developing a blueprint for data, applications, and technology.* Wiley, New York, 1993.
20. G. Wagner, A. Giurca, S. Lukichev, G. Antoniou, C. V. Damasio, and N. E. Fuchs. Language improvements and extensions. Technical report, Munchen, Germany, April 2006.

Proposal of a Graph-Oriented Approach to Verification of XTT2 Rule Bases*

Agata Ligęza, Grzegorz J. Nalepa, Antoni Ligęza

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

Abstract. The main focus of this paper is on the discussion of formal rule verification using graph-based approach. XTT2 is a custom rule representation method, that introduces a structured rule base composed of extended decision tables linked in a tree-like structure. Considering the complex nature of the XTT2 structure, only the local, table level verification has been considered so far. However, graph-oriented verification is a powerful solution to the analysis of rule-based systems. It can be applied to provide global verification of the XTT2 knowledge bases. The principal idea consists in representing XTT2 rules as a directed hypergraph. All of rule formulas are transformed into vertices and appropriate hyperarcs are determined. This restructuring of the XTT2 knowledge base allows to provide verification using graph algorithms. Preliminary evaluation of this approach shows that the graph-oriented verification is a promising solution to provide a formal analysis of XTT2 rules.

1 Introduction

Rule-based systems (RBS) [1] are an important class of intelligent systems [2]. Their formal description allows for a formal analysis of important system properties. Therefore, it is possible to ensure their quality and safety at the early design stages.

The main focus of the paper is the formal verification of rules [3,4]. Formal rule properties have to be considered w.r.t. to a given knowledge formalization format. Therefore, the rule formalization for the XTT representation is given [5,6]. The representation introduces a structured rule base composed of extended decision tables linked in a tree-like structure. The rule formalization is given using the ALSV(FD) logic [1,7]. Considering the complex nature of the XTT knowledge base structure, so far only local, table level verification has been provided.

The approach proposed in this paper uses graph-based representation. Graph-oriented verification is a powerful solution to the analysis of rule-based systems. It can be applied to provide global verification of the XTT2 knowledge bases.

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

The principal idea consists in representing XTT2 rules as a directed hypergraph. All of rule formulas are transformed into vertices and appropriate hyperarcs are determined. This restructuring of the XTT2 knowledge base allows to provide a verification using graph algorithms. Preliminary evaluation of this approach shows that the graph-oriented verification is a promising solution to provide a formal analysis of XTT2 rules. In the paper a practical example is provided.

2 XTT2 Rule Language Formalization

The formalization for XTT^2 representation is based on ALSV(FD) logic [1,7,6]. The ALSV(FD) provides a much higher expressive power than the propositional calculus, while providing tractable inference. Therefore, a format of rule is more complex. In a general case, rule expressed by attributive logic is represented as (1)

$$\begin{aligned} \mathrm{rule}(i) : \psi \ \wedge & \\ & A_1 \in t_1 \wedge A_2 \in t_2 \wedge \cdots \wedge A_n \in t_n \\ & \longrightarrow \\ & retract(B_1 = b_1, B_2 = b_2, \ldots, B_b = b_b) \\ & assert(C_1 = c_1, C_2 = c_2, \ldots, C_c = c_c) \\ & H_1 = h_1, H_2 = h_2, \ldots, H_h = h_h \\ & next(j), \ \ else(k) \end{aligned} \tag{1}$$

In (1) a formula ψ describes context. $A_1 \in t_1 \wedge A_2 \in t_2 \wedge \cdots \wedge A_n \in t_n$ is a precondition formula. C_j $(j = 1, \ldots, c)$ and B_i $(i = 1, \ldots, b)$ correspond to facts to assert and to retract from the knowledge base. Conclusions (decisions or actions) are represented by $H_1 = h_1, H_2 = h_2, \ldots, H_h = h_h$. There is also a control statement introducing the next or alternative rule.

In the case of XTT^2 representation, the logical rule format is as follows:

$$r\colon (A_1 \propto_1 V_1) \wedge (A_2 \propto_2 V_2) \wedge \cdots \wedge (A_n \propto_n V_n) \longrightarrow RHS, \tag{2}$$

where $\propto_i \in \{=, \neq, \in, \notin\}$ for simple attributes (taking single values) and $\propto_i \in \{=, \neq, \subseteq, \supseteq, \sim, \not\sim\}$ for general attributes (taking set values). The form of RHS is as in (1). For more details on the XTT2 rule formalization using ALSV(FD) see [7,6].

3 Basic Formal Analysis of XTT Rules

The quality of a rule-based system is dependent on the quality of a knowledge base. What is more important, anomalies in the set of rules could result in serious faults in system's responses. Therefore the analysis of knowledge base is an important step during development of a rule-based system.

The issues of verification and validation were discussed by many authors. Differences in their approaches to V&V start at the definition level. In this paper verification and validation processes are defined as follows:

Verification is a process in early design phase, aimed at checking if the system meets its constraints and requirements ([8,9,10,11,12]).

Testing is a process aimed at analyzing the system work, by comparing system responses to known responses for special input data ([8]).

Validation is a case of testing, aimed at checking if the system meets user's requirements ([8]).

A summary of analysis techniques and tools is presented in [13].

The classification of potential errors and deformation of knowledge base was also widely discussed, with some practical taxonomies of anomalies given (see [14,1]). In this paper, from the formal point of view, rule anomalies are be divided into three main categories: 1) *incompleteness*, 2) *indeterminism*, and 3) *overdeveloped set of rules*.

Let the knowledge base be described by rules:

$$\begin{array}{l} r_1: \ \Psi_1 \rightarrow h_1 \\ r_2: \ \Psi_2 \rightarrow h_2 \\ \qquad \vdots \\ r_n: \ \Psi_n \rightarrow h_n \end{array} \tag{3}$$

Completeness ensures that for any input state the system reacts and produces some response (conclusion, decision or action) ([15]). In other words, the system with the set of rules (3) is *logically complete* if a disjunction of preconditions is a tautology: $\models \Psi_1 \ \vee \ \Psi_2 \ \vee \ \ldots \ \vee \ \Psi_n$ The knowledge base is incomplete, when *unreachable*, or *dead-end* rules exist in a rule set, or some rules are *missing*.

Determinism guarantees that the system always produces the same reaction for the same input data. In other words for any input state the system finds a unique solution ([15]). From the formal point of view the set (3) is indeterministic "if there exists a state described by formula ψ, such that simultaneously $\phi \models \Psi_1$ and $\phi \models \Psi_2$ and $h_1 \neq h_2$" ([16]).
The system is indeterministic, if there are *contradictory rules* in knowledge base. *Inconsistency* also is a cause of indeterminism.

Minimal number of rules indicates a set of rules without *redundant*, *subsumed* rules. What is more, the set of rules should produce the same reactions as a overdeveloped set.

All of above features – completeness, determinism, minimal number of rules – should be provided to assure reliability, safety and efficiency of the rule-base system ([15]). The XTT^2 representation introduces a structure of knowledge base by identifying *rule contexts*. Implicitly the context is identified with an extended decision table. Isolation of contexts allows to provide *local analysis* – contexts can be verified separately. In practice, the analysis of the XTT knowledge base is provided by HalVA Verification Framework [17]. The main purpose of the framework is the local verification. HalVA in implemented in Prolog. The verification framework was developed as a plug-in of the HeaRT inference engine [18]. The

HalVA provides the verification of completeness, contradiction and subsumption. What is more, the number of rules can be reduced. HalVA verification is focused on a local level, the schema of the XTT table is considered.

All of verification procedures are based on inference rules for ALSV(FD) introduced in [19]. For state described by ϕ a precondition $(A_i \propto_i V_i)$ (where $\propto_i \in \{=, \neq, \in, \notin\}$ for the simple attribute and $\propto_i \in \{=, \neq, \subseteq, \supseteq, \sim, \not\sim\}$ for the general attribute, $i = 1, \ldots, n$) is satisfied, if simultaneously:

- V_i is a value from the domain of A_i,
- ϕ_{A_i} is a value from the domain of A_i, where ϕ_{A_i} is a value of attribute A_i in formula ϕ,
- a formula $(A_i = \phi_{A_i})$ is a logical consequence of a formula $(A_i \propto_i V_i)$.

The basic idea in the verification of *completeness* consists in checking all input states. Domains of attributes are considered. The Cartesian product of domains determine all states for the context (table). For every tuple corresponding to an input state, the algorithm checks if preconditions of any rule are satisfied. If there is no rule to execute, the considered state is reported as uncover. Based on all uncovered states, a proposal of a new rule is given. The analysis ends, when all states are checked. Since domains of attributes are finite, the verification procedure terminates after a finite number of discrete steps.

Verification of *contradiction* is based on a pairwise comparison of rules. Two rules, executable in the same time (for the same state), are taken into consideration. The comparison concerns the right-hand side of rules. If conclusions are inconsistent, the conflict is reported. The verification procedure stops when all possible comparisons are done.

The pairwise comparison of rules is also used to verify *subsumption*. However, the analysis concerns both sides of rules. One rule is subsumed by another, if its preconditions are more specific, but simultaneously conclusions are more general. The verification procedure finds two rules in the context (table), executable for the same state. Then checks, whether there is relation between conclusions. The algorithm provides all comparisons. This strategy allows to detect identical rules. In this case, a rule is reported as subsuming another and the other subsuming the first one.

HalVA allows to reduce an overdeveloped set of rules. The reduction can be done by using the dual resolution ([16]). If rules produce the same conclusions and in the precondition part exists at least one the same formula, the remaining formulas are joined into one. All possible reductions are reported. What is more, proposals of new rules are introduced.

Unfortunately, solutions provided by HalVA are limited. The serious issue is computational complexity of provided algorithms. Verification of completeness could cause a combinatorial explosion, if domains of attributes are outsized. The other technique – the pairwise comparison of rules – is also dependent on a size of the considered case. Generally, to verify a set of n rules, $\binom{n}{2}$ comparisons need to be done. What is most important, all of HalVA verification features are focused on a local analysis. The limitation of the scope indicates a verification of some context (implicitly a table) only. Therefore, HalVA procedures can point out anomalies in some specific area. Unfortunately, the problem of the global quality of the knowledge base remains unsolved. This is where the new approach proves to be useful.

4 Graph-oriented Verification Solution Proposal

To verify rule-based systems in global scope the graph-oriented approach is introduced. The main concept consists in representing XTT2 rules as a directed hypergraph. Necessary basic definitions are introduced bellow.

"Let $X = \{x_1, x_2, \ldots, x_n\}$ be a finite set, and let $\mathcal{E} = (E_i | i \in I)$ be a family of subsets of X. The family $\mathcal{E}$ is said to be a *hypergraph on* X if

1. $E_i \neq \emptyset \ (i \in I)$
2. $\bigcup_{i \in I} E_i = X$.

The couple $H = (X, \mathcal{E})$ is called a *hypergraph*. $|X| = n$ is called the *order* of this hypergraph. The elements $x_1, x_2, \ldots, x_n$ are called the *vertices* and the sets $E_1, E_2, \ldots, E_m$ are called the *edges* ([20])."

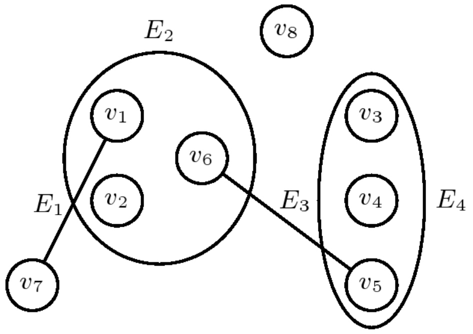

Fig. 1. A hypergraph: $X = \{v_1, \ldots, v_8\}$ – vertices, $\mathcal{E} = \{E_1, \ldots, E_4\}$ – edges, where: $E_1 = \{v_1, v_7\}$, $E_2 = \{v_1, v_2, v_6\}$, $E_3 = \{v_5, v_6\}$ and $E_4 = \{v_3, v_4, v_5\}$.

A hypergraph is presented in Fig. 1. For *directed* edges (hyperarcs) *initial* and *terminal* endpoints can be pointed out (vide Fig. 2).

Let a hypergraph $G = (X, \mathcal{E})$ be considered (vide Fig. 3). "The *outer demi-degree* $d_G^+(x)$ of a vertex x is defined as the number of arcs having x as their initial endpoint ([20])." On the other hand, "the *inner demi-degree* $d_G^-(x)$ of a vertex x is defined as the number of arcs having x as their terminal endpoint ([20])." A vertex x is said to be a *source*, if $d_G^-(x) = 0$. A vertex x is said to be a *sink*, if $d_G^+(x) = 0$.

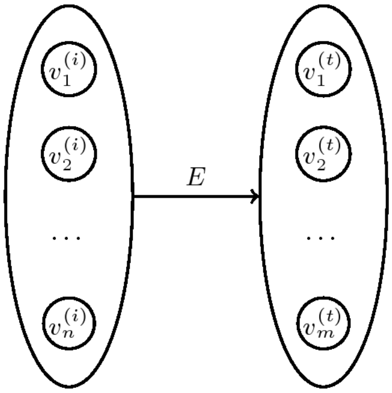

Fig. 2. A hyperarc $E = \{v_1^{(i)}, \ldots, v_n^{(i)}, v_1^{(t)} \ldots, v_m^{(t)}\}$. Vertices: $v_1^{(i)}, \ldots, v_n^{(i)}$ are initial endpoints and $v_1^{(t)}, \ldots, v_m^{(t)}$ are terminal endpoints of the hyperarcs

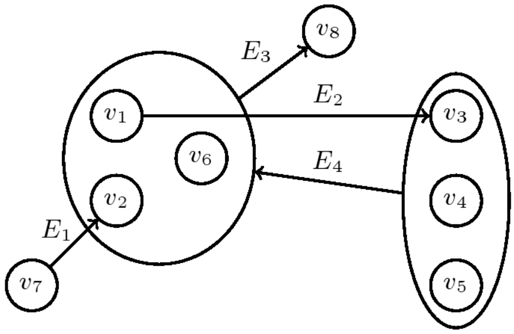

Fig. 3. A directed hypergraph. Degrees of v_1: $d^+(v_1) = 2$, $d^-(v_1) = 1$. Vertices v_4, v_5, v_7 are sources. A vertex v_8 is a sink

For a directed hypergraph $H = (X, \mathcal{E})$ the *incidence matrix* is a matrix $((a_{ij}))_{m \times n}$ "with m rows that represent the edges of H and n columns that represent the vertices of H, such that" ([20]):

$$a_{ij} = \begin{cases} -1 & \text{if } x_j \in E_i \text{ and } x_j \text{ is an inital endpoint of } E_i \\ 1 & \text{if } x_j \in E_i \text{ and } x_j \text{ is an terminal endpoint of } E_i \\ 0 & \text{if } x_j \notin E_i \end{cases}$$

Incidence matrix for the hypergraph in the Fig. 3 is introduced in Table 1.

To provide verification using the graph theory an appropriate transformation of XTT rules needs to be done. The first step is to determine vertices. If vertices correspond to all possible values from domains of attributes, it causes a combinatorial explosion for outsized domains. Therefore, not single values, but whole

Table 1. The incidence matrix for the hypergraph in the Fig. 3.

	E_1	E_2	E_3	E_4
v_1	0	-1	-1	1
v_2	1	0	-1	1
v_3	0	1	0	-1
v_4	0	0	0	-1
v_5	0	0	0	-1
v_6	0	0	-1	1
v_7	-1	0	0	0
v_8	0	0	1	0

formulas are transformed into vertices. However, using formulas to describe vertices results in an additional assumption. The graph-oriented approach can be provided, if the table-level verification of completeness is done.

There are three classes of vertices in the graph representation of rules:

- *sources*: the source corresponds to a formula $(A_i \propto_i V_i)$, where the attribute A_i is used only in a precondition part of rules.
- *sinks*: the sink corresponds to conclusion part of rules.
- *others*: all of others vertices, which are neither source nor the sinks, are related to formulas $(A_i \propto_i V_i)$, where attribute A_i appears in both a precondition and a conclusion part of rules.

The assumption of local verification of completeness ensures the absence of isolated vertices. Therefore, for the source x: $d^+_G(x) > 0$, for the sink x: $d^-_G(x) > 0$, and for *others* x: $d^+(x) > 0$ and $d^-(x) > 0$. In this representation, a rule from the knowledge base is transformed into a hyperarc. The rule also determines the direction of the hyperarc. Therefore, each rule has a unique representation. Initial endpoints of the hyperarc are related to formulas in a precondition part of the rule. The hyperarc is terminated in several ways. Firstly, if a conclusion in the rule is related to a sink, the sink becomes a terminal endpoint of the hyperarc. Secondly, if an attribute in a conclusion appears in a clause in a precondition part of other rule and if the clause is more general than the conclusion, the vertex related to the clause terminates the hyperarc. Finally, if control statements (*next*, *else*) are introduced in the rule, appropriate vertices (clauses) become terminators of the hyperarc.

Table 2. An example of a XTT2 system

r_1	A in $[a_1, a_2]$	B set b_1	C set c_1
r_2	A eq a_3	B set b_2	C set c_2

r_3	C eq c_1	D set d_1

The hypergraph for the system is presented in Fig. 4. Let the XTT2 system be described by rules r_1, r_2 and r_3 (vide Table 2). The attribute A appears only in a precondition part. On the other hand, attributes B and D appear in a decision part. Therefore, a hypergraph for the system has vertices:

v_1: (A in $[a_1, a_2]$)	v_3: (B set b_1)	v_5: (C eq c_1)	v_7: (D eq d_1)
v_2: (A eq a_3)	v_4: (B set b_2)	v_6: (C set c_2)	

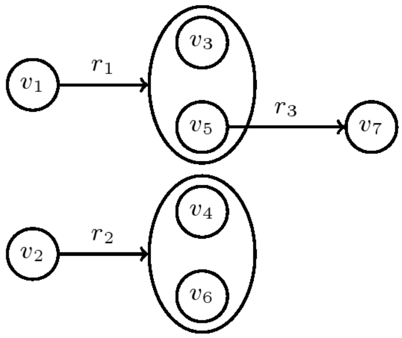

Fig. 4. A directed hypergraph for the subsystem

The introduced representation of rules allows to define anomalies of the knowledge base as follows ([21]):

Inconsistency – exists if there exists a path from vertex X to its exclusive vertex $\neg X$ ([21]).

Contradiction – exists if two paths from vertex X to vertices Y and $\neg Y$ exist in the graph.

Redundancy – exists if at least two different paths from vertex X to Y exist.

Circularity – exists if there is a cycle in the graph.

Unreachability – exists if a vertex which is not the beginning of a path to any output node and is not the end of a path from any input node can be found in the graph.

Locally verified inconsistency, contradiction and redundancy assure correctness of rules in the table scope only. The new representation of rules allows to examine paths in the graph. Deformed ones are reported. However, the vital question is the global verification of completeness. Local analysis of verification is helpful, but it is not limitative in the global scope. Fortunately, graph-oriented approach allows to detect *unreachable formula* or *dead-end formula* – main sources of incompleteness.

As it was mentioned earlier, if the vertex $v \in V$ is neither the source nor the sink, its outer and inner demi-degree are greater than zero. If $d^+(v) = 0$ and

$d^-(v) > 0$, the vertex is a dead-end formula. On the other hand, if $d^+(v) > 0$ and $d^-(v) = 0$, the vertex is a unreachable formula.

The graph-oriented verification is a powerful solution to the analysis of rule-based systems. It can be applied to provide a global verification of the XTT2 knowledge bases ([5,22,6,23]). The most important feature of this approach is its ability to verify completeness in a global scope.

5 Verification Example

Consider the thermostat control system ([16]). The general schema of the system is presented in Fig. 5. Tables *dt* and *th* are considered. The subsystem will be transformed into a hypergraph (vide Fig. 6).

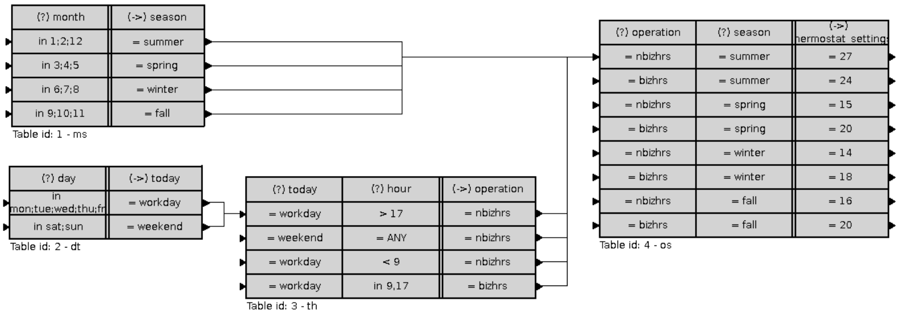

Fig. 5. The general schema of the thermostat control system

Table 3. The incidence matrix for the thermostat control system

vertex	formula	dt/1	dt/2	th/1	th/2	th/3	th/4
1	day in [mon;tue;wed;thu;fri]	-1	0	0	0	0	0
2	day in [sat;sun]	0	-1	0	0	0	0
3	today eq workday	1	0 (**1**)	-1	0	-1	-1
4	today eq weekend	0	1 (**0**)	0	-1	0	0
5	hour gt 17	0	0	-1	0	0	0
6	hour eq any	0	0	0	-1	0	0
7	hour lt 9	0	0	0	0	-1	0
8	hour in [9 to 17]	0	0	0	0	0	-1
9	op eq nbizhrs	0	0	1	1	1	0
10	op eq bizhrs	0	0	0	0	0	1

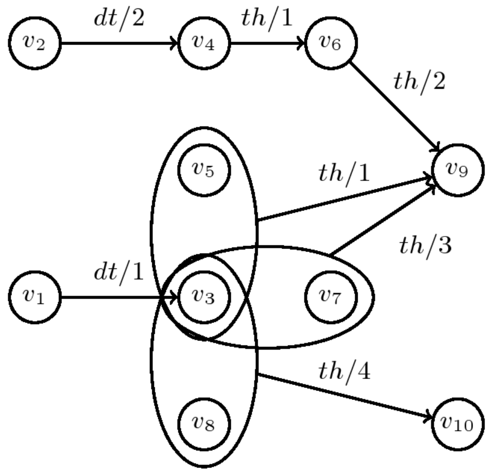

Fig. 6. A directed hypergraph for the subsystem

The hypergraph related to the subsystem is presented as an incidence matrix (vide Table 3). Let the deformation during the knowledge acquisition phase be introduced. The altered knowledge base could be described by hypergraph where vertices 3 and 4 are modified as shown in the parentheses. Locally the knowledge base is still valid: it is complete and there is no contradiction or redundancy. However, consider the global scope: vertices 1, 2, 5, 6, 7 and 8 are sources; vertices 9 and 10 are sinks. Demi-degrees (inner and outer) of other vertices – 3 and 4 – should be greater than zero. Unfortunately, the vertex 4 is deformed. Its inner demi-degree equals zero. There is no path from any source to this vertex. It is an *unreachable formula*. Therefore, the knowledge base is globally incomplete.

6 Concluding Remarks and Future Work

In this paper a graph-based representation for XTT2 rules is proposed. It is aimed at the graph-oriented verification, which is a powerful solution to the analysis of rule-based systems. It can be applied to provide global verification of the XTT2 knowledge bases. In the paper a practical example is provided, and a preliminary evaluation of the approach is given.

In the global verification of XTT2, formulas in precondition and decision parts are transformed into vertices. Adequate hyperarcs are determined. The analysis of hypergraph structure allows to detect unreachable and dead-end formulas. This allows for a more efficient analysis of completeness of the rule base.

The research presented in the paper is work in progress. Detection of redundancy or contradiction needs to examine paths in the hypergraph. This is a more difficult problem than the analysis of structure of the hypergraph. The correlation between vertices on a path needs to be checked. Therefore, this method should be improved. Another important question is reconstructing the system from the hypergraph. At the moment the transformation to hypergraph is not entirely reversible. Yet another issue is the global inference support based on the hypergraph structure. Apparently, in this approach the inference can be optimized on both the table level and global level.

References

1. Ligęza, A.: Logical Foundations for Rule-Based Systems. Springer-Verlag, Berlin, Heidelberg (2006)
2. Hopgood, A.A.: Intelligent Systems for Engineers and Scientists. 2nd edn. CRC Press, Boca Raton London New York Washington, D.C. (2001)
3. Liebowitz, J., ed.: The Handbook of Applied Expert Systems. CRC Press, Boca Raton (1998)
4. Giarratano, J., Riley, G.: Expert Systems. Principles and Programming. Fourth edition edn. Thomson Course Technology, Boston, MA, United States (2005) ISBN 0-534-38447-1.
5. Nalepa, G.J., Ligęza, A.: A visual edition tool for design and verification of knowledge in rule-based systems. Systems Science **31**(3) (2005) 103–109
6. Nalepa, G.J., Ligęza, A.: Hekate methodology, hybrid engineering of intelligent systems. International Journal of Applied Mathematics and Computer Science (2009) accepted for publication.
7. 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, RuleApps2008: Patras, 22 July 2008, Patras, University of Patras (2008) 11–15
8. Tepandi, J.: Verification, testing, and validation of rule-based expert systems. Proceedings of the 11-th IFAC World Congress (1990) 162–167
9. Andert, E.P.: Integrated Knowledge-Based System Design and Validation for Solving Problems in Uncertain Environments. International Journal of Man-Machine Studies **36**(2) (1992) 357–373
10. Nazareth, D.L.: Issues in the Verification of Knowledge in Rule-Based Systems. International Journal of Man-Machine Studies **30**(3) (1989) 255–271
11. Nguyen, T.A., Perkins, W.A., Laffey, T.J., Pecora, D.: Checking an Expert Systems Knowledge Base for Consistency and Completeness. In: IJCAI. (1985) 375–378
12. Suwa, M., Scott, C.A., Shortliffe, E.H.: Completeness and consistency in rule-based expert system. In: Rule-Based Expert Systems. Addison-Wesley Publishing Company (1985) 159–170
13. Wentworth, J.A., Knaus, R., Aougab, H.: Verification, Validation and Evaluation of Expert Systems. World Wide Web electronic publication: http://www.tfhrc.gov/advanc/vve/cover.htm
14. Vermesan, A.I., Coenen, F., eds.: Validation and Verification of Knowledge Based Systems. Theory, Tools and Practice. Kluwer Academic Publisher, Boston (1999)
15. Ligęza, A.: Logical Support for Design of Rule-Based Systems. Reliability and Quality Issues. ECAI'96 Workshop on Validation, Verification and Refinement of Knowledge-Based Systems (1996)

16. Ligęza, A.: Logical Foundations for Rule-Based Systems. Uczelniane wydawnictwa Naukowo-Dydaktyczne AGH w Krakowie (2005)
17. Ligęza, A., Nalepa, G.J.: Proposal of a formal verification framework for the xtt2 rule bases. In: CMS'09: Computer Methods and Systems 26–7 November 2009, Kraków, Poland, AGH University of Science and Technology, Cracow, Oprogramowanie Naukowo-Techniczne (2009)
18. Nalepa, G.J., Ligęza, A., Kaczor, K., Furmańska, W.T.: Hekate rule runtime and design framework. In Adrian Giurca, Grzegorz J. Nalepa, G.W., ed.: Proceedings of the 3rd East European Workshop on Rule-Based Applications (RuleApps 2009) Cottbus, Germany, September 21, 2009, Cottbus, Germany (2009) 21–30
19. Nalepa, G.J., Ligęza, A.: HeKatE Methodology, Hybrid Engineering of Intelligent Systems. Int. J. Appl. Math. Comput. Sci. **18**(3) (2009) 1–15
20. Berge, C.: Graphs and Hypergraphs. Elsevier Science Ltd (1985)
21. Arman, N., Richards, D., Rine, D.: Structural and Syntactic Fault Correction Algorithms in Rule-Based Systems. International Journal od Computing and Information Sciences **2**(1) (2004)
22. Nalepa, G.J., Ligęza, A.: XTT+ Rule Design Using the ALSV(FD). In: RuleApps. (2008)
23. Nalepa, G.J., Ligęza, A.: On ALSV Rules Formulation and Inference. In: FLAIRS Conference. (2009)

Wiki-Enabled Semantic Data Mining – Task Design, Evaluation and Refinement

Martin Atzmueller, Florian Lemmerich, Jochen Reutelshoefer, and Frank Puppe

University of Würzburg,
Department of Computer Science,
Am Hubland, 97074 Würzburg, Germany
lastname@informatik.uni-wuerzburg.de

Abstract. Complementing semantic data mining systems by wikis and especially semantic wikis yield a flexible knowledge-rich method. This paper describes a system architecture of a collaborative approach for semantic data mining. The goal is to enhance the design, evaluation and refinement of data mining tasks using semantic technology. Collaborative aspects are introduced by utilizing wiki technology. We present the components and describe their interaction and application in detail.

1 Introduction

Wikis provide flexible ways for supporting the quick and simple creation, sharing and management of content. Based upon the established wiki-technology, semantic wikis (e.g., [1,2]) enhance this by providing enriched content and features. For example, flexible inline queries and according results that are generated based on these dynamically are such prominent features. While the queries and answers (results) can be flexibly handled by the system, and can usually be formalized as textual content, the wiki system also provides appropriate means for the persistent storage and management of the generated content.

Semantic data mining systems enable the inclusion of a large set of background knowledge, for example, in order to access knowledge services, for selecting the applied data mining methods, or for postprocessing the obtained data mining results. Thus, integrating wikis is a convenient option for semantic data mining systems, since the semantic core components can support the semantic mining features, while the wiki component provides for a convenient front-end and user-management, enables the persistent storage of queries and mining results, and supports their extended annotation.

This paper presents a wiki-enabled approach for collaborative semantic data mining: The semantic data mining system VIKAMINE [3] is combined with the semantic JSPWiki (`http://www.jspwiki.org`) extension KnowWE [1]. We describe the interaction and exchange of query and results data, and the integration of semantic information and knowledge.

The rest of the paper is structured as follows: Section 2 describes the basics of semantic data mining, provides an overview of the presented approach, and describes its implementation. Section 3 concludes with a summary and interesting directions for future work.

2 Method

This section briefly introduces the general semantic data mining approach. After that, we first give a general overview, before we describe the architecture of the proposed approach in detail. Finally, we discuss related work.

2.1 Semantic Data Mining

Semantic data mining can be considered as an approach utilizing formal methods and techniques in order to explicitly integrate data semantics, background knowledge, or reasoning in the mining process. The knowledge is typically represented in a knowledge repository, such as an ontology, or a knowledge base. The main aspect of semantic data mining is the explicit integration of this knowledge into the data mining and knowledge discovery process, where the algorithms for data pre-processing, mining or post-processing make use of the formalized knowledge to improve the overall process. There has been growing interest in this issue, e.g., [4–6], in various domains, especially in the medical domain [4, 7, 8].

With the advent of the semantic web and standardized knowledge representations of semantic web techniques, e.g., the web ontology language *OWL*, utilizing these knowledge representation formalisms for data mining is a promising direction for task design, evaluation and refinement, as discussed below. In the following, we outline the different aspects of semantic data mining, and discuss their implications.

The general data mining process can be structured along the CRISP-DM process model (`http://www.crisp-dm.org`) and consists of the following phases: (a) Business understanding, i.e., understanding the application domain, (b) Data Understanding, i.e., considering the (potential) objects of analysis, (c) Data Preparation, e.g., preprocessing and schema-matching of the data elements, (d) Modeling, e.g., given by concrete mining sessions, (e) Evaluation, i.e., assessment of the mined models, (f) Deployment, i.e., putting the extracted knowledge into action. The semantic data mining approach integrates ontologies in each of the six steps [7]. In the following, we provide examples for each of the phases structured along the dimensions of task design, evaluation, and refinement.

- **Task Design**:
  - In the **Business Understanding** phase ontologies help inexperienced users getting accostumed to the domain, by structuring the relations between the concepts, and explaining the concepts in terms of their properties.
  - In the **Data Understanding** phase, important data elements (contained in the ontology) need to be selected. Then, missing attributes, or redundant attributes can be added or removed from the data set. This can be accomplished by a *data-to-ontology mapping* step [5] where the data elements are mapped to concepts of the ontology, e.g., for integrating heterogenous data.
  - The **Data Preparation** phase is strongly connected to the *Modeling* phase. Depending on the latter, for example constraints on attributes or values can be derived. This concerns constraints on the relations between the attributes, as described in [4], for example, grouping constraints or exclusion constraints for

certain attribute groups that should not be considered. A further possible inclusion of the ontology is given by a more abstract task composition phase, for which the modeling phase can be hierarchically decomposed along the generalization/specialization hierarchies modeled in the ontology. Then, more concise results can potentially be obtained on lower levels, but for efficiency reasons higher levels can be considered first and be used for filtering interesting hypotheses in an earlier stage, cf., [5].

- **Task Evaluation**:
  - During the **Evaluation** phase (of CRISP-DM), the discovered patterns can be interpreted and explained in a structured way using the concepts and/or contained patterns. Various post-processing options are available at this point, cf., [5]. Specifically, due to the data-to-ontology mapping, the discovered patterns can be matched to semantic relations or more complex relations between these. Additionally, such knowledge provides a potential (explaining) context for the discovered patterns. Furthermore, prior knowledge can be compared to the patterns, e.g., for confirming known relations, identifying new knowledge, and/or detecting exceptions and conflicts with formalized expectations. Concerning possible explanations, causal relations can often help in this respect, for validating and confirming discovered patterns, or for their analysis.
  - The **Deployment** phase concerns the integration of the discovered models into the business setting. It is easy to see, that for distributed processing and storage (e.g., on the semantic web) a shared ontology is inevitable. This is especially relevant for deploying results as *semantic analytic reports* (an extension of *analytic reports* [5]), described below. In a late evaluation step, the models/patterns can be tested during their practical application. In that case, the persistent sessions stored in the wiki provide direct access in a collaborative manner.
- **Task Refinement**: The task refinement step is activated after the evaluation step has been performed. It is accomplished either manually using the wiki system – by modifying the textual task description, or by applying formalized knowledge with respect to the applied data mining method. Then, parameters and/or the method itself can be adapted. Refinement is performed according to the results of the *evaluation* phase, so both steps are tightly coupled. Due to the application of the wiki, different persons can collaborate in separate sessions, such that previous results can be included in the refinement of other (related) sessions. Furthermore, previous experiences can be documented using the wiki, for example, explanations/comments by previous users. Furthermore, special refinement and/or evaluation knowledge can be formalized for further improving the respective steps.

2.2 General Overview

As discussed in the last section *semantic data mining* is concerned with the utilization of ontological knowledge and semantic annotations to be used throughout the data mining and knowledge discovery process, similar to *ontology-enhanced* [5] data mining. However, further semantic features are enabled by including a *semantic core* component, e.g., a RDF-Store: Using that, results can be incrementally formalized and provided to the store, while subsequent mining and semantic queries can make use of the collected knowledge. The data mining query, results, and additional knowledge can then be transparently integrated into a *semantic analytic report*: The idea of such reports is based on *analytical reports* [5] that are simple text documents containing the mining results with additional text (which is created by humans). In the semantic setting, we can automatically transform the mining results into a format suitable for the report. Additionally, the content can be enriched using semantic annotations and links between the reports (and background information). The wiki also provides for flexible versioning which is especially useful in a collaborative setting.

The sketched scenario is especially suitable for inexperienced users that are mainly interested in reporting features of a data mining system. Such reports provide high-level access to pre-specified queries that can be evaluated routinely. However, using the wiki query mechanism, such queries can also be formalized in an ad-hoc fashion. Further more detailed reports, analyses and mining sessions can then be implemented using more advanced data mining tools, e.g., by applying the VIKAMINE [3] system.

On the application side, specialized sessions with domain experts, e.g., medical doctors, and data mining engineers can be easily implemented using the collaborative tool. In this context, the proposed approach provides, for example, flexible query formalization, versioning, a history of queries and results, and the potential for knowledge and experience management since the obtained semantic analytical reports can be commented on, and can be linked to other (similar) documents. Further sessions can thus easily build on results of previous sessions, with the same or new participants. For experience management, the wiki can also be combined with a tagging system, e.g., [9].

2.3 Architectural Overview

Due to the limited space, we only provide a brief summary of the architecture of the proposed approach. A more comprehensive discussion and overview is given in [10]. The architecture consists of two core components: The basic wiki system (provided by JSPWiki (`http:/www.jspwiki.org`) is extended by the semantic wiki extension KnowWE [1]. The wiki component provides basic features like editing, versioning, user management, access management and attachment management. Additionally, it directly supports the collaborative aspects of the sketched semantic mining approach. KnowWE itself is designed as a highly extensible minimal core providing basic semantic wiki features like formalization and reasoning. Therefore, for communication with the mining component we designed the connector plugin *KnowWE-RIP* (REST [11] Interface Plugin) that facilitates the connection to the mining web-service. The semantic core component for storage and reasoning is given by a combination of the *Sesame* (`http://www.openrdf.org`) framework and OWLIM. Sesame is a java-based framework with support for storing and analyzing RDF data. OWLIM is a semantic repository with reasoning capabilities that is packaged as a storage and inference layer for sesame. As such, KnowWE integrates a semantic component and contains a connector to the Sesame/OWLIM components for providing the semantic functionality.

We utilize the VIKAMINE [3] system (`http://www.vikamine.org`) for data mining. VIKAMINE features a web-service that can be queried using XML based on a specialized query language. The result (i.e., the answer) is also formulated as XML and can thus transparently be integrated with the wiki.

The semantic mining process is initiated by the user, that is, by formulating a query to the wiki system. Similar to other wiki-systems, the query is provided in the form of an *inline-query* (e.g., [2]): The query is directly entered in textual form. Whenever the wiki page is stored and/or reloaded with a new or modified query the result is requested. In addition, we provide 'extended' inline queries, such both the query and the result (i.e., the 'answer') can also be shown as required. Technically, the query is first transformed to an XML-representation (VPDL, the VIKAMINE *Pattern Description Language*, and then forwarded to the mining engine that produces an result in XML/VPDL format. Finally, this result is re-transformed into human-readable textual form to be displayed by the wiki. However, internally the 'raw' result can be retained by the versioning system of the wiki, such that always the latest result is available and can be cached for efficiency. Therefore, changes, for example, due to an updated dataset, can be easily extracted. The general architecture is shown in Figure 1. The seamless integration of the result presentation enables (inexperienced) users to quickly evaluate the obtained results by themselves and according to the formalized ontological knowledge.

2.4 Related Work and Discussion

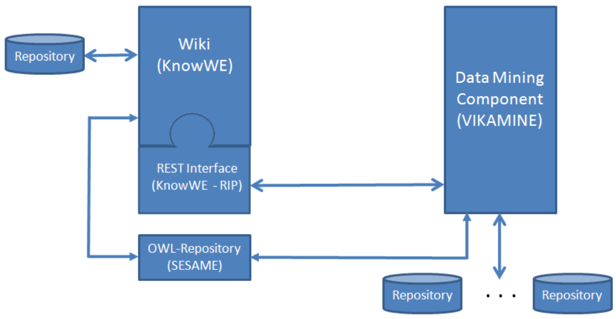

Fig. 1. Semantic Data Mining Architecture

Using ontologies for enhancing data mining has been discussed, e.g., by Svatek et al. [5] and by Antunes [6] in the context of mining association rules. Furthermore, Cespivova et al. [7] and Kuo et al. [8] describe applications in the medical domain. While the application of ontologies is also a focus of the presented approach, the proposed method aims at a more comprehensive integration of semantic information and knowledge. In contrast to the existing approaches, the proposed approach considers a comprehensive *two-way* integration of semantic and data mining methods for semantic data mining, with feedback in both directions. In this way, prior knowledge can be transparently integrated. Using the wiki-support of the presented approach collaborative sessions can be implemented. Furthermore, semantic annotations using the wiki, linking unstructured, semi-structured and structured information is another novel issue with respect to the presented approach. Semantic analytical reports can include semantic annotations at the document level, global tagging, and associated query – data mining results that are stored in the semantic store and thus provide powerful options for knowledge-rich applications.

3 Conclusions

In this paper, we have presented an approach for collaborative semantic data mining. We discussed the considerations of task design, evaluation and refinement in the context of semantic data mining. Additionally, we have presented an overview on the approach and have described its architectural considerations in detail, utilizing the VIKAMINE system for semantic data mining with a connector to a wiki system.

For future work, we aim to extend the mining approach towards text mining and information extraction, e.g., [12]. This opens up further potential for incremental knowledge refinement, discovery and semantic annotation.

Acknowledgements

This work has been partially supported by the German Research Council (DFG) under grant Pu 129/8-2.

References

1. Reutelshoefer, J., Haupt, F., Lemmerich, F., Baumeister, J.: An Extensible Semantic Wiki Architecture. In: Proc. 4th Workshop on Semantic Wikis - The Semantic Wiki Web. (2009)
2. Krötzsch, M., Vrandecic, D., Völkel, M., Haller, H., Studer, R.: Semantic Wikipedia. Web Semantics: Science, Services and Agents on the World Wide Web **5**(4) (2007) 251 – 261
3. Atzmueller, M., Puppe, F.: Semi-Automatic Visual Subgroup Mining using VIKAMINE. Journal of Universal Computer Science (JUCS), Special Issue on Visual Data Mining **11**(11) (2005) 1752–1765
4. Atzmueller, M., Seipel, D.: Using Declarative Specifications of Domain Knowledge for Descriptive Data Mining. In: Proc. 18th International Conference on Applications of Declarative Programming and Knowledge Management, Berlin, Springer Verlag (2008)
5. Svátek, V., Rauch, J., Ralbovský, M.: Ontology-Enhanced Association Mining. In: Semantics, Web and Mining. Volume 4289 of LNCS. (2005) 163–179
6. Antunes, C.: Onto4AR: A Framework for Mining Association Rules. In: International Workshop on Constraint-Based Mining and Learning (CMILE 2007), Warsaw, Poland (2007)
7. Cespivova, H., Rauch, J., Svatek, V., Kejkula, M.: Roles of Medical Ontology in Association Mining CRISP-DM Cycle. In: Proc. ECML/PKDD 2004 Workshop on Knowledge Discovery and Ontologies, Pisa, Italy (2004)
8. Kuo, Y.T., Lonie, A., Sonenberg, L., Paizis, K.: Domain Ontology Driven Data Mining: A Medical Case Study. In: DDDM '07: Proceedings of the 2007 international workshop on Domain driven data mining, New York, NY, USA, ACM (2007) 11–17
9. Atzmueller, M., Haupt, F., Puppe, F.: Knowta: Wiki-Enabled Social Tagging for Collaborative Knowledge and Experience Management. In: Proc. 2nd International Workshop on Design, Evaluation and Refinement of Intelligent Systems (DERIS). (2009)
10. Atzmueller, M., Lemmerich, F., Reutelshoefer, J., Puppe, F.: An Extensible Architecture for Wiki-enabled Semantic Data Mining. In: Technical Report. University of Wuerzburg. (2009)
11. Fielding, R.T., Taylor, R.N.: Principled Design of the Modern Web Architecture. ACM Transactions on Internet Technology **2**(2) (2002) 115–150
12. Atzmueller, M., Kluegl, P., Puppe, F.: Rule-Based Information Extraction for Structured Data Acquisition using TextMarker. In: Proc. LWA 2008 (Knowledge Discovery and Machine Learning Track), University of Wuerzburg (2008)

The Patterns Experience Evaluation Program

Klaus P. Jantke

Fraunhofer IDMT, Children's Media Dept.,
Hirschlachufer 7, 99084 Erfurt, Germany
e-mail: klaus.jantke@idmt.fraunhofer.de

Abstract. Patterns in the experience of game playing are key to understand the impact of digital games. The author's approach starts from a firmly based pattern concept which is introduced first. Next, the relevance of the concept is exemplified by means of a few digital games studies. Finally, after the concepts has been made clear and the occurrence of the concepts in practice has been demonstrated, the author's evaluation program is shortly layed out. This program shall be implemented in forthcoming experiments.

1 Formal Approaches Meet the Beauty of Media Experience

Understanding digital games comprises an understanding of fun when playing any game. To the design of serious games that work drastically better than what we have today, being in control of fun seems to be crucial. Raph Koster has identified patterns in the experience of game playing as a key to understand fun [1].

But is it realistic to approach the richness and beauty of media experience wielding formal tools? Classical music performance may serve as a truly challenging case (see [2] for a comprehensive treatment).

It is one of the greatest challenges to computer scientists to tell something essential–using their own concepts and terminology–to specialists of the performing or the virtual arts. Questions appear sometimes so easily: What makes the difference between piano play of Daniel Barenboim when compared to Mitsuko Uchida? Good question. But answers . . . ? Imagine computer scientists to be able to give at least some asnwer understandable, at least to some extent. What would be a criterion of success, a measure for the quality and sustainability of their answer?

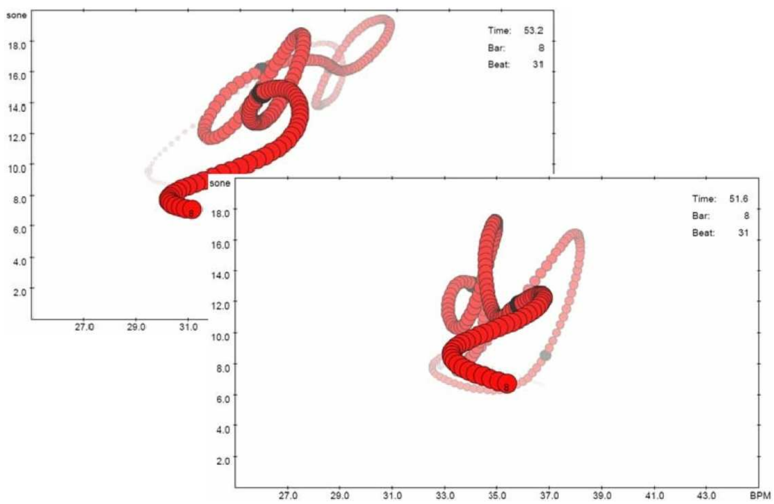

Fig. 1. Representation and Visualization Concepts of [3] – Worms of Performance

Clearly, they should demonstrate that their answer may be exploited somehow operationally. Suppose an answer is given, let's use it for the computerized control of a piano which plays more like Barenboim than like Uchida or vice versa. It works!

There has been found a rather simple descriptional language just using loudness and speed of play (measured in beats per minute) to describe game playing over time [3]. If the two parameters of loudness and speed are recorded and visualized in a 2-dimensional diagram, say by read dots, the progress over time shows as the movement of the dots. If dots stay for a while in the diagram, perhaps, just shrinking in size, the result looks like a worm moving through the 2-dimensional space.

When playing the same piece of music, Uchida's worm (in the left diagramm of figure 1) moves differently from Barenboim's worm (right in the front). Features such as bending and accelerations may be extracted and used for reproducing sound resembling a particular artist's play [2].

Now, that we have seen that it works for music, let's try it for digital game playing as well. In a recent application project with Ubisoft, patterns are investigated.

2 An Intuitive Approach to Patterns in a Study Experiment

The author has developed a particularly simple track game named GORGE which mostly serves as a research tool to study elementary problems of media impact. There are several dozens of slightly verying implementations for different purposes.

In GORGE, although the game is extremely simple, players may choose from a spectrum of tactics varying from an altruistic through a widely ignorant to an aggressive and defecting behavior. Non-player characters (NPCs, for short) may be tuned to the one or the other characteristics. When human players meet those NPCs, they experience quite different developments of stories during game play. In [4] there has been elaborated the hypothesis that setting appropriate characters of NPCs may trigger the emergence of substantially different stories. These authors' central hypothesis is: *Character is crucial to interactive storytelling.*

But how does it happen? GORGE serves as a tool to find answers by example. On the board, all players have to move from the start area close to the top to the end of track at the bottom of the window. There is a target area of 6 subsequent fields on which the score is the higher the closer the players are to the very end of the area. When players have rolled a dice, they may choose among their 4 pawns which one to move. A variety of conflicts may arise. The most interesting problem is to cross a gorge. Gorges can only be passed by one player's pawn, if another pawn stepped down into the gorge before. The precondition for the quite altruistic act of stepping down into a gorge is that two pawns meet immediately before the gorge to form what is called a roped party. One of them may climb down.

The game GORGE is currently in use within a series of qualitative research experiments[1] with subjects of an age ranging from 13 to 18. It was a first surprise that even 18 years old young male subjects used to play COUNTERSTRIKE: SOURCE found it interesting to play GORGE and to set up NPC characters to drive game playing experience.

The subjects implicitly identified patterns in game playing experiences and found it exciting to adjust NPC characters such that instances of patterns occur repeatedly.

These subjects have been slightly later introduced to the point & click adventure SECRET FILES: TUNGUSKA (see below) in which they found instances of patterns self-reliantly.

The author's approach has been motivated by Bruce Philips call for concepts describing game playing experience [7].

[1] The author's experiments have been supported by his colleagues Swen Gaudl, Denise Lengyel, Melanie Meder, Alexandra Neumann, and Claudia Staats. The social-sciences grounding (see [5, 6], e.g.) of the experiments has been supervised by Imke Hoppe.

3 Introduction to Pattern Concepts of a Varying Generality

When you, in playing a digital game such as GORGE, repeatedly experience instances of a behavior such as [building a roped party] – [stepping into the gorge] – [passing the gorge], e.g., this results in a certain atmosphere of cooperation. Continuously struggling and fighting results in a different atmosphere. There are paramount cases in which the atmosphere perceived by a human player strongly correlates with the repeated appearance of some structural regularities of game play.

These structural regularities shall be called *patterns* and whatever we recognize of such a pattern is called an instance of this pattern. Note that you never see a pattern, you only see instances. And when you see an instance, it might be an instance of several different patterns.

There is a large variety of approaches to patterns in science and engineering [8, 9, 10, 1]. Very roughly speaking, patterns represent generalities of structures which may show in different instances differently. Therefore, scientific usage of the pattern idea requires structural representations [11].

What humans experience throughout media reception is highly individual, rarely explicit and typically not formal. Science always means abstraction [12]. The author's present research is based on hierarchically structured abstractions of the activities that take place when playing a digital game. There has been developed an original approach to patterns in game playing expressed in logical terms [13].

For more clarity, a few simple notions and notation will be introduced.

Assume we have a particular game under consideration. Whenever a reference is necessary, this game will be named G. In the most simple cases, there is a clearly distinguished set of actions that may be performed when playing the games. Those actions may be performed by a single player, by several players, or by the digital game, i.e. by a computer system. What is taken into account depends on the scientific interest driving our investigations. When a decision is made, M is used to denote the set of all considered actions-M is chosen as reference to the term "move"-in the game.

Playing a game means interacting intensively and extensively. One action follows the other. Abstracting from many details, one may represent game playing by sequences of actions from M. In theoretical computer science, it is common to denote the set of all possible finite sequences of elements from some set M by M^*. For theoretical reasons, the empty word ε is enclosed.

Given a game G and the actions M of interest, M^* is completely specified. But what is the game play we are interested in?

To keep it short and simple in the present publication, the concepts introduced focus mostly those games in which it makes sense to speak about a completed game play. This applies to most simulation and sports games, to all jump 'n' run games, to all point & click adventures, and to all games that tell a story. Some sequences of actions establish a completed game play whereas others do not.

Given a game G and the actions M of interest, the term $\Pi(G)$ denotes the subset of M^* of all those sequences that represent some completed game play, $\Pi(G) \subseteq M^*$.

$\Pi(G)$ consists of sequences of symbols from M. Every sequence $\pi \in \Pi(G)$ describes what happens during a particular game play. The choice of M reflects our decision about the granularity of game play descriptions. Two different sequences $\pi_1, \pi_2 \in \Pi(G)$ of game playing experience may describe game plays of different players or of one game player at different occasions. For interesting digital games, $\Pi(G)$ is usually infinite.

The conceptualization of this section provides a firm background of the present investigations in formal language theory [14]. The stage is set to see *patterns* as *properties of substrings* of a given string. Any string that has the corresponding property is called an *instance* of the pattern under consideration [13].

4 Patterns of Game Playing Experience – Search for Impact

Following Raph Koster's outline [1] based in his own game design practice, playability–surely a key issue of digital game design and development–depends on the player's ability to get the game mechanics under control. This means learning.

In the quite conventional, but rather fascinating point 'n' click adventure named Shadow of Destiny, your avatar is frequently stabbed to death or murdered in a different way. Successful game play includes learning of how to overcome these problems. There is a general pattern of problem solving: Travel back in time and remove necessary preconditions of your murder.

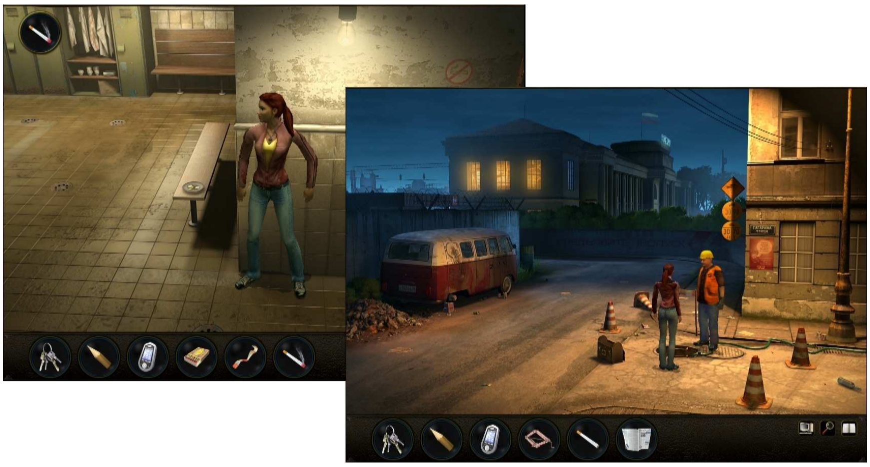

Fig. 2. Secret Files: Tunguska – Two Instances of a Game Play Dominating Pattern

In the game Secret Files: Tunguska you play the female avatar Nina seen in both screenshots of figure 2 above. Nina wants to find her father who somehow mysteriously disappeared. There are several obstacles on Nina's way. It turns out that there is a general pattern of overcoming these difficulties.

A first instance shows as follows. You have to go to some railway station which is heavily guarded. There is no way to get in. Next to the railway station you find a worker at a manhole (right screenshot in figure 2). Successful game play proceeds as follows: (i) You need to understand that you have to lure away this particular person. (ii) You need to gather some information that might be useful. (iii) You need to set a trap based on the information you could acquire. (iv) If you did well, the person is somehow brought out of your way. (v) You can proceed in game playing. In the particular case under consideration, you send Nina into the manhole.

Instances of this pattern show about a dozen times in Secret Files: Tunguska.

Anno 1404 is the newest game of the quite successful Anno series. It is highly complex and seems to rely on almost uncountably many patterns. A closer look reveals that there are already a few quite elementary patterns that may be crucial to the acceptance resp. rejection of the game by particular recipients.

In this strategic game your core activities are setting up and developing settlements as well as fighting battles; many of these activities are triggered by requests from persons who ask you for help. Everyone playing the game has to respond to those requirements, i.e. to get engaged in quests to solve them.

How do the quests, their occurence and their relative mutual positions influence the experience of game play? What about nesting vs. sequentiality? Is there an optimal or a maximal depth of nesting relevant to fun of playing the game? What about overlapping quests? Does overlapping confuse the players? Or is overlapping an indication of freedom to decide which of the problems to solve first?

Being in control of a game means, somehow, being able to answer those questions.

5 The Patterns Evaluation Problem in Playing Experience

Given any digital game, the study of the experience and perception of patterns in game playing is just one approach among many others.

Here is the core approach to the *evaluation of experiencing patterns*. We assume that a particular game G is given.

When we succeed in a process like the one on display on figure 3, we have–according to our notations and, in particular, according to the chosen level of granularity–found the occurrence of patterns in human game playing. Furthermore, we may have recorded game play (videos, e.g.) documenting the players' reaction to what happened in the game play itself. So, we are ready for an evaluation. The pre-evaluation process does provide the data.

Did the players recognize what we have considered to be the instance of a pattern? Did they perceive the pattern instances consciously? Did the players react to the occurrence? How did they react?

For particular classes of patterns and their instances, concerning violence, e.g., one may ask more specific questions.

Last but not least, there will be surely a need for feedback in the process of figure 3. In dependence on some outcomes of experiments, there may become explicit a need of changing the expressive tools such as M or the modification of the pattern(s) under consideration. It is an exciting problem in its own rights to find an appropriate level of abstraction supporting the current research interest in an optimal way.

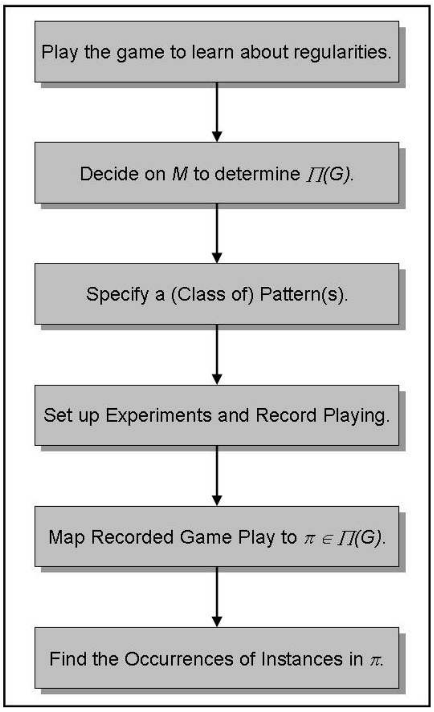

Fig. 3. Pre-Evaluation Process Model

To which of the problems under consideration are qualitative or quantitative methods appropriate? In which cases (in dependence on the game or independence on varying patterns for a fixed digital game) do we need a particular combination of qualitative and quantitative methods?

The author's *pattern experience evaluation program* is an attempt to systematize all the issues sketched above.

6 The Perspective of Learning and Knowledge Discovery

The few words in this closing chapter are surely going beyond the limits of the present publication. However, the author is finding it worth to widen the horizon and pointing to the quite enormous potentials of interdisciplinary communication and, perhaps, even cooperation.

Let us turn the perspective outlined above and let us drop the pre-evaluation process model on display in figure 3. Let us take, instead, a player-centered stand.

We are observing human game playing behavior, we are recording it in different forms such as log files or videos, e.g. From this source we extract effects and affects of interest–moments when the playing subjects are frightened or periods of playing time when they appeared particularly excited, concentrated or obviously bored, e.g.

Now we map the recorded game playing experience or, at least the parts of interest, to formal language representations over some alphabet M as done before. For certain effects of interest, we have–formally speaking–some recorded game play $\pi \in \Pi(G)$ in which we can mark certain substrings $\pi_1, \pi_2, \ldots\ \pi_n$ which are of potential interest. $\pi_1, \pi_2, \ldots\ \pi_n$ are hypothesized of being instances of a currently unknown pattern φ. It arises a pattern inference problem as studied in [9]. However, our underlying data are a bit more vague than in the cases studied by Dana Angluin. iven any of the instancesπ_i, we are not definitely sure about the begin and the end of π_i in π.

This difficult constellation does clearly call for the exploitation of knowledge discovery experience or, even better, for an in-depth cooperation with experts in the field of knowledge discovery.

7 Acknowledgement

The first version of the present manuscript has been submitted when the author has been visiting the Meme Media Lab. at the Hokkaido University Sapporo, Japan.

The present research and development has been partially supported by the Thuringian Ministry for Culture within the project iCycle under contract PE-004-2-1.

References

1. Koster, R.: A Theory of Fun for Game Design. Scottsdale, AZ, USA: Paraglyph Press, Inc. (2005)
2. Widmer, G.: In search of the Horowitz factor. In Lange, S., Satoh, K., Smith, C.H., eds.: Proceedings of the 5th International Conference on Discovery Science. Number 2534 in LNCS, Springer Verlag (2002) 13–21
3. Dix, S.: On the analysis of musical expression in audio signals. In Yeung, M.M.Y., Lienhart, R.W., Li, C.S., eds.: Storage and Retrieval for Media Databases 2003, SPIE - International Society for Optical Engine. (2003)
4. Gaudl, S., Jantke, K.P., Woelfert, C.: The good, the bad and the ugly: Short stories in short game play. In: Proceedings of the 2nd International Conference on Digital Storytelling. (2009)
5. Bosshart, L., Macconi, I.: Media entertainment. Communication Research & Trends **18** (1998) 3–8
6. Burkart, R.: Kommunikationswissenschaft: Grundlagen und Problemfelder; Umrisse einer interdisziplinären Sozialwissenschaft. Böhlau (2002)
7. Philips, B.: Talking about games experience – A view from the trenches. interactions (2006) 22
8. Alexander, C.: A The Timeless Way of Building. New York: Oxford University Press (1979)
9. Angluin, D.: Finding patterns common to a set of strings. J. Computer and Systems Science **21** (1980) 46–62
10. Björk, S., Holopainen, J.: Patterns in Game Design. Hingham, MA, USA: Charles River Media (2004)
11. Grünvogel, S.M.: Formal models and game design. Game Studies **5** (2005)
12. Popper, K.: Logik der Forschung. Tübingen (1934)
13. Jantke, K.P.: Patterns in Digital Game Playing Experience Revisited: Beiträge zum tieferen Verständnis des Begriffs Pattern. Diskussionsbeiträge 33, TU Ilmenau, Inst. Media & Comm.-Sci. (2008)
14. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation. Addison-Wesley (1979)

This article was processed using the LaTeX macro package with LLNCS style

Author Index