# Table of Contents

Preface ................................................................. 3  
    *Kerstin Bach and Michael Meder*

## Long Papers

**InKnowE: Enhanced Product Innovation by Application of Knowledge Management, Knowledge Networks, Business Intelligence, Web and Social Analytics** ................................................................. 5  
    *Mareike Dornhöfer and Madjid Fathi*

**A Document-centered Authoring Approach for Ontology Engineering** .... 13  
    *Jochen Reutelshöfer, Joachim Baumeister, Georg Fette and Frank Puppe*

## Short Papers

**Modeling the Structure of Spreadsheets** ........................................ 21  
    *Christian Liguda*

## Resubmissions

**iTree: Skill-building User-centered Clarification Consultation Interfaces** .. 26  
    *Martina Freiberg and Frank Puppe*

**Confidence in Workflow Adaptation** ............................................. 31  
    *Mirjam Minor, Mohd. Siblee Islam and Pol Schumacher*

**Semantic Alliance: A Framework for Semantic Allies** ....................... 32  
    *Catalin David, Constantin Jucovschi, Andrea Kohlhase and Michael Kohlhase*
Preface

FGWM-2012: Workshop on Knowledge and Experience Management

Kerstin Bach
Competence Center CBR, German Research Center for Artificial Intelligence DFKI GmbH and University of Hildesheim, Germany
kerstin.bach@dfki.de

Michael Meder
DAI-Lab
Technische Universität Berlin
Germany
meder@dai-lab.de

Knowledge and Experience Management

The use of existing experience based upon a shared knowledge is one of the main human approaches in problem solving. Knowledge and Experience Management aims to enable and support the conversion of this basic human approach into intelligent systems. The approaches and techniques used to provide and manage the knowledge and to gather and reuse experience are the subjects of this workshop. To be able to use experience the first set of open questions lies within the field of knowledge acquisition. In the view of the development of the Web 2.0 and especially social networks a wide field of sources for experience is given. Additionally there exist rich web-based sources of knowledge in the form of Linked (Open) Data. The acquisition of knowledge and experience can also reach into the sources of, for example, best practices used in business or text documents. However, the acquisition from all of these sources bears some challenges regarding a variety of open questions. These questions are given by the problems of proper and efficient knowledge formalization especially the design of efficient forms of knowledge representation for a given task, like for example the design of ontologies. Further the extraction of knowledge and experience from complex data poses a complex problem too. To ease these problems the workshop examines new approaches in using semantic web technologies to aid the knowledge and experience exchange and acquisition. Another set of problems occurring in the process of reusing knowledge and experience is given by the tasks of fast and accurate retrieval of knowledge and experience.

The use of knowledge and experience in collaborative environments like e-business or e-government involves questions about the authoring, maintenance and exchange of knowledge and experience. Those questions deal with the secure storage of knowledge for example in cloud storage environments, the proper integration of existing business processes into knowledge management systems and the possibilities to capture knowledge just in time within such e-collaboration systems.

The mentioned increase in variety and number of sources for knowledge and experience and the rapid development of mobile technologies induce a set of advanced questions regarding knowledge management and the gathering and use of experience. These advanced topics the workshop deals with, are the enhancement of knowledge formalization approaches to enable explanation capable and context sensitive systems. Further the challenges a foreseeable development of ambient intelligent environments pose to just in time knowledge capture and retrieval in such a highly mobile environment. These challenges can be met by new approaches as of agile knowledge and experience acquisition and agent based systems for knowledge management enriched with distributed on demand knowledge retrieval techniques. These new approaches are to be examined and discussed within the workshop.

FGWM 2012

The workshop is a part of the workshops organized at the LWA 2012 conference in Dortmund, Germany (namely the “German Workshop on Knowledge and Experience Management”). The objective of the workshop is to provide an opportunity for exchanging ideas related to the application of various approaches and techniques within the field of knowledge and experience management. The workshop aims at providing a forum for the discussion of recent advances in this research field and to offer an opportunity for researchers and practitioners to identify new promising research directions. The workshop also aims to provide a platform for young researchers to present their work and receive feedback from the knowledge and experience management community.

This year we received 11 submissions and after a peer review, we were able to accept 6 contributions for presentations. Among the accepted papers we have one long paper by Mareike Dornhöfer and Madjid Fathi discussing knowledge management driven product innovation, one short paper by Christian Liguda discussing an approach for representing underlying spreadsheet models and four resubmissions of papers that have been presented on different international conferences during this year. Further, we are very happy that Benno Stein gives an invited talk on “The Web as a Corpus”.

Like in previous years, the FGWM will have two joint session with the Information Retrieval (FGIR) and the Knowledge Discovery and Similarity (KDML) workshop, which are both co-located at the LWA.

2012 Program Committee

- Andreas Abecker, disy Informationssysteme GmbH
- Klaus-Dieter Althoff, University of Hildesheim
- Joachim Baumeister, denkbares GmbH Würzburg
- Ralph Bergmann, University of Trier
- Simone Braun, Forschungszentrum Informatik (FZI)
- Jörg Cassens, University of Lübeck
- Andrea Kohlhase, Jacobs University
- Christoph Lange, Jacobs University
- Andreas Lommatzsch, Technische Universität Berlin
In closing this preface, we would like to recognize all the colleagues who helped to review the submissions and to guarantee the quality of the papers that have been included. And last but not least, we would like to thank Prof. Dr. Katharina Morik, Nico Piatkowski, Hendrik Blom and Tobias Beckers for hosting this year’s LWA at the University of Dormund, Germany.

September 2012
Kerstin Bach & Michael Meder
InKnowE: Enhanced Product Innovation by Application of Knowledge Management, Knowledge Networks, Business Intelligence, Web and Social Analytics

Mareike Dornhöfer, Madjid Fathi
Universität Siegen
D-57068 Siegen, Deutschland
m.dornhoefer@uni-siegen.de, fathi@informatik.uni-siegen.de

Abstract

The given paper focuses on the enhancement of product innovation by means of technologies and disciplines like knowledge management, especially knowledge networks, business intelligence, web and social analytics. Based on the connectivity of product consumers via the internet, it is no longer possible for companies to develop their products from scratch without considering the public customer feedback. The innovation process needs input not only from employees of the company, who are tasked with creating a new product or product generation, but also by applying different knowledge sources, indicators or analyses. Therefore the paper proposes a concept idea for leading the innovation process even closer to the customers needs by applying technologies from the disciplines and methods mentioned above in addition to innovation management methods.

1 Introduction

The given paper proposes a concept called InKnowE for the enhancement of product innovation through applying methods from the fields of knowledge management, especially knowledge networks, business intelligence (BI), web and social analytics. In this context the aspect of open innovation, communities for innovation or innovation labs will be breached as well. The objective of InKnowE is to improve the classical innovation process detailed e.g. in [Vahs and Burmester, 2005] or [Müller-Prothmann and Dörr, 2006] and to make a product innovation even more market and consequently customer oriented. Today push/technology/company initiated product innovation and pull/market/customer initiated innovation have to be brought together by companies to be successful on the customer market [Bergmann and Daub, 2008], [Nerdinger et al., 2010], [Völker et al., 2007] and to make a product innovation even more market and consequently customer oriented.

Enterprises, which are selling their products via internet, involuntarily have to face the challenges, as well as the advantages of positive or negative customer feedback e.g. regarding the characteristics, application or durability of their products. Sales platforms (e.g. Amazon, www.amazon.com), social networks (e.g. Facebook, www.facebook.com), product comparison pages or blogs, offer a huge amount of data, consisting of selling numbers, consumer profiles, target groups, product evaluations, customer feedback, hints for improvement and even price comparisons with fellow competitors. Evaluating and using these information may lead to more consumer oriented products, more positive feedback and less consumer complaints, which consequently improves the companies image concerning their product quality.

In this context, the proposed concept of IKnowE faces the challenge of bringing results from feedback analyses together with e.g. strategic goals, product portfolio or the innovation budget of the company. InKnowE focuses on integrating the technologies mentioned in the first paragraph into one framework to support the decision making process for a new product innovation idea and the consequent realisation. Figure 1 depicts a mind map about the product innovation environment in a company and the possible influencing factors.

In the following section the disciplines and technologies applied in IKnowE are introduced and summarized, before section three gives a literature study of current interactions between the different disciplines. Afterwards section four analyses the product innovation context based on the aforementioned mind map and introduces the concept idea for integrating the disciplines into one framework. At the end section five concludes the paper and gives an outlook on the next steps.

2 InKnowE disciplines

This section introduces the disciplines relevant for the InKnowE framework proposed in this paper for the enhancement of product innovation through means of knowledge management especially knowledge networks, as well as features from business intelligence and results from web and social analytics.

2.1 Innovation management

In his work about how the worlds best companies manage their innovation machine, Wentz [Wentz, 2008] defines four important factors for innovation: 1) innovation has to be consistent with the companies strategy, 2) there has to be a synergy with the core competencies of the company, 3) there has to be a success potential for the approached field of innovation and 4) if there are different fields of innovation, they have to have synergies with each other. The focus of this work will be product innovation, although the proposed concept may also be transferred to a decision component for service innovation.

Product innovation is essential for the success of a manufacturing company on the market. In this context Vahs and Burmester [Vahs and Burmester, 2005] define product
innovation as a process of creating new material and immaterial products, which satisfy customer needs and have a positive impact on the companies' profits. There are different strategies for companies, how to approach the release of a new innovation on the market. Bergmann and Daub [Bergmann and Daub, 2008] diversify the different forms of innovation strategies, e.g. pioneer, second to the market or me-to-strategy.

To be even closer at the customer requirements, companies realized that the classical form of only inhouse innovation teams without the input of external sources e.g. experts or customers is in most branches no longer sufficient enough to be successful on the market. Therefore the methods of open innovation, community based innovation [Füller et al., 2004], [Bretschneider et al., 2012] and innovation labs emerged, aided by the further development of the internet. Open innovation allows the customer to give input towards the design, development and production of new, or enhancement of existing products. Reichwald and Piller [Reichwald and Piller, 2007] define the process of open innovation as a value-added cooperation between companies, external experts and consumers for the development of mass products. In the same context the aspect of product individualization and mass customization of products (e.g. individual customized shirts) is addressed. Howaldt, Koop and Beerheide [Howaldt et al., 2011] go even one step further. They declare the Innovation management 2.0 in reference to the Web 2.0 and the Enterprise 2.0, thus hinting the importance of web technologies for today's innovation management. The concept of community based innovation [Füller et al., 2004] or virtual communities for innovation [Bretschneider et al., 2012] is another approach to be considered. "Virtual Communities for Innovation are becoming increasingly popular as platforms for firms to engage customers in generating new ideas" [Bretschneider et al., 2012]. Therefore the input customers provide for improving or developing new ideas via web communities should be part of the innovation management processes of a company, just like the classical inhouse innovation or research department. Another type of community of innovation are the living labs. "Living labs offer a collaborative partnership framework in which user-centred, innovation activities can take place" [Mulvenna et al., 2010]. They may be established for different innovation purposes and branches by research institutions, civic institutions or companies. From these short definitions and examples, it is apparent that communities for innovation as well as innovation labs are part of the open innovation movement.

2.2 Knowledge management including knowledge networks

There is a lot of literature regarding the field of knowledge management, knowledge work, knowledge workers as well as knowledge networks available today. Based on the definition of Maier, Hädrich and Peinl [Maier et al., 2005] "Knowledge Management is defined as the management function responsible for regular (1) selection, implementation and evaluation of knowledge strategies (2) that aim at creating an environment to support work with knowledge (3) internal and external to the organization (4) in order to improve organizational performance. The implementation of knowledge strategies comprises all (5) person-oriented, product-oriented, organizational and technological instruments (6) suitable to improve the organization-wide level of competencies, education and ability to learn."

Knowledge work summarizes different characteristics, which differ from traditional work and are e.g. accumulated by Maier, Hädrich and Peinl [Maier et al., 2005]. According to the authors, knowledge work is among other characteristics communication, cooperation and network-oriented, it uses semi-structured data, uses different tools like document/content management systems, experience data bases, newsgroups, mail folders or other groupware. In the context of enhancing innovation, the collaboration and communication aspect of knowledge work is an essen-
tial fact. In a former work, Dornhöfer, Holland and Fathi [Dornhöfer et al., 2012] propose a knowledge-based innovation detection and control framework. One aspect of their concept model is the set up of a collaboration community on different levels. This collaboration community is one form of a knowledge network or community for innovation (→ 2.1).

Back et al. [Back et al., 2005] define knowledge networks "as social networks between knowledge players, which allow the creation and transfer of knowledge among individuals, groups, organizations, and between hierarchical levels." The authors furthermore stress the importance of the interconnection of business and knowledge networks. Knowledge networks have to be integrated in the knowledge management and business processes, as otherwise they are lacking up-to-date input and do not contribute with proper results to the knowledge management process. Based on a world of web technologies, the working tools of knowledge networks lead directly to the field of social networks and communities.

2.3 Business intelligence

Since the mid 1990s the topic of business intelligence (BI) and its components (e.g. data warehouse) came into focus and is still under constant development and expansion. Business intelligence is a term, which doesn’t seem to have an exact definition in a mathematical sense. This opinion is concluded from different definitions as in [Kemper et al., 2010], [Gluchowski et al., 2008], [Chamoni and Gluchowski, 2006], [Bauer and Güzel, 2009] or [Gabriel et al., 2009]. A definition which somehow summarizes different approaches is the one of Gluchowski, Gabriel and Dittmar [Gluchowski et al., 2008]. They define BI as an approach, concept and application environment for the purpose of generating, representing and analyzing data to support decision making on a management level. Furthermore they define how close certain methods like OLAP, ad-hoc reporting, balanced scorecard or data warehouse belong to an BI-system. Based on their approach OLAP and MIS/EIS (Management Information System/Enterprise Information System) are the core components of a BI-system [Gluchowski et al., 2008].

Kemper, Mehanna and Baars [Kemper et al., 2010] apply this approach as well. Additionally the authors define a BI framework consisting of a data layer, a data processing layer (e.g. in form of a data warehouse), an information generation and distribution layer and a presentation layer in the form of a BI-portal. The information generation and distribution layer allows the integration of knowledge and information management methods. BI-tools as well as standard text, calculation and presentation tools are widely applied (commercial) tools and core components for decision making of managers in today’s companies. BI applies techniques for merging, calculating ad visualizing product sales numbers. With hindsight to the IKnowE concept proposed in section → 4, these characteristics of BI integrate the commercial or business point of view into the knowledge flow for evaluating new innovation ideas. Based on existing sales numbers it is possible to deduce top sellers, trends or selling failures and consequently change business strategies for new innovations.

2.4 Web and social analytics

Due to the strong development towards e-commerce it is essential for managers to consider not only commercial numbers, but web analytics results in their strategies and decisions as well. Web analytics itself is a technology for gathering data and establishing optimization recommendations or target group analyses with the help of this data [Reese, 2008]. The core components of web analytics are client based page tagging mechanisms, server based logging, multi-variant testing, online surveys, interviews or an observation how a user interacts with a concrete website [Hassler, 2012]. The gathered data allows the analysis of web-traffic, to some extend user origin and behaviour as well as the effectiveness of marketing on the web or even in social networks. In this context the analysis of a company’s own web shop or specific product sites are especially interesting for a company. There are different commercial or open web analytics solutions available on the market. Additionally commercial sales platforms like ebay [ebay, 2012] or Amazon [amazon, 2012] offer sales reports and evaluations for their sellers.

Social analytics is a special form of web analytics focused on the usage behaviour and content in social media platforms. Today social media is not only used as a tool for communication between private persons, but also by companies as a marketing tool, form of a news letter or direct customer contact channel. Hassler [Hassler, 2012] defines two types of social analytics: 1) social web analytics for measuring the social web activities of the company, 2) social media analytics for measuring social media content related to the company or the companies products, but generated through persons outside the company. The first type of analysis e.g. considers the activities on the applied social networks, while the second one evaluates the companies or brands impact on the web and in social media [Hassler, 2012].

Before Web 2.0, the term social network analysis was related to the analysis of the social interaction between different stakeholders or social circles. "Social Network Analysis (SNA) is the study of social relations among a set of actors. The key difference between network analysis and other approaches to social science is the focus on relationships between actors rather than the attributes of individual actors" [Mika, 2007]. Coulon for example executed a study about social network analysis in innovation research [Coulon, 2005]. In the context of online social networks or social media not only the personal interactions, but also the content are part of the analysis process, thus the term of social network data analytics (e.g. [Aggarwal, 2011]) is also common. The large amount of content in social networks, allows the application of social analytics in combination with linguistic techniques like text mining or sentiment analysis. The combination of these techniques visualizes opinions about a certain product in a condensed form and gives indicators about the acceptance of the product on the market. On a larger scale the image of the company can be made visible as well.

Aggarwal [Aggarwal, 2011] defines two "primary kinds of data which are often analyzed in the context of social networks:” 1) "Linkage-based and Structural Analysis” and 2) "Adding Content-based Analysis”. The first form of analysis detects and analyzes the structure including the important nodes and links of a social network. The second method focuses on the analyses of the user content and the associated media files. According to the author, research areas like statistical analysis of social networks, data and text mining, community detection, link prediction or social influence analysis are part of social network data analytics.
Scott [Scott, 2011] describes the potential of social analytics and especially data mining techniques advantageous for "large-scale data sets of the kind that have not generally been possible to investigate using conventional social network analytic techniques" before. Based on these definitions the line between social web, data and media analytics and social network analysis (in online contexts) blurs, therefore all of the aforementioned variants will be called social analytics in the following sections. The results from web and social analytics will be integrated into the analytic flow of the IKnowE concept proposed in section $\rightarrow 4$ to support funded decision making about new innovation ideas. Especially interesting in this context is the customer feedback about existing products and their lacks or negative points.

3 Literature study of current interactions between the different disciplines

This section gives an introduction of literature and project studies, where two or more of the above mentioned disciplines are interconnected, thus creating the context of the concept proposed in the next section. Stegebauer and Häußling [Stegebauer and Häußling, 2009] define a knowledge based innovation management, which supports the innovation process as well as the knowledge exchange on the different levels between individuals, groups or organisations.

The research of Gentsch [Gentsch, 2001] moves into a similar direction. He analyses the interaction of innovation and knowledge management, with a special focus on knowledge discovery from text for enhancing innovation processes.

Völker, Sauer and Simon [Völker et al., 2007] focus their work on the interaction aspects of innovation and knowledge management as well. They propose a matrix organization, where the innovation management is the vertical dimension, while knowledge management is the horizontal one. Next to different knowledge oriented models, they also consider the organizational requirements of a company in the context of innovation.

Zhou and Uhlramer [Zhou and Uhlramer, 2009] executed a study, where they evaluated the impact of enquiring external knowledge for the innovation behaviour of a firm. They conclude that “knowledge management can develop absorptive capacity of a firm, which consequently contributes to innovation orientation and in turn, innovation behaviour of a firm.” [Zhou and Uhlramer, 2009].

Another case study has been executed by Costa et al. [Costa et al., 2009] in a company in Brazil, where they analyzed the benefit of web based social networks for furthering knowledge management.

Müller-Prothmann established a work, where he analysed the advantages of knowledge management, especially knowledge communication and social networks for innovation. His core focus is about "social network analysis in research and development” [Müller-Prothmann, 2006].

A project bridging the gap between innovation and collaborative networks at the MIT is called COIN (Collaborative Innovation Network). The core idea of COIN is setting up different collaborative networks, which contribute towards each other and further innovation and the interaction of knowledge workers in different organizations. The structure of COIN contains a core network, which is surrounded by an extended group in a concentric way [COIN, 2012].

Another project funded by the German Ministry for Research and Development is called Smarte Innovation [Sinn, 2012]. Smarte Innovation focuses on bringing the relevant stakeholders like experts of research institutions, economy and social life together to further product innovation [Sinn, 2010]. In this context the project aims to improve all steps of the product lifecycle. The project brings together the aspects of systems, people, anticipation, resources and technology and constantly feeds innovation impulses and knowledge in a counterflow towards the singular steps of the product lifecycle and especially the product development [Sinn, 2010].

An Italian project called TasLab (Trentino as a Lab) is part of the European Network of Living Labs [Liv. Lab, 2012], [TasLab, 2012]. The core idea of TasLab is to bring industry, research and user together to further collaborative innovation in a living lab environment. The TasLab portal “provides knowledge management facilities (e.g., competence matching) for collaborative innovation” [Švaiko et al., 2010] thus integrating knowledge and competence management features into the innovation management process. A current field of application is e.g. in eHealth.

MUSING (MUlti-industry, Semantic-based next Generation business INtelligenGence) was an EU-project running from 2006 to 2010 bridging the gap between BI and Semantic Web. The focus of MUSING was "to integrate Semantic Web and Human Language technologies and combine declarative rule-based methods and statistical approaches for enhancing the knowledge acquisition and reasoning in BI applications towards industries with a deep socio-economic impact" [MUSING, 2010]. From a company’s point of view the innovative improvement of the three areas: finances, risk management and internationalization were the main goals of MUSING [MUSING, 2010].

The examples from literature and practical projects above allow for two conclusions: 1) there are many different approaches towards integrating one or more of the disciplines of innovation management, innovation communities, knowledge management, knowledge networks and BI with each other and 2) while there are literature and commercial methods for web analytics available, running scientific projects regarding this aspect as well as social analytics in online contexts are not yet as prominent.

4 Concept approach

4.1 Analysis of the product innovation context in today’s companies

Figure 1, introduced in section $\rightarrow 1$, depicts a mind map of the product innovation context, environment and dependencies in today’s companies. Product innovation is influenced by different strategic and operational factors. The innovation process for a new product or product generation bases on the classical product lifecycle consisting of design, development, production, usage and recycling. Depending on the usage of modern technologies the innovation process may either be only internal and apply classical innovation methods, apply concepts of open innovation and the usage of social networks to gather customer input or a combination of both approaches. What kind of approach is chosen of a company often depends on the enterprise strategy. Then criteria like Make or Buy? Do we want to be first-to-market? or What is the budget for research, development and innovation? play a central role and pave the way for the innovation process [Müller-Prothmann and Dörr, 2011].
2006], [Völker et al., 2007]. The strategic decisions again depend on the sales and analytical numbers gathered from the previous product generation. The customer feedback is a deciding factor. Next to the strategic stakeholders, the knowledge experts, like product designers and engineers, have to be involved into the process. If the company has got an already established innovation community ($\rightarrow$ 2.1) or knowledge network ($\rightarrow$ 2.2), for example between internal and external experts, research centers, or customers, an exchange or feedback about new product ideas gives additional security for being on the right path.

As surmised, the creation of a new product innovation is no longer an isolated process, but depends on different factors, knowledge and stakeholders. The following InKnowE concept introduces a knowledge based way, to integrate these factors into one framework and support decision making for the most promising innovation idea.

4.2 InKnowE concept

This section illustrates the InKnowE concept approach for improving innovation processes in Enterprise 2.0 with the support of innovation communities ($\rightarrow$ 2.1), knowledge management and knowledge networks ($\rightarrow$ 2.2), BI features ($\rightarrow$ 2.3) as well as web and social analytic results ($\rightarrow$ 2.4). The target group for the framework are the heads of research departments during the innovation process as well as financial managers deciding about new innovation ideas. The concept is depicted in figure 2 as a modular interaction between the different disciplines. There are the four flows of innovation, knowledge, business and analytics, leading towards a founded decision making based on knowledge and semantic technologies.

Business flow: The starting point of the innovation process is the problem analysis of given products or the objective to create an entirely new product. This most often origins from non satisfactory sales numbers, the wish to change or extend the companies product portfolio or a change in the innovation strategy or budget. The relevant numbers and key indicators are stored in a BI solution or a general database and are the starting point for the innovation process. During the acquisition of new knowledge, these indicators or reports are stored as facts inside the knowledge base of the knowledge flow.

Knowledge flow: In addition to the business flow, a knowledge flow will be established. A knowledge base is the core element of this flow. New or additional knowledge for the knowledge base origins from the BI flow as detailed above, from knowledge acquisition sources, innovation communities or knowledge networks, especially knowledge experts like product designers and developers inside the company. If the products, which are sold by the company, require certain minimal standards, these are stored inside the knowledge base as well. The collaboration between internal knowledge networks and customers not only influences the knowledge base, but also the creation of new ideas during the innovation process. Depending on the type of company, they may move more or less in the direction of open innovation. The type of technology for the knowledge base has not yet been defined.

Analytic flow: Next to the knowledge flow, a parallel flow focusses on evidence from web and social analytics. The results are regularly stored in what we declare an analytics database. The gravitation of the technology (BI or knowledge management techniques) for this database depends on the type of acquired data and is not specified yet. If the majority of data are numerical, the solution will tend towards a BI tool or a SQL database. Should there be requirements form data or text mining features, the solution will most probably be a knowledge or semantic representation method. Another alternative could be the built up of several different databases and the merging of data flows inside the decision engine.

Independent from the underlying database, the results will be provided towards the decision module. For the data flows from knowledge base and analytics database an exchange format like xml or even a domain ontology will be applied.

Innovation flow and decision module: Inside the innovation flow layer, the innovation process starts with the aforementioned problem analysis, which receives initial input from business, knowledge and analytical flow, to summarize the product aspects, which have to be improved or newly invented. Based on this initial problem survey the relevant stakeholders start the creation of ideas, possibly in cooperation with an innovation community. After this phase is finished, there is a survey and rough filtering of ideas, which are not realizable or which don’t appear to be profitable. The ideas, where there is a chance for success on the market are formalized and handed over to the decision engine. The decision engine of the solution is still somehow a black box, as there is no final decision yet, which knowledge representation and inference method will be applied. A possible approach would be the application of semantic technologies. Maier, Hädrich and Peinl [Maier et al., 2005] define a bridge between knowledge work and semantic web technologies: “Knowledge Work requires content- and communication-oriented modelling techniques that define meta-data and provide taxonomies, ontologies, user models, communication diagrams, knowledge maps and diagrams that show what objects, persons, instruments, roles, communities, rules and outcomes are involved in the main knowledge-related activities.” This kind of interconnection allows for semantic technologies as part of the decision engine, as they also provide inference possibilities. Overall, the following alternatives are currently evaluated for application:

1. the creation of a domain ontology for the decision frame with a rule based inference engine on top of it,
2. the application of business rules or classical rules in addition to an inference engine,
3. the application of case based reasoning (CBR); CBR would have the advantage of formalizing the inventions as a case and adding the relevant facts from knowledge base and analytics database. This way a use case would already be there for the market launch of the possible new project.
4. Another possibility would be a combination of semantic methods and a numerical or statistical analysis; this approach would require a form of indicator, which would give a combined result and possible recommendation or refusal for the given invention.

The mentioned possibilities document a problem caused by the different types of data brought together for the decision process. A final decision about which singular or combined methods to apply for the aimed decision engine is only possible after deciding which tools and database technologies are applied for the other modules. Overall the aspiration is to create an innovation decision framework, containing different modules supporting the decision process towards
one or more innovation idea(s), leading to realization and market launch of the product.

Next to the inference and reasoning methods for the decision engine, a visualization, e.g. in the form of dashboards or condensed data views is aspired. The key aspect of the visualisation is to create a solution, where the user may see, if the given innovation idea has potential for success on the market or if there are factors and analysis results contradicting the new idea.

5 Conclusion & Outlook

The given paper proposes a concept called InKnowE for integrating different technologies like innovation management, innovation communities, knowledge management including knowledge networks, business intelligence, web and social analytics into one solution for enhancing the innovation process, especially the decision making for an invention, which should be realised and brought to market.

At the beginning of the paper the different disciplines are introduced, before a literature and project survey of similar approaches is given. After introducing the InKnowE concept, the question is: How to proceed from concept to realisation? There are additional open questions regarding the necessary technologies for databases, analytics tools as well as the decision engine and its visualization. Therefore the next step will be an intensive market research for tools and methods available for the different disciplines and interfaces to answer how to combine them. An essential decision to make is, which knowledge representation and inference methods are to be used for the decision engine and how to visualize the results not only for product developers, but also for management levels.

References


A Document-centered Authoring Approach for Ontology Engineering

Jochen Reutelshoefer, Joachim Baumeister
denkbares GmbH
Friedrich-Bergius-Ring 15
Würzburg, Germany

Georg Fette, Frank Puppe
Institute of Computer Science
University of Würzburg
Würzburg, Germany

Abstract
Most ontology development tools employ graphical user interfaces as interaction paradigm between system and developer. While being efficient for expert users, this is not always the best form of interaction when novice users are supposed to contribute to the development process. In this paper, we propose the use of the document-centered authoring paradigm for this human-computer interaction task where the ontology is modified by editing source documents using suitable markup languages. This alternative interaction paradigm shows several advantages for collaborative development involving participants of diverse expertise. We discuss the advantages and challenges of this approach and derive the requirements for a corresponding authoring environment. Further, we present a prototype implementation of such a tool and report about its use in case studies.

1 Introduction
The development of ontologies is a major challenge within the implementation of the semantic web. Today, expressive ontology representation languages including powerful reasoning mechanisms are available. While these foundational topics can be addressed in a strictly formal way, the actual process of the manual creation of an ontology, capturing a computer interpretable representation of a specific subject domain, strongly implies human factors. This knowledge acquisition effort implies a process of human-computer interaction (HCI) that has been found to be nontrivial. Nevertheless, it appears to be indispensable for the wide-scale employment of semantic technologies in various applications. Especially, the active involvement of domain specialists from the respective subject domain, usually being non ontology engineering experts, within the entire ontology development life-cycle is strongly desired [1]. However, enabling non expert ontology engineers to contribute directly to an ontology using some ontology design tool in practice shows to be challenging. Currently, most ontology engineering tools carry out this human-computer interaction (HCI) task which provides several advantages for the development of ontologies. In the so called document-centered authoring approach the user interacts with the ontology indirectly by editing documents, basically by using some common text editing interface. Segments complying to some predefined syntax are automatically processed and added to the formal model of the ontology while the user is provided with visual feedback accordingly. We claim that this editing paradigm shows some considerable benefits and can be a beneficial alternative to using graphical user interfaces in many cases.

The contribution of this paper is an introduction of document-centered development of ontologies including a discussion of the benefits but also the challenges of this approach. We derive a requirements specification for such an authoring environment and present an implementation of a corresponding tool. Further, we report about our experiences in using that system in projects to develop ontologies in collaboration with domain specialists. The rest of the paper is organized as follows:

In Section 2, we explain the document-centered ontology authoring approach in more detail. An implementation of a corresponding tool is presented in Section 3 by the use of examples. To demonstrate its applicability in practice, we report about real world case-studies in Section 4. In Section 5, we compare our approach to related work. We conclude with a summary and outlook in Section 6.

2 Document-centered Ontology Authoring
Document-centered authoring is an alternative authoring paradigm to graphical user interfaces providing different benefits and challenges for the development of complex digital artifacts. There, the authoring environment provides access to a set of documents, that can be modified and extended by the users in an unconstrained manner, employing some basic text-editing interface. To actually create components of the ontology, the user has to employ a formal syntax provided by the authoring environment. After each document modification, statements complying to this syntax are then translated to the ontology repository as shown in Figure 1 and visual feedback is given to the user. Figure 2 shows an example document from the pizza domain, where beside informal content the Manchester Syntax [7] is used to define an OWL class expression. The process of compiling the documents to the ontology repository decou-
structures the user from the machine readable representation of the ontology.

That way, the documents can be structured according to the users’ needs and are forming a kind of human-oriented representation layer of the knowledge. This freedom of structuring provides a number of benefits: It allows for the simple inclusion of informal support knowledge, e.g., documentation, comments, and figures, can be interwoven with the formalizable syntax statements at any place and in any style. Due to the declarative nature of the most common ontology representation languages, where the atomic parts (e.g., distinct axioms in OWL) are order independent of each other, the ordering of the statements can be left up to the user. Also the partition of the documents (including their names) can freely be chosen. Documents can freely be interlinked with others making interrelations of content parts explicit. The possibility to structure the domain knowledge allows for natural modularization into sub-domains. All these means of content organization can be exploited to give the document base a structure that is readable and memorable for the user considering his mental model of the domain. Studies in software engineering have shown that while working on programming code, the amount of time spent on reading the source compared to the amount of time spent on actually editing is higher than ten to one [8]. We believe, that in ontology engineering, where also complex digital artifacts are developed, a similar ratio applies. Consequently, the aspect of readability plays a very decisive role with respect to development productivity. Considering the freedom of structuring discussed above, the document-centered authoring approach provides excellent possibilities to improve readability of the gathered knowledge.

Editing of ontology statements using formal syntax in many cases is a challenging task for the user, implying considerable complexity. However, this complexity is to a large extent rooted in the expressiveness of the target representation language (c. f. general purpose programming languages). It is intrinsic and independent of the authoring paradigm, i.e., also contained in the GUI-based approach. For (subsets of) target representations of lower expressiveness also simple syntax can be introduced, also called domain specific language [9] (DSL), which is then also simple to read and edit. DSLs always should be designed to have the lowest expressiveness possible to reduce error rates and maximize productivity. There are also studies indicating that textual input can lead to higher productivity compared to GUI-based approaches if rich visual feedback is provided [10]. In the following, we call a syntax (DSL), together with a mapping instruction which determines its translation to the target ontology representation, a markup language (or in short markup). Hence, the specification of a set of appropriate markup languages is a foundational task when designing a document-centered ontology authoring environment. There should be expressive markup languages allowing advanced users to create components of high complexity, but also markups of low complexity allowing novice users to create very simple components without being too error-prone. We claim this approach has high potential to provide an authoring environment of high usability, if a suitable structuring of the content can be found considering the participating users and the domain. While we regard the freedom of structuring to be the strongest advantage of document-centered authoring, there are several other notable benefits when compared to the strict GUI-based approach:

**Low Barriers for Basic Contributions:** The level of the technical skills of the participating users often is rather diverse in ontology engineering projects and especially the support of users with low expertise of this task needs consideration. Editing text documents is a rather simple editing paradigm, when compared with some complex menu- and form-based user interfaces existing and already part of the daily work for a wide range of professionals in various domains. Therefore, it allows for basic contributions (e.g., adding informal descriptions, proof reading) without any training of a new tool. Being used to these simple contributions without difficulty, contributors often feel encouraged to explore more demanding activities.

**Example-based Authoring:** The actual ontology can only be modified by using the provided markup language in a proper way. The idea of example-based learning proposes, that initially markup statements are inserted into the document base (either modeling initial parts of the ontology or as some toy example). If a user can comprehend the meaning of these statements, he can easily adopt it for himself to express other parts of the ontology using simple copy/paste&modify. Often only the entity names need to be exchanged to create new valid ontology relations.

**Incremental Formalization** In the GUI-based approach, usually the formalization of a new component needs to be performed by the user completely in one action. The process of incremental formalization on the other hand aims to break up the formalization task into multiple steps. It starts with the insertion of informal content describing the domain, such as text and figures, in a completely unconstrained way. This content is either created by domain specialists or adopted from documents often already existing in the domain context. At first, it serves as a startup for the formalization process and later it forms the documentation and context of the ontology components. In the next step those content parts, which need to be formalized to form the intended ontology, are identified. After that, a tentative formalization is made, that is, the selected content is transformed towards the markup language. This initial, potentially erroneous or incomplete, formalization can then be refined gradually. These distinct steps require different
degrees of expertise in the domain, in ontology engineering, and usage of the employed acquisition tool. These different kinds of competencies often are distributed heterogeneously. Therefore, a decomposition of the formalization task into distinct steps, possibly involving different persons on different stages, simplifies the accomplishment of the formalization task. Hence, the incremental formalization workflow helps different participants contribute according to their respective capabilities and expertise.

**Version Control and Quality Management**
The documents managed by the authoring environment can easily be put under version control. This not only provides backups and undo-functionality, but also allows the straightforward application of the quality management practice Continuous Integration\(^1\) (CI), that has been established in the context of agile software engineering [8]. Hence, by employing a CI-based development workflow it is possible to continuously guarantee quality and transparency. Different kinds of automated tests can be executed regularly after modifications, such as competency tests [11], consistency checks, or profiles checks\(^2\).

**Challenges:**
Beside advantages the document-centered authoring approach also bears some additional challenges when compared to the GUI-based one:

- **Authoring Assistance:** Considering this most obvious issue, one can take advantage of experiences and techniques from software engineering which can be adopted to a far extent. It is very important to give the user feedback explaining whether and what ontology component had been created from some markup statement.

- **Navigation and Search:** While the freedom of structuring is beneficial for creating and maintaining a comprehensive structure of the content, it also imposes the problem of finding the position of some content piece or the appropriate location for some new content element. A system should provide full-text search as well as semantic search (based on the ontology entities and their interrelations) and efficient navigation mechanisms.

- **Refactoring:** It might happen that the structure of the current document base is considered not to be the most beneficial one at some point. Then restructuring of the content, which retains the compiled version of the ontology unaffected, is desired. A support for this restructuring task (beyond manual cut & paste) would be desirable.

- **Redundancy Detection:** In a document-centered ontology authoring environment, that is providing a maximum in flexibility of structuring, any ontology entity can be inserted at any location of any document. Therefore, it might occur more easily than in GUI-based tools that the same relations are defined/asserted

\(^1\)http://martinfowler.com/articles/continuousIntegration.html
\(^2\)http://www.w3.org/TR/owl2-test/
multiple times (possibly at different locations by different users). These types of redundancies should be detected by the system and pointed out to the users.

**Requirements:**
Considering this discussion of advantages and challenges we can derive the requirements for a document-centered authoring environment for ontology engineering: Most important is the simple access to the documents and a common way to create informal content (e.g., text, tables, figures) by creating and modifying documents. Further, a set of convenient and well readable markup languages, geared to the target ontology representation language, need to be supported including simple ones to be usable by non-expert users as well as expressive ones. For each markup, explained examples should be given at startup. The challenges of document-centered ontology authoring mentioned above should be addressed and a testing framework for continuous integration, connected to the document versioning system, should be included. Further common features necessary for effective ontology development, for instance considering visualization and debugging, need to be considered. These features can be implemented in a similar fashion as done by known existing GUI-based tools. However, the visualization and debugging views have to provide links to the corresponding document locations to allow for quick modifications during browsing or debugging sessions. Therefore, the environment requires a bidirectional mapping, to be capable to identify the text source which is the origin of a component in the ontology repository.

**3 A Document-centered Ontology Authoring Tool**
We have implemented an extension for the wiki KnowWE\(^3\) that forms a document-based ontology development environment as proposed in Section 2. KnowWE supports two different modes for ontology engineering, either in RDFS or OWL. In either mode different markups are available to form ontology statements, including for example the Turtle Syntax\(^4\) for RDF and the Manchester Syntax for OWL [7]. The design rationale of KnowWE follows the idea of lowering the barriers for participation of low experienced users via document-centered authoring. Considering the user interface, the system is designed to be fully backwards compatible to a normal wiki engine if no advanced ontology engineering features (markups) are used. Hence, novice participants use a standard wiki interface to access a set of documents, possibly only browsing the content or adding informal comments. They can step-wise commit to more advanced contribution activities when they feel capable, without changing to another tool or authoring paradigm. Also the design and user assistance of markups is intended to primarily support non-expert users.

**Entity Declaration and Compilation:** Within the specification of OWL 2 also the declaration of entities within an ontology document has been addressed to detect spelling errors within identifiers. This issue is even more relevant in the context of an ontology authoring environment where the ontology expressions are intended to be frequently edited by human users employing a text editor. Therefore, KnowWE also requires new entities to be explicitly declared. All referenced identifiers of a statement are automatically validated against the set of declared entities and error messages in case of undeclared use of identifiers (including misspellings) are generated. Error messages are visualized by red underlining and the corresponding statement is not translated to the ontology repository. In KnowWE all declarations are valid all over the wiki. Any statement can be defined at any position within the wiki, providing full flexibility to the user. A sophisticated compilation algorithm [12] after each modification calculates which statements have become valid or invalid considering the set of declared entities. Accordingly, those statements are inserted into or removed from the ontology repository respectively and corresponding error messages are generated. The incremental nature of this algorithm, always considering only the current document modification, guarantees instant response independently of the overall size of the wiki/ontology.

**Basic Markups:** The example page shown in Figure 3 shows different basic markups provided by the system KnowWE to create ontologies in RDFS. Declarations of new entities are highlighted in purple while references to existing ones are rendered in green. Predefined vocabulary (RDFS/OWL) is rendered in bold black font. Individuals can be introduced by using the 'def' keyword (1), while roles are defined using the keyword 'Class' (2). Object-properties can be defined in a similar way and optionally class references as domain and range can be given in brackets (3). Further, entities can also be defined as triples (4). Simple triple relations (5) can also be asserted by '>'. Basic RDFS/OWL vocabulary is available from the beginning. For more advanced users a version of the turtle markup is available to create more complex RDF expressions introduced by 'ttl'. For the quick and simple definition of explicit class hierarchies, KnowWE provides a dash-tree markup (7). There each class of a tree node is considered as rdfs:subClassOf its dash-tree father class, i.e., its predecessor having one dash less in its prefix.

**User Assistance:** A click on any term name opens a context menu providing additional information and options (8). Always a link to an overview page, describing the entity’s use within the overall document base, is given. For references a direct link to the declaration of the entity is provided. If a reference cannot be resolved an error message is rendered, indicated by red underline of the reference term. In this case, the context menu will present quick fix propositions based on edit distances to declared entities as assistance in case of typing errors. To simplify editing, the context menu also provides an edit button that allows to edit the corresponding statement inline in the context of the page view.

Full RDFS does not give limitations in meta-modeling, i.e., intermixing properties, class and individual names. However, KnowWE aims to support less experienced developers in creating simple ontologies. In this context, this kind of inter-mixture often is not an intended act of meta-modeling but basically a modeling error, sometimes leading to undesired undecidability. Therefore, the system shows warnings in different cases of ‘suspicious’ entity inter-mixtures. For example, KnowWE asserts that entities defined as range and domain, shown in Figure 3 markup example (3), are actually declared as classes and

---

\(^3\)http://www.knowwe.de

\(^4\)http://www.w3.org/TR/turtle/
show a warning otherwise. Additionally, the employment of classes or individuals in the predicate positions of triples will be marked. Another assistance mechanism is considering the use of rdfs:domain and rdfs:range. The semantics of these terms often appears to be unintuitive to novices in RDFS-modeling, which tend to consider them as constraints to the use of the respective property. KnowWE provides a mode that renders warnings, if assertions lead to a class membership only being derived by the RDFS inference rules for domain and range (without being asserted/derived otherwise). Hence, the use of domain/range as constraints for the use of properties is supported by the authoring system (without affecting the resulting ontology/inference).

Manchester Syntax: To make available the full expressiveness of OWL, we have implemented the Manchester Syntax. The Manchester Syntax has been designed for improved readability of OWL statements but also provides a convenient way for editing. The syntax is frame-based, that is each ontology entity is defined as a frame describing the various characteristics of the entity. The keywords for the entity definitions (e.g., ‘Class:’, ‘Individual:’) indicates the use of a Manchester Syntax statement to the KnowWE system. In KnowWE the slots of an entity frame can be distributed to different locations if convenient. The main frame declares the entity and in other parts of the wiki additional characteristics can be added to the entity using the ‘EXTEND’ keyword followed by the term name of the entity. In Figure 2 an example wiki page from the pizza domain is shown describing Pizza Margherita in the system KnowWE. On the lower part, the Manchester Syntax is used to add the definition of a corresponding class to the ontology (a). The right hand part (b) shows the corresponding wiki source text of the page.

Inspecting Ontology Competency: Keeping track of the competency of an ontology provided with an expressive reasoning mechanism, e.g., for RDFS or OWL semantics, is a challenge — for novice and experienced users. Therefore, in KnowWE different methods for inspecting the current competency of the ontology by the use of the underlying reasoning engine are provided. For classes and properties an overview on the members/relations can be visualized using the context menu (RDFS mode only). In addition to common query interfaces, also inline queries (in SPARQL for RDFS, in Manchester Syntax for OWL) can be embedded into the documents, always rendering the

---

5http://www.w3.org/TR/owl2-manchester-syntax/

6http://www.co-ode.org/ontologies/pizza/
current query result on page load. Further, for any knowledge statement to be inserted into the ontology an inference diff (RDFS only) can be generated and visualized on demand. The diff for a statement is the set of triples which is the result of the set subtraction of the inference closure of the entire ontology including and excluding respectively the particular statement. That way, the "impact" of a particular statement for the current version of the ontology is made explicit for the user, showing an empty set for redundant assertions.

**Quality Management with Continuous Integration:**
We have integrated the following test methods for the KnowWE Continuous Integration framework to support ontology engineering:

- **Competency Test:** The user can define competency tests for OWL ontologies by specifying a set of individuals that are expected to be derived as members of some class. In case of RDFS, a SPARQL query is defined together with a set of resources as expected result.
- **Consistency Test:** Tests whether the connected OWL-reasoner discovers any inconsistencies in the current version of the ontology.
- **Profile Check:** Allows to test whether the current ontology is in OWL DL.

Of course the capabilities of these tests strongly rely on the underlying reasoner KnowWE is connected with. The tests can be attached to a CI-Dashboard that manages the versions and test results. One dashboard is shown in Figure 4 containing a consistency test and one competency test. On the left, the version history of builds is shown. The consistency test has failed within the most recent version and therefore the current build is marked as failed by the red bubble. A dashboard can be configured to one of three trigger modes: `onChange`, `onSchedule`, and `onDemand`.

**Import, Export and Namespace Handling:** In the basic configuration, namespace handling is entirely managed by the system to make it easy for the user. However, this implies the unique name assumption for entity names for within the scope of the system. To develop an ontology using terms of other existing ontologies, those ontologies can easily be imported into the system (and also connected to the compilation mechanism, i.e. registered as existing declarations). To allow the created ontologies to be used in other contexts, an export feature is given, delivering the entire ontology in RDF/XML format. Imported ontologies however cannot be modified within the system currently.

**Implementation Overview:** For the management and reasoning of the semantic data we utilized the following FOSS software components: [OWLAPI][13], Hermit[7], Sesame[8], JSPWiki[9], swiftOWLIM[10].

The KnowWE software described in this paper is LGPL-licensed and the sources can be downloaded from [https://fisc.informatik.uni-wuerzburg.de/][11].

### 4 Case Studies

**HermesWiki:** In the eLearning platform HermesWiki[14], in addition to informal knowledge about the domain of ancient history, an ontology is built. This domain ontology can be used for different advanced features, such as semantic search and navigation, augmented content presentation, or the generation of automated quiz-sessions for the students. The ontology is defined using different markups, e.g., the turtle markup, dash-tree markup (for subclass hierarchies) and the Manchester Syntax. The HermesWiki currently contains about 900 wiki pages forming an ontology containing about 600 individuals such as persons or cities. During the last three years it has been developed in cooperation of the department of Ancient History and the department of Intelligent Systems at the University of Würzburg by more than 16000 edits. About a quarter of the edit operations originate from automated refactoring scripts converting the markup statements according to the introduction of improved versions of the markup languages. The members of the Department of Ancient History work autonomously on informal contents and A-Box assertions. More complex extensions of the ontology vocabulary are usually evolved in close cooperation with the computer scientists. However, we also observed that significant parts of the ontology have been developed by the domain specialists independently and without explicit training sessions. These components, in parts showing modeling flaws, indicate that the authors are only limited by conceptual ontology modeling expertise but not by the usage of the tool. More information about the Hermes ontology and its creation using special markups can be found in [14].

**Other Projects:** The WISSKONT project considers the creation of an intelligent information system in the medical domain of cataract surgery. The system is currently under development and it will support the ophthalmologist during the treatment process before, in-between, and after the cataract surgery. Therefore, the system needs to present relevant knowledge of the domain, which is integrated at varying degrees of formality. In order to make this content accessible for a semantic search engine effectively, an ontology of the domain is created. It currently consists of three major parts: (1) a concept hierarchy of currently 340 concepts using dash-tree markup; (2) further semantic links between the concepts implemented as triple-markup; (3) annotations establishing relations between domain concepts and content elements (e.g., textbook content with images describe particular aspects of a treatment process), implemented by a custom markup. The WISSKONT project is part of the WISSASS project, a cooperation of the Karlsruhe Institute of Technology, Germany (KIT) and the denkbares GmbH. It is funded as a ZIM-KOOP project by the German Federal Ministry of Economics and Technology (BMWI).

In the KnowSEC project, which is a cooperation with the Umweltbundesamt[13] and the denkbares GmbH, the main aim is the management of knowledge about substances of high concern. New properties about substances can be defined, when new knowledge arrives. Aggregations and overviews on substances are flexibly created by specifying SPARQL queries. About 50 properties are used to form relations between substances, substance groups and environment risks to create an ontology which is then used in the decision process for the examination of substances.

---

7 [http://hermit-reasoner.com](http://hermit-reasoner.com)
8 [http://www.openrdf.org/](http://www.openrdf.org/)
10 [http://www.ontotext.com/owlim](http://www.ontotext.com/owlim)

---

12 Knowledge on Substances of Ecological Concern
13 Federal Environment Agency Germany
of potentially high ecological concern.

In both projects the initial experiences are promising for an effective collaborative ontology development with the domain specialists.

5 Related Work

In this section, we compare the presented approach to different classes of tools:

**GUI-based Ontology Editors:** Prot´ege with its OWL-Plugin [2] is probably the most widely used and most mature free ontology development tool available. As a rich client application it provides a sophisticated graphical user interface. The ontology entities can be browsed using different tab-based views. Clicking on an entity starts a form-based editor allowing to extend or modify the entity. Additionally, graph-based visualizations can be generated as alternative views. Other tools employing a similar editing paradigm are TopBraid-Composer\(^\text{14}\), the NeOn toolkit [15], and SWOOP [16]. The editing paradigm applied by these tools is fundamentally different compared to the document-centered authoring paradigm proposed in this paper. Although, the tools are widely used, we argue that there are scenarios where the document-centered approach can be more effective due to the advantages discussed in Section 2, especially when non-expert users are involved.

**Semantic Wiki:** A class of tools providing simple access to documents are (semantic) wikis. Semantic MediaWiki for example, being the most common semantic wiki implementation, has been developed as a wiki extension providing a semantic model of the content. It has not been intended as a general purpose ontology development environment and most annotations correspond to simple A-Box statements [17]. Semantic MediaWiki imposes several restrictions on structuring the formal knowledge considerably reducing the freedom of structuring. For example, a property can only be defined on a separate property page and a class/category definition only on a category page. In recent years, numerous extensions for SMW have been created to allow for the definition of various kinds of knowledge (c.f. SMW+\(^\text{15}\), [18]). However, to the best of our knowledge until now there are no extensions providing carefully designed markup languages posing low barriers including compilation and user assistance to support the development of ontologies in RDFS or OWL as proposed in this paper. These features implemented for SMW would result in a well-suited document-centered ontology authoring environment in the sense of this paper. There is another extension to SMW called Semantic MediaWiki Ontology Editor\(^\text{16}\). This toolkit however integrates form-based editing components departing from the document-centered authoring approach.

**Software Development Environments:** Software is developed in a way that is quite similar to the document-centered authoring approach described in this paper since decades. Today's sophisticated integrated development environments (e.g., Eclipse, Netbeans) support the programmers allow the entire development workflow. Many methods of user assistance found in these IDEs can be transferred to document-centered ontology engineering. However, compared to software engineering, there are also important differences. Ontology engineering projects usually bring together people with different backgrounds and strongly differing expertise with respect to the subject domain on the one hand and ontology modeling on the other. While in software engineering the source files usually are only edited by professional software engineers, in this context also domain specialists, often with low technical backgrounds, are supposed to step into the author role. Further, informal content describing the domain is less important in software engineering as the code typically cannot be derived from domain knowledge by a formalization process. While IDE interfaces are designed for expert users we propose to design the interface rather simplistic with advanced functionality being hidden for non-expert users to provide a low entry barrier for domain specialists. Novice users should not be distracted by the complexity of possibilities but be able to consider the tool as a basic document editor.

\(^{14}\)http://www.topbraidcomposer.com

\(^{15}\)http://www.ontoprise.de/de/loesungen/smw-plus/

\(^{16}\)http://www.phpkode.com/projects/item/semantic-media-wiki-ontology-editor/
In contrast to common software development tools, where comments need to be marked as such and cause compile errors otherwise, contents not complying to some markup language should be considered as regular informal content (e.g., comments, documentation). Further, the technical barrier to access the tool should be as low as possible, for instance providing a web-based application accessible using a standard web-browser.

In sum, the document-centered ontology development approach aims to combine the document-centered and textual language-based authoring paradigm approved in software engineering with a simple user-interface (e.g., provided by Semantic MediaWiki) incorporating support for development and maintenance of expressive ontologies as provided by existing GUI-based tools such as Protégé.

6 Conclusion

Current ontology development tools perform the HCI task by the use of graphical user interfaces. Novice users therefore are forced to deal with a new tool of high complexity, demanding rather high introductory training even for very simple activities. In this paper, we propose the use of the document-centered authoring paradigm, being an alternative to graphical user interfaces, which allows less experienced users to contribute according to their capabilities more easily. We provide fundamental concepts how a corresponding authoring environment can be built. A prototype implementation of such a tool is presented. Further, we report about initial experiences of this ontology authoring method employed in two case studies. According to our experiences this approach facilitates for untrained domain specialists to express their knowledge and insert it into a system initially as a starting point for collaborative incremental formalization. More experienced users are able to formalize their knowledge using the markup languages. However, even if they are not restricted in the usage of the tool, missing ontology modeling skills can lead to unfavorable ontology design. In this case, counteraction by ontology engineers is required and possible by driving an agile evolutionary development process including design refactoring if necessary. The system KnowWE is under ongoing development and a demo of the described version is accessible. For the future, we try to gain still more experiences about how documents should be structured/organized and how markup languages can be designed and used to provide optimal usability for the authors. An established catalog of best practices is one important task for further research.

References

Modeling the Structure of Spreadsheets

Christian Liguda
German Research Center for Artificial Intelligence (DFKI GmbH),
Cyber-Physical Systems, Bremen
Christian.Liguda@dfki.de

Abstract
Spreadsheets are widely used in many different domains like business planning or science, e.g. for calculation, planning, statistical analysis or test evaluation. For these and many other domains it is important that spreadsheets are error free, easy to interpret, maintain and change. However, in the last years it has become more and more evident that spreadsheets are highly erroneous and hard to maintain. The main reason for this is the lack of high level structures in spreadsheets, which rather allow a more structured than a cell based view. In this paper we present an abstract model for spreadsheets, which is able to represent the underlying structure explicit and is independent of the concrete layout. This model is part of a project in which the abstract model is used to assist a user in creating and understanding large and complex spreadsheets.

1 Introduction
Spreadsheets are a widespread tool for financial administration, science, business planning and much more. The main reason for the great acceptance of spreadsheets is their very simple and flexible structure. A user can place names, values or formulae in cells and can thereby arrange the data in any way he likes. On the one hand this easy and flexible structure leads to the application of spreadsheets in many different domains. On the other hand this cell based structure does not provide any support for abstract modeling and semantic interpretation. While spreadsheets have become more and more complex, professional spreadsheets are highly erroneous which often result in expensive misleading decisions. For example the West Baraboo Village Board announced on 09.12.2011 that they have to pay $ 400,0000 more for their most recent 10-year borrowing plan than originally projected. The reason for this erroneous assumption was a wrong formula in their spreadsheet which ignored a relevant cell (see [1] for this and many more examples).

The aim of the SiSsI (Software Engineering for Spreadsheet Interaction) project[2], which is a cooperation between the DFKI and the Jacobs University Bremen, is to develop tools and methods for augmenting spreadsheet applications with semantic techniques. These tools and methods should help users to create, change, maintain and understand large and complex spreadsheets by bringing ideas from software development and verification to the development of spreadsheets. As part of the SiSsI project we have developed an abstract spreadsheet model, which abstracts spreadsheets from cell and layout information and contains additional information about the underlying structure of spreadsheets.

Our intention for developing an abstract spreadsheet model is to use it as an intermediate layer between a concrete spreadsheet and an ontology, which describes the semantics of the concrete spreadsheet. Thereby it can be used for verifying that the concrete spreadsheets represents the semantics correctly or can help a user to understand the structure of a spreadsheet. Furthermore an abstract spreadsheet model can be used to assure that a rearrangement of the content in a spreadsheet does not destroy the underlying structure.

In the next section we will analyse the problem of the cell based structure of spreadsheets in more detail. In Section 3 we will discuss some other approaches for formal spreadsheet models as well as the overall semantic framework which was developed within the SiSsI project. In Section 4 we will introduce the main aspects of our abstract spreadsheet model. The mapping between a concrete spreadsheet and the abstract structures is then defined in Section 5. In the last section we will discuss our approach and give some outlooks to future work.

2 Problem Analysis
The high error rate of spreadsheets often results from the WYSIWYG principle of common spreadsheet programs. The problem of that principle is the lack of mechanisms for representing the underlying structure of spreadsheets. For a better understanding we will analyse the example spreadsheet from Figure 1 here in more detail and describe some underlying structures.

In Figure 1 we see that the years 1984 to 1988 appear in the spreadsheet and that 1984 to 1986 are labeled as Actual and 1987 and 1988 are labeled as Projected. Furthermore we infer from the layout that Salaries are Expenses and that 1986 is labeled as Actual, whatever that means. Lets assume, that the values for Revenues and Expenses of the actual years 1984 to 1986 are directly entered and the values for the projected years are calculated from the years before. This
structural difference between actual and projected years is illustrated by using different colors for the expenses. But for the revenues this structural difference can just be recognized by inspecting the cells. The same problem occurs for the cells that are related to Total Expenses and Profit (Loss) if their values are calculated from some other cells. Furthermore we can not see, whether the values of all cells that are related to Total Expenses (B12:E12) are calculated by the same formula or not without inspecting all cells.

As we have seen, the semantic structure is not part of the spreadsheet, but is partly given by the layout of a spreadsheet. Some semantic structures like the formulae are just visible by inspecting all cells. Therefore a user must be very careful if he wants to change the spreadsheet, because he must be aware of the underlying semantics of a spreadsheet and assure that the layout of the changed spreadsheet still represents the underlying semantics.

To prevent such errors we will introduce a model which separates the structure from the layout of a spreadsheet. Therefore we define two maxims that must be satisfy by our model. Our first maxim is that the model should be completely independent of the layout. More precisely, the model should be invariant of all rearrangement operations, like exchanging two rows or columns or shifting tables to other places. Therefore the position, order or alignment of elements should not be part of the abstract model. The second maxim is that the model should be able to represent underlying structures of a spreadsheet or at least be easily augmentable to represent such structures. By defining a model which fulfills these maxims we are able to represent the data from spreadsheets in a clearly structured way as well as representing the structural relations between the data.

3 Related Work
David et al. [3] developed a semantic framework and implemented it as a semantic framework for spreadsheets within the SiSsI project. Their framework is able to “complement existing software applications with semantic services and interactions based on a background ontology”. We integrated our model in their framework so that the abstract model could be used for spreadsheet interaction.

Our work is based on the previous work of [5] by using some of their terms. They introduce the term legend as those non-empty cells that do not contain input or computed values but contain text strings that give auxiliary information on the cells that do. Furthermore they define a grid region as functional block, if that region could be interpreted as a function which maps elements from a legend to values. Thereby it is not critical, if the values are calculated or inputted, because the function is meant to be an intended function of the spreadsheet creator. For example, the region B13:F13 of Figure 1 could be interpreted as a function, which maps years to the total expenses in that year and the region B4:F4 as a function that maps a year to the revenues of that year.

Their have been some other approaches for structural models of spreadsheets from other authors. Engels and Erwig [4] developed an object oriented model which they call ClassSheets. They define “ClassSheets consist of a list of attribute definitions grouped by classes and are arranged on a two dimensional grid”. Thereby the attributes of a ClassSheet can be distributed over the grid. Figure 3 shows a ClassSheet for the Spreadsheet from Figure 2, in which the different colors are used to represent different ClassSheets by what the carve-up of the ClassSheets is illustrated. As we see in this example, the structure and layout information are somehow depending and ClassSheets are similar to real spreadsheets as they contain the structural layout without all the data. The abstract syntax of a ClassSheet, which can also be found in [4], contains the layout information also and therefore the definitions for ClassSheets are broken up into multiple parts like the visual representation.

An other model was developed by Paine [6]. The main idea is that spreadsheets can be expressed as a set of equations e.g. \( \text{Year}[2000] = \text{Profit}[2000] - \text{Expenses}[2000] \). By replacing variables by cell addresses (e.g. \( \text{Year}[2000] \) by A3) a spreadsheet can be created from this equations. In addition to those equations, Pain uses in [7] informations about tables to represent regions that are similar to the regions of the above mentioned functional blocks. This tables just contain a name and the size, like \( \text{Builds}[2000:2010,1:120] \). Furthermore Paine shows how these tables, equations and layout information are enough to represent spreadsheet and rearrange them within a spreadsheet.
Even if ClassSheets and Paine’s model are able to represent the structure of a spreadsheet in a more abstract way, they do not satisfy our maxims from Section 2. Therefore we develop a new abstract spreadsheet model and use a mathematical notation so that we are able to do proofs later, like that an abstract model is isomorphic to a spreadsheet.

4 Abstract Spreadsheet Model

We start by defining an abstract data model in Subsection 4.1 which is able to represent all the data from a spreadsheet without referring to cells anymore. In the next two subsections we define some high level structures which can be used to make the hidden structure of a spreadsheet explicit.

4.1 Abstract Data Model

As mentioned above we want to define a model which abstracts from cell and layout information. Therefore we define three sets whose elements represents cells, the content and underlying formulae of cell in $\Lambda$-notation.

Definition 4.1.

a) $\Lambda$ is an infinite set \(^1\) which is used to represent all elements of a spreadsheet, more precisely an element of $\Lambda$ is a proxy for one or more cells of a spreadsheet.\(^2\)

b) $V$ is a set of names and values (e.g. $\mathbb{R} \cup \text{Strings} \cup \text{Dates} \cup \ldots$).

c) $\Lambda$ is set of $\lambda$ expressions of the form $\lambda x_1 \ldots x_n.\phi$ where $\phi$ is a spreadsheet function. We note $\text{ord}(\lambda x_1 \ldots x_n.\phi) = n$.

Now we have defined some sets which can be used to represent cells ($\Lambda$), content ($V$) and formulae ($\Lambda$). For representing the content of a concrete spreadsheet we have to define, what the content of each proxy for a cell (an element $a \in \Lambda$) is and if the proxy is related to a formula. We give a definition first and explain it in more detail afterwards together with an additional example.

Definition 4.2. We call $D = (v_\Lambda, \lambda_\Lambda, p_\Lambda)$ an abstract data model, where

- $\lambda_\Lambda : \Lambda \rightarrow p\Lambda$ maps an element to a spreadsheet function.
- $p_\Lambda : \Lambda \rightarrow \bigcup_{n=1}^{\infty} \prod_{n} \Lambda$ maps an element to parameters for $\lambda_\Lambda$. Thereby $p_\Lambda$ must fulfill the condition: $\forall a \in \Lambda$, if $\lambda_\Lambda(a)$ is defined, $p_\Lambda(a)$ must be defined with: $\text{ord}(p_\Lambda(a)) = \text{ord}(\lambda_\Lambda(a))$.
- $v_\Lambda : \Lambda \rightarrow V$ maps an element to a value or name. $\forall a \in \Lambda$, if $\lambda_\Lambda(a)$ is defined, $v_\Lambda(a)$ must be defined with: $v_\Lambda(a) = \lambda_\Lambda(a)(v_\Lambda(p_\Lambda(a)_1)(\ldots), v_\Lambda(p_\Lambda(a)_n))$.

The main idea behind this definition is to use elements from $\Lambda$ to represent the cells from a spreadsheet. The content and formula of a cell is represented by mapping the corresponding element from $\Lambda$ to a value in $V$ and a formula in $\Lambda$. While formulae in spreadsheets are related to concrete cells (like $=A3+A4$) a $\lambda$ expression depends on variables and not on concrete values. Therefore we need the function $p_\Lambda$ which maps an element to a $n$-tuple which must be applied to the $\lambda$ expression. For example the $\lambda$-expression for $=A3+A4$ is $\lambda x_1 x_2 x_1 + x_2$. Obviously the information about the concrete values is now missing and therefore we define $p_\Lambda(a) = (a_{A3}, a_{A4})$ where $a_{A3}$ and $a_{A4}$ are the abstract elements for the cells $A3$ and $A4$ and $a$ is the abstract element which contains the formula. The constrain in the definition of $p_\Lambda$ means, that the size of the $n$-tuple must be equal to the number of variables in the lambda expression.
The constrain for $v_\Lambda$ just says that the value of an element is the result of the $\lambda$ expression when it is applied to the values of the corresponding parameters like it is described in the following example.

Example 4.3. Let $a_1, a_2, a_3 \in \Lambda$ be abstract representations for the cells $C4$, $C13$ and $C15$ in Figure 1 with $v_\Lambda(a_1) = 4,992$, $v_\Lambda(a_2) = 3,291$ and $v_\Lambda(a_3) = 1,701$. The formula from cell $C15$ is $=C4+C13$ and will be represented by $\lambda x y.x + y$ with parameters $p_\Lambda(a_3) = (a_{C4}, a_{C13})$. These functions fulfill the constrains for the abstract data model, because $v_\Lambda(a_3) = \lambda_{\Lambda_{(a_3)}}(v_\Lambda(a_1), v_\Lambda(a_2)) = \lambda_{\Lambda_{(a_3)}}(4,992, 3,291) = (4,992 + 3,291) = 4,992 + 3,291 = 1,701$.

Because the parameters of a formula are given as elements of $\Lambda$ our model is completely independent of the concrete spreadsheet layout and is therefore very robust for layout changes. Furthermore it is easy to determine, if some elements have the same underlying formula or not.

Another advantage is that elements are not differentiated by their position alone. In the spreadsheet of Table 1 the city name Hannover appears two times, but once referred to the city Hannover in Germany and once to Hannover in North Dakota (USA). By using the abstract data model this could easily be modeled by two elements $a_1, a_2 \in \Lambda$ which are mapped to the same value in $V$ but to different positions in the concrete spreadsheet.

Until now we are able to represent all cells in an abstract way which does not depend on the cell position any more. Therefore an abstract data model satisfies our first maxim from Section 2. However, the abstract data model is not able to represent the structures that were discussed in Section 2 and therefore our second maxim is unfulfilled. Therefore we introduce now some high level concepts which make some underlying structures of a spreadsheet explicit so that the second maxims will be fulfilled as well.

4.2 Legends

We augment $\Lambda$ with a simple structure for representing legends like Years or Costs.

Definition 4.4. Let $L \subset \Lambda$ be finite and $\tilde{L} \in \Lambda \setminus L$. We call the tuple $\gamma = (\tilde{L}, \tilde{\Lambda})$ an $\mathcal{E}$-Structure. We say that $L$ is the underlying set of $\gamma$ and that two $\mathcal{E}$-Structures $\gamma_1 = (L_1, \tilde{L}_1)$ and $\gamma_2 = (L_2, \tilde{L}_2)$ are disjoint, if $\tilde{L}_1 \neq \tilde{L}_2$ and $L_1 \setminus L_2$ and $L_2 \setminus L_1$ are pairwise disjoint.

We use the set $L$ to represent informations about the different items of a legend and the element $\tilde{L}$ to represent information about the complete legend. Whether the given name $v_\Lambda(\tilde{L})$ of a legend appears in a spreadsheet or not will just depend on the mapping between abstract and concrete
In the spreadsheet of Figure 1 the names of all the different legends (e.g. Years) do not appear in the spreadsheet except the name of the legend Expenses.

**Example 4.5.** To explain the above definition we use the cutout of a spreadsheet which is given in Table 1. In this spreadsheet we have three legends. We see that the city name Hannover appears two times, but we would assume that they refer to two different cities. Furthermore we see that m and f as abbreviation for male and female appear several times and we assume that they all have the same meaning. The abstract representation for this cutout can be modeled as follows: We define $\gamma_1, \gamma_2$ and $\gamma_3$ with $L_1 = \{a_{\text{Germany}, a_{\text{USA}}}, a_{\text{Hannover}, a_{\text{Berlin}}, a_{\text{NYC}}, a_{\text{Hannover}2}\}$ and $L_3 = \{a_m, a_f\}$ to represent the items of each legend. To represent the headers we define $\theta_1 = a_{\text{Country}}, \theta_2 = a_{\text{City}}$ and $\theta_3 = a_{\text{Gender}}$. For the reconstruction of the spreadsheet we just need to map each element of $L_1, L_2$ and $L_3$ to one or more positions in the spreadsheet and $\theta_1, \theta_2$ and $\theta_3$ each to one position.

Now we are able to represent legends in a more structural way but we cannot express the relations among legends, e.g. that NYC is related to USA and not to Germany. Therefore we define:

**Definition 4.6.** Let $\gamma_1, \ldots, \gamma_n$ be L-Structures with underlying disjoint sets $L_1, \ldots, L_n$. A subset $\delta \subseteq L_1 \times \ldots \times L_n$ is called a legend relation of $\gamma_1 \ldots \gamma_n$.

Thereby the order of the sets is not important for the interpretation of a legend relation $\delta \subseteq L_1 \times \ldots \times L_n$ and we just say that $L_1, \ldots, L_n$ are related to each other. For the legends in Table 1 we could say that the city depends on the country but it would not make any sense to say that the gender depends on the city. Furthermore the order of the legends in the spreadsheet is exchangeable which is not compatible with a fixed $L_1$ depends on $L_{n-1}$ semantic and therefore the order is not part of the semantic.

**Example 4.7.** We explain the definition by representing the relations between the 3 legends of Table 1. Therefore we can define one legend relation as a subset of $L_1 \times L_2 \times L_3$ or split it into two legend relations, one as a subset of $L_1 \times L_2$ and the other as a subset of $L_2 \times L_3$. We choose the first option and define $\delta = \{(a_{\text{Germany}, a_{\text{Hannover}}, a_m}, (a_{\text{Germany}, a_{\text{Hannover}2}}, a_f), \ (a_{\text{Germany}, a_{\text{Berlin}}, a_m}), (a_{\text{Germany}}, a_{\text{Berlin}}, a_f), (a_{\text{USA}}, a_{\text{NYC}}, a_m), (a_{\text{USA}}, a_{\text{NYC}}, a_f), (a_{\text{USA}}, a_{\text{Hannover}2}, a_m), (a_{\text{USA}}, a_{\text{Hannover}2}, a_f)\}$. From this subset of the Cartesian product it is easy to infer that Hannover2 is related to Germany and Hannover2 is related to USA.

4.3 Functional Blocks

In spreadsheets the dependencies between a cell in a functional block and the corresponding legend items are not explicitly modeled but can usually inferred by the user from the structure of a spreadsheet (see Section 2). These implicit dependencies should be modeled in the abstract spreadsheet model explicitly. A functional block will be represented as a more dimensional function which depends on the surrounding legends. For example, the functional block of Table 1 can be modeled as 2 or 3 dimensional function with domains $\text{Costs} \times \text{Years} \times \text{YearType}$.

**Definition 4.8.** An injective function $\phi : \Sigma \subseteq L_1 \times \ldots \times L_n \rightarrow \Lambda$ is called a functional block.

**Example 4.9.** Suppose we model the legends of the Winograd Spreadsheet by $L_1 = \{a_{\text{Actual}}, a_{\text{Projected}}\}$, $L_2 = \{a_{1985}, \ldots, a_{1988}\}$ and $L_3 = \{a_{\text{Salaries}}, \ldots, a_{\text{Other}}\}$. We decided to model the first column from the spreadsheet by different L-Structures, because the revenues, expenses and the profit have different structures and semantics. Therefore $L_3$ just represents the different expenses. Now we model the functional block in $\bar{\Xi} : \bar{\Xi}_1 \bar{\Xi}$ by a function $\phi : \Sigma \subseteq L_1 \times L_2 \times L_3 \rightarrow \Lambda$. The function $\phi$ is defined on a subset of $L_1 \times L_2 \times L_3$, because not every triple appears in the spreadsheet, e.g. $(a_{\text{Actual}}, a_{1988}, a_{\text{Salaries}})$ does not appear.

Obviously the function $\phi$ should be injective, because no two different elements from the domain should be mapped to the same element in $\Lambda$.

5 Mapping

In Section 4 we have discussed a model to represent the content and structure of a spreadsheet in a more abstract way. Therefore our model ignores all cell information and so the original spreadsheet could not be reconstructed from an abstract spreadsheet model alone. In this section we discuss the main aspects of the mapping between a concrete and an abstract spreadsheet, but we will not discuss all details which are necessary to prove that the mapping is isomorphic.

We represent a concrete spreadsheet by using the definition of an abstract data model, but replace the set $\Lambda$ by $\Sigma = \mathbb{N} \times \mathbb{N} \times \mathbb{N}$. Thereby a tuple $(n, x, y) \in \Sigma$ represents the cell position in the worksheet $n$, e.g. (1,5,3) stands for cell D6 in the first worksheet.

**Definition 5.1.** We call $\mathcal{S} = (\Sigma_v, \Lambda_\delta, p_{\delta})$ a formal spreadsheet model, where $\Sigma_v, \Lambda_\delta$ and $p_{\delta}$ fulfill the same conditions as $\gamma_1, \lambda_\delta$ and $p_\delta$.

A very intuitive idea is to define a mapping between $\Lambda$ and $\Sigma$. While this is enough to map an abstract spreadsheet to a formal spreadsheet model, this mapping does not provide enough information to build an abstract model from a formal one, because it does not say anything about L-Structures and functional blocks. We give a formal definition of mappings for legends and functional blocks first and then explain them by example afterwards.

**Definition 5.2.**

a) A legend mapping is a triple $\psi = (\psi_L, \psi_\Lambda, \psi_\Delta)$ with a function $\psi_L : \Omega \rightarrow \Lambda$, $\Omega \subseteq \Sigma$, elements $\hat{a} \in \Lambda \setminus \psi_L(\Omega)$, $\omega \in (\Sigma \setminus \Omega)$ and the constraint: $\psi_\Delta(\omega_1) = \psi_L(\omega_2) \Rightarrow \psi_L(\omega_2) = \psi_L(\omega_1)$.

b) A functional block mapping is a tuple $\psi = (\psi_L, \psi_\Sigma)$ of an injective function $\psi_L : \Omega \rightarrow \Lambda$ and a bijective function $\psi_\Sigma : \Omega \rightarrow \Sigma$ with $\Sigma \subseteq L_1 \times \ldots \times L_n$ and $\psi_L(\omega) \cap L_i = 0, 1 \leq i \leq n$.

We start by taking a closer look at the legend mapping by explaining the constraining and how to create an L-Structure.
(L, 1) from a formal spreadsheet and a mapping. The given constraint for a legend mapping assures that whenever two different cells are mapped to the same element in the abstract spreadsheet model, the cells contain the same values, formula and parameters (e.g. the four cells containing “m” in Table 1 are mapped to a single abstract element. Therefore all cells must contain the same value, formula and parameters). Given a formal spreadsheet and a mapping we can create an L-Structure by defining \( L = \psi_2(\Omega) - \{a\} \) and \( 1 = a \). In doing so we have a partial mapping between \( \Lambda \) and \( S \) and furthermore have enough information for building an L-Structure. \( \Omega \) is used to map \( 1 \) back to a spreadsheet.

A functional block mapping should provide enough information to define a functional block \( \varphi : \Sigma \to \Lambda \) from a cutout of a spreadsheet. Like we have mentioned above, the spreadsheet does not contain the information about the intended function of a functional block. The relation between a cell and the legends (e.g. that the cell D7 is related to the year 1986 and to the String Salaries) is provided by the function \( \psi_\Sigma : \Omega \to \Sigma \). The mapping from a cell in a grid to an element of \( \Lambda \) is provided by \( \psi_1 : \Omega \to \Lambda \).

**Example 5.3.** We explain the mapping for the cell \( D7 \) of Figure 1. This cell is related to \( a_{\text{Actual}} \in L_1, a_{1986} \in L_2 \) and \( a_{\text{Salaries}} \in L_3 \). We take an unused element \( a \in \Lambda \) and define \( \varphi_1(1,6,3) = a \). The mapping in Table 1 uses this mapping to define a functional block for this element by \( \varphi(a_{\text{Actual}}, a_{1986}, a_{\text{Salaries}}) = a \). By repeating this procedure for all cells in a given area we can map an unstructured area in a spreadsheet to a functional block structure which makes the underlying structure in the spreadsheet explicit.

### 6 Discussion

We have developed a new model which is able to represent the abstract structure of a spreadsheet and which satisfies our two maxims from Section 2. Even if we have not provided a formal proof that a concrete spreadsheet and an abstract spreadsheet model are somehow isomorphic, the mathematical notations allows such formal proofs, whereby such proofs need assumptions about some properties that are not defined here.

In future work we want to develop methods for a semi-automated extraction of the abstract structure for a concrete spreadsheet. Although an implementation of the abstract spreadsheet model is already integrated in the SisSi project, future work will analyse how this abstract model can be used in user interaction to prevent common errors or to understand the structure of the spreadsheet by connecting it to an ontology with contains the semantic of a spreadsheet. Instead of creating formulae in Excel which refer to different cells a user should be able to create a formula which refers to semantic objects in the ontology, like \( \text{Profit}: \text{Year} \to \text{Float} \), \( \text{Profit}(y) = \text{Revenues}(y) - \text{Expenses}(y) \). Afterwards an area in a spreadsheet (like B15:F15 in Figure 1) can be linked to that ontology term. Thereby the corresponding formulas in the cells can be created as concrete instances of the given ontology term or the existing formulae can be validated against the ontology term by using a theorem prover. For this kind of validation we need the abstract spreadsheet model as an intermediate layer, because on the one hand it contains all explicit formulas from a spreadsheet and on the other hand contains the information about the relation to the legends. In Example, without the knowledge that the elements for the cells B4, B13 and B15 are related to the year 1984 and to revenues, expenses and profit the formula in the cell B15 could not be validated against the ontology term \( \text{Profit}(y) = \text{Revenues}(y) - \text{Expenses}(y) \).

### Acknowledgments

This work was funded by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) under grant HU 737/6-1.

### References


RESUBMISSION

iTree: Skill-building User-centered Clarification Consultation Interfaces

Martina Freiberg and Frank Puppe
University of Würzburg
D-97074, Würzburg, Germany
freiberg|puppe@informatik.uni-wuerzburg.de

Abstract

Developing web-based, knowledge-based systems (wKBS) still challenges developers, mostly due to the inherent complexity of the overall task. The increased focus on knowledge base development/evaluation and consequent neglect of UI/interaction design and usability evaluation raises the need for a tailored wKBS development tool, leveraging the overall task while specifically supporting the latter activities. As an example for such a tool, we introduce the wKBS development tool ProKEt. With the help of that tool, we developed the novel UI concept interactive clarification tree (iTree) with skill-building ability, that specifically is suitable for clarification consultation systems. Also, we report a recent case study, where iTree was implemented for knowledge-based clarification consultation in the legal domain.

Keywords: Knowledge-based Systems, Clarification Consultation, UI Design, Skill-building UI, Usability Evaluation

1 Introduction

Despite increasing distribution in many domains, web-based knowledge-based systems (wKBS) still challenge developers: Development of appropriate knowledge bases alone is an effortful task in terms of time and money; thus, intentional UI/interaction design and usability evaluation activities remain rather neglected. Yet, wKBS are often applied in critical or specialized contexts—e.g., consultation in the medical or legal domain—where the chosen UI/interaction style can contribute strongly to either the success or the failure of the system. Thus, UI/interaction design and usability evaluation should rather be a key factor for wKBS development. This increases the need for a tailored software tool that fosters experimentation and evaluation of novel wKBS styles. We propose the tailored software tool ProKEt, that supports efficient affordable, UI/interaction design-focussed wKBS development while at the same time seamlessly integrating usability evaluation functionality. To the best of our knowledge there exists no previous work to date regarding similar tools.

Regarding consultation in contexts such as the legal or medical domain, it often can be valuable to not only have general consultation systems available that derive one or more diagnoses based on the user input, but additionally to have specialized clarification systems, for investigating only one distinct diagnosis—potentially pre-selected by general consultation systems or by the users themselves. In this paper, we introduce the interactive clarification tree (iTree) as a novel, hierarchical clarification UI/interaction style that we developed with the help of the tool ProKEt: iTree thereby is particularly suitable for a mixed, diverse user population and additionally provides skill-building ability. A first study in the course of a current project in the domain of legal consultation suggests general benefits of the proposed iTree UI style.

Related Work With regards to general KBS/wKBS development there exist various tailored software tools—such as JavaDON (Tomic et al., 2006), or KnowWE (Baumeister et al., 2011)—and methodologies—e.g., MIKE (Angele et al., 1998), or CommonKADS (Schreiber et al., 2001). However, such approaches still mostly focus on the design and evaluation of the knowledge base; in contrast, we propose ProKEt as tailored wKBS development tool that seamlessly couples efficient, agile wKBS development, creative experimentation regarding KBS/UI/interaction design, and semi-automated usability evaluation activities. ProKEt can be further seen as user-centered prototyping tool for wKBS—a concept defined by (Leichtenstern and Andre, 2010) as an all-in-one tool solution for enabling efficient, effective and satisfactorily design, evaluation and analysis of developed artifacts.

Probably due to the numerous benefits of web-based systems—e.g., availability, acceptance, or maintainability—to date an increasing number of knowledge-based/expert systems seems to be developed for the web—i.e., integrated in websites or as separate, complex web applications; recent examples are (Patil et al., 2009) or (Rahimi et al., 2007). However, such wKBS apparently are being developed in a rather ad hoc manner, not following systematic methods or processes, and not (re)using (neither providing) any patterns or best practices especially regarding the UI/interaction design—probably due to a general lack of scientific research in web-based expert systems, cf. (Duan et al., 2005). Similarly, wKBS developers seem to be individuals, performing all tasks required for developing and distributing a wKBS by themselves. This further increases the need for a tailored tool that not only renders overall wKBS development an efficient,
pragmatic task, but equally important specifically supports
design and experimentation with web-based UI/interaction
forms and their usability evaluation.

**Paper Structure**
The rest of the paper is organized as follows: In Section 2, we shortly introduce the tailored
wKBS development tool ProKEt. Afterwards, we discuss
* iTREE: a novel hierarchical UI concept for knowledge-based clarification consultation systems with skill-building ability in Section 3. We report on a recent case study in Section 4, where the proposed UI style was practically implemented for a wKBS in the legal domain. We conclude with a short summary of the presented research
and an outlook to prospective future work in Section 5.

## 2 ProKEt

ProKEt is a tailored, Prototyping and Knowledge systems Engineering tool for web-based, knowledge-based systems (wKBS), that additionally provides integrated support for various usability evaluation related activities. Thereby, ProKEt specifically supports web-based consultation and documentation systems, which can be developed equally well as (pure) prototypical demo systems and as fully-fledged systems for productive use. Thereby, extensible prototyping is put into action, facilitating a nearly effortless transition from prototype to productive system; for a more extensive introduction of the agile, extensible prototyping and engineering process with ProKEt, see (Freiberg et al., 2012). The main application logic is implemented in Java. The resulting artifacts are Servlet-based web applications, using HTML, String-Template, and CSS for UI creation, and JavaScript for interactivity. Regarding the knowledge representation, an XML-based specification is used for the pure prototypes, which can be directly cerated/edited with ProKEt itself. For productive systems, _d3web_ (URL d3web, 2012) knowledge bases are integrated and (mostly) replace the XML specification; the latter, however, can not directly be edited with ProKEt, thus in that case an external d3web-supporting tool such as KnowWE (Baumeister et al., 2011) is required. Yet recently, we implemented a mechanism to couple KnowWE and ProKEt artifacts, thus drastically improving and easing the workflow of UI/front-end development, knowledge base development and their integration into a productive wKBS.

For supporting the straightforward evaluation of its artifacts, ProKEt further allows for seamlessly integrating both qualitative and quantitative data collection both for prototypes and productive wKBS; this enables developers to assess the current development state in a favorable way at any time by conducting manifold, potentially iterative, evaluations. For qualitative data collection, ProKEt supports both the integration of form-based questionnaires/surveys—standards such as the SUS (Brooke, 1996) and the NasaTLX (Hart, 2006) are supported out of the box, but tailored own questionnaires can be added with no effort—and of anytime feedback—mechanisms for collecting free user feedback at any time during a wKBS session. Regarding quantitative data, ProKEt provides a tailored, mouse click and keyboard event logging mechanism that records all relevant actions during wKBS sessions. Based on that data, ProKEt furthermore automatically can calculate a bunch of known usability metrics—such as Success Rate, or Average Task Time—proposed e.g. by (Constantine and Lockwood, 1999), but it is equally well possible to just export qualitative and quantitative data into a standard CSV format for further investigation with external tools, e.g., standard spreadsheet calculation or advanced statistical software. A more detailed introduction of that usability extension of ProKEt can be found in (Freiberg and Puppe, 2012).

## 3 iTREE for Clarification

ProKEt particularly supports the development of consultation- and documentation systems. A consultation system thereby provides decision support in a particular domain based on given user input, whereas a documentation system contrastingly focusses on supporting uniform, efficient and high quality data entry. In this paper, we propose the interactive clarification tree (iTREE) UI style specifically for clarification systems as a sub-class of consultation systems.

### 3.1 Clarification Consultation

As a subarea of classification, clarification relates to hypothesize-and-test as follows: Separate, general multiplex consultation systems can be applied first for narrowing the complete set of potential diagnoses/hypotheses down to one or several most suitable elements (hypothesize step); each of those hypotheses can then be further investigated by a corresponding clarification module (test step). As shortcut, users could alternatively start directly with a clarification system for a chosen hypothesis themselves.

### 3.2 iTREE: Skill-building Clarification

We propose iTREE as a novel UI with skill-building ability that fosters an efficient and usable user experience in the context of clarification systems.

<table>
<thead>
<tr>
<th>Core Issue to Clarify…</th>
<th>Question 1 (Rating Core Issue)</th>
<th>Question 2 (Rating Core Issue)</th>
<th>Question 2.1 (Rating Question 2)</th>
<th>Question 2.2 (Rating Question 2)</th>
<th>Question 3 (Rating Core Issue)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>Y</td>
<td>N</td>
<td>?</td>
<td>N</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Y</td>
<td>N</td>
<td>?</td>
<td>N</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Y</td>
<td>N</td>
<td>?</td>
<td>N</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Figure 1: Schematic drawing of the iTREE UI style

Figure 1 presents a schematic drawing of iTREE for clarification systems. The core issue to be rated is presented as root element of the hierarchical tree structure (Figure 1, Core Issue). Its rating is derived from the ratings of any desired number of top-level questions, placed directly underneath the core issue (Figure 1, Quest.1, Quest.2, Quest.3). Questions are a tailored form of yes/no questions with an additional value Neutral/Uncertain; provided answers further can be withdrawn/adapted at any time, indicated by the X button. The current implementation allows three possible abstract ratings for the core issue as well as for all questions: Confirmed, uncertain/neutral and rejected, which correspond to the answers Yes, ?, and No per default. Some domains may require to swap that mapping for particular questions in favor of a more understandable question wording. For example, see Figure 2 which depicts the current implementation of the iTREE UI style: The core issue is confirmed (rating: yes) if the cancellation was NOT prohibited due to time limitations;
in that case, the swapped yes/no mapping allows for rewording the question as depicted, which is much clearer than its negated alternative. In case the user cannot answer a question directly, more detailed refinement questions— if available—can be retrieved for the current element, represented by the D button in the scheme and by the arrow in Figure 2; as example, the second top-level question in Figure 2 contains two refining questions, which list in more detail the conditions which confirm/reject its parent question. Question ratings are always propagated from inner levels of the hierarchy up to the topmost question(s) by either AND or OR connections. Let \( pm \) be a parent node and \( cn_p \) a child node of \( pm \); for calculating the rating of \( pm \), the following rules apply:

AND nodes:
- IF \( \exists cn_p = \text{no} \) THEN \( pm = \text{no} \)
- IF \( \exists cn_p = \text{yes} \) THEN \( pm = \text{yes} \)
- IF \( \exists cn_p = \text{rated AND} \ \exists cn_p = \text{neutral} \ \exists cn_p = \text{no} \) THEN \( pm = \text{neutral} \)
- IF \( \exists cn_p = \text{unknown} \ \exists cn_p = \text{neutral} \ \exists cn_p = \text{no} \) THEN \( pm = \text{unknown} \)

OR nodes:
- IF \( \exists cn_p = \text{yes} \) THEN \( pm = \text{yes} \)
- IF \( \exists cn_p = \text{no} \) THEN \( pm = \text{no} \)
- IF \( \exists cn_p = \text{rated AND} \ \exists cn_p = \text{neutral} \ \exists cn_p = \text{yes} \) THEN \( pm = \text{neutral} \)
- IF \( \exists cn_p = \text{unknown AND} \ \exists cn_p = \text{neutral} \ \exists cn_p = \text{yes} \) THEN \( pm = \text{unknown} \)

This means, e.g., that the core issue in Figure 2 is rated yes only if all of its children are rated yes, as those are connected to the core issue by AND (second rule above); likewise, cancellation prohibited due to time limitation is rated yes as soon as one of its children is rated yes (fifth rule above) due to the OR connection. One advantage of iTree is the suitability for a diverse user population—i.e., users with different background and expertise might be able to profit from the same system. This is achieved by the possibility to derive the solution rating both by answering more abstract top-level questions (domain specialist level) or by stepping into more refined, elaborate questions (less expertise required) in iTree. By the visual representation of the knowledge base structure, moreover a form of focus-and-context view is created: Not only the currently active/processed question(s) are visible, but also surrounding elements are indicated—limited only by the display size. As the user thus can visually trace the result of an answer by the distinct presentation of the questions and their current state, that is propagated all throughout the tree, the core issue rating becomes more transparent. The chosen visual representation of the knowledge further supports users in gaining a thorough understanding of the investigated core issue and the coherences between its clarifying questions and the core issue itself. Thus, users acquire additional knowledge by means of the system, yet are also enabled to bring in their existing knowledge for potentially shortening the clarification session or for focussing on only those parts in detail that are rather unclear. Together with optional, auxiliary information that can be integrated for each of the elements (not contained in the scheme—see e.g. the auxiliary information panel in Figure 3, E), iTree specifically can serve as a skill-building UI type.

4 Case Study—JuriSearch

At the beginning of 2012, the JuriSearch project was initiated as a cooperation between the university of Würzburg and the Reno Star corporation (partly founded by the Free State of Bavaria). JuriSearch aims at building a wKBS for the legal domain: The target system is intended to integrate both a standard consultation (entrance) module—hypothesize—and various clarification modules for each potential core issue—test—as to provide encompassing advice on various legal topics, such as the right of cancelation or the law of tenantry. Potential target users are diverse, ranging from legal laymen—searching for a basic understanding/estimation of their case to (fresh) lawyers seeking for guidance regarding legal (sub)domain(s) that are not exactly their special field of work. Those framing conditions provided a perfect opportunity to implement and evaluate the iTree UI style. Therefore, a comparative study with iTree and a more common, conversational UI style was conducted; the latter was implemented as a one-question UI style (oneQ). In contrast to the free, explorative interaction with iTree, oneQ is based on the metaphor of a conversation: The system always presents only the one suitable next question at a time, thus imitating a strict dialogue between a user and the system. Yet, both UI types are based on the exact same knowledge base with their core difference being the presentation of the questions: Hierarchical tree (iTree) vs. single question (oneQ). Refinement questions are as well available in oneQ; yet there, the former current question is folded and the first of the refinement level questions is presented, thereby destroying much of the 'contextual knowledge' that iTree facilitates by always presenting all questions of the current hierarchy level in addition to the surrounding, further structure (limited by display size only). Figure 3 presents the iTree implementation in JuriSearch (A-F) as well as the alternative, conversational oneQ UI (G).

4.1 User Study—Framing Conditions

21 members from our department—all male, mostly between 25 and 35 years—participated in the first study; as computer scientists, they all were versed in general computer and web system usage, yet in most cases had little to no experience regarding the specific wKBS types, and no experience regarding the target domain at all. Two exemplary problem descriptions from the domain of cancellation were created, and participants were asked to solve one problem with iTree and the other with oneQ; to avoid biased results due to the sequence of using the UIs, that sequence was altered between participants. The study was conducted remotely: The test systems were deployed on a specifically configured server—enabling the integrated logging and feedback/questionnaire mechanisms—and the participants were given all required instruction material per email.
4.2 User Study—Results & Discussion

The collected log-data revealed a general applicability of the iTree concept for implementing a clarification wKBS UI in the legal domain and specifically the following results: First, iTree exhibited an average task time of 13m 38s±6m 49s in contrast to 10m 39s±5m 49s for oneQ (by a narrow margin statistically not significant on a one-sided unpaired t-test, \(p=0.068\)). The higher task time of iTree could possibly be explained by its ability to provide intuitively for free, extensive exploration of the system. Yet on the other hand, task time should not be overrated at all, here; the extent of usage of the test systems depended in larger parts on a) the reading speed of the participants regarding the questions and explanations, b) the usage conditions (during daily job routine vs. after end of work) which, due to the setting of the remote study could not be controlled strictly, and c) the potentially already existing knowledge regarding the problem at hand, in turn leading to highly subjective task time results between users. Regarding the success- and error rate, a case was classified successful, if the correct rating of the core issue was derived by the user with the respective system type, and not successful if either the wrong or no solution was found. For iTree, the success rate was 42.86% and 38.1% for oneQ (both: no statistical significance on a one-sided binomial test with \(p=0.11\) and \(p=0.16\); along with subjective user feedback, this clearly indicates the need to rework the knowledge base contents/structure for yielding better results. Furthermore, both anytime- and questionnaire-based feedback were collected as qualitative user data. The first remarkable finding was the fact, that iTree nearly concordantly was perceived more intuitively usable, and that it thus further was reported to be the preferred UI type by 81% of the study participants, whereas oneQ only was preferred by 14% and no preference was stated by 5%; this is statistically significant on a \(\chi^2\) test with \(p<0.05\) and with an anticipated distribution of 50% (iTree), 30% (oneQ), and 20% (both equally). One possible explanation might be the specific characteristics of the participant population, that—as computer scientists—might simply be used to tree representations and thus perceived iTree as naturally more intuitive to use. Regarding further subjective (questionnaire) topics iTree scored better all over; on a scale from 0 (worst) to 6 (best) the results were: Comprehensibility of the system reactions 4.43±1.54 (iTree) vs. 2.76±1.45 (oneQ) or of the derived results 4.53±1.54 (iTree) vs. 3.33±1.85 (oneQ), and the mediation of domain knowledge to the user 4.05±1.32 (iTree) vs. 2.95±1.72 (oneQ); those differences are all statistically significant using an unpaired, one-sided \(t\)-test with \(p\leq0.05\). Especially the latter value affirmed our assumption that iTree particularly evinces skill-building abilities. Additional insights from anytime feedback included: The wording of the questions was perceived as incomprehensible/empowering in 11 cases (52%) due to often used duplicate negations and legal specialist language, probably further aggravated by the fact that the chosen participants were legal laymen and thus not at all familiar with legal terms and language: also, the hierarchical structure and representation of the knowledge base—that followed the legal subsumption logic—was perceived unfavorable. In such a hierarchy/sequence, the questions most interesting for legal laymen appear far down while at the same time more abstract concepts are contained at the upper levels; this led to (laymen) users having difficulties to make sense of the concepts at the top/beginning of the hierarchy/questioning sequence. A solution to this issue might be a complete restructuring of the knowledge base,

Figure 3: JuriSearch clarification module as iTree (large) and oneQ UI (small). AND/OR rules for rating the (sub-)questions (A) are visually represented; reversed question example (underneath B); dummy node example, only serving for rule connection modeling (above B); four simple buttons (B) for rating the questions (Y:yes—N:no—?:neutral—X:empty:retract), rating highlighted by background color (C); core issue rating prominently displayed and updated continuously (D); additional information displayed in separate panel when mouse-overflowing question (E); anytime feedback/data collection features integrated in UI header (F); clarification core component in oneQ style (G) always displays current active question with additional-information panel, previously answered questions remain presented in a more condensed view.
so that the most relevant questions and distinctions—from the users’ point of view, e.g., typical reasons for dismissal, size of company, etc.—also appear on rather top levels, sure posing a difficult trade off between legal correctness/schemaic thinking and understandability; yet it appears that we could greatly contribute to tailoring the UI to the users in enabling them to bring in their own perspective and knowledge in the dialog. Thus a further refinement of the knowledge base with regards to a clear, easily understandable language and structure turned out indispensable. Another interesting finding was the fact, that in 4 (19%) cases, the real meaning of the ‘-?’ button as an answer alternative was not grasped; users rather expected the system to display more elaborate explanations on the issue at hand or to open up the next refinement level of the questions instead of receiving just a rating of the current question. Similarly, the ‘X/empty’ button—designated to clearing a previously entered answer—was not intuitively understood in 3 (14%) cases.

5 Conclusions

In this paper, we claimed the importance of a careful UI/interaction design for web-based, knowledge-based systems. Regarding the consultation systems’ sub-class clarification systems, we suggested iTree as novel UI/interaction style for increased efficiency and usability. In a first comparative user study from the legal domain, an initial iTree prototype as well as an alternative, one-question style prototype were implemented using the prototyping and knowledge systems engineering tool ProKEt. The results suggest, that iTree generally is a favorable UI style for clarification systems, that supports free, explorative system usage and thus provides skill-building potential on the side of the users. Yet, the study also showed the need to rework the knowledge base of the system, regarding both the question wording as well as their structuring. One assumption requiring further studies is, that the legal iTree at its current state is satisfying for legal experts, whereas a restructured system could be more appropriate for non-expert users. Additionally, we plan on developing and evaluating similar iTree systems for the medical domain. This raises the requirement of more fine-granular rating options, e.g., by scoring rules. Finally, further experimentation with potential UI enhancements is intended to help improve the iTree concept; one such idea is the integration of an interactive system state preview that is overlaid when mouse-overing the respective answer option.

Acknowledgements

We thank the RenoStar (Großwallstadt, GER) corporation for valuable discussions and cooperation.

References


RESUBMISSION
Confidence in Workflow Adaptation

Mirjam Minor, Mohd. Siblee Islam and Pol Schumacher
University of Trier
Department of Business Information Systems II D-54286 Trier, Germany
[minor|islam|pol.schumacher]@uni-trier.de

Abstract
This paper is on assessing the quality of adaptation results by a novel confidence measure. The confidence is computed by finding evidence for partial solutions from introspection of a huge case base. We assume that an adaptation result can be decomposed into portions, that the provenance information for the portions is available. The adaptation result is reduced to such portions of the solution that have been affected by the change. Furthermore, we assume that a similarity measure for retrieving the portions from a case base can be specified and that a huge case base is available providing a solution space. The occurrence of each portion of the reduced solution in the case base is investigated during an additional retrieval phase after having adapted the case. Based on this idea of retrieving portions, we introduce a general confidence measure for adaptation results. It is implemented in the area of workflow adaptation. A graph-based representation of cases is used. The adapted workflow is reduced to a set of sub-graphs affected by the change. Similarity measures are specified for a graph matching method that implements the introspection of the case base. Experimental results on workflow adaptations from the cooking domain show the feasibility of the approach. The values of the confidence measure have been evaluated for three case bases with a size of 200, 2,000, and 20,000 cases each by comparing them with an expert assessment.
RESUBMISSION
Semantic Alliance: A Framework for Semantic Allies

Catalin David, Constantin Jucovschi, Andrea Kohlhase and Michael Kohlhase
Computer Science
Jacobs University Bremen
http://kwarc.info

Abstract

We present an architecture and software framework for semantic allies: Semantic systems that complement existing software applications with semantic services and interactions based on a background ontology. On the one hand, our Semantic Alliance framework follows an invasive approach: Users can profit from semantic technology without having to leave their accustomed workflows and tools. On the other hand, Semantic Alliance offers a largely application-independent way of extending existing (open API) applications with MKM technologies. The Semantic Alliance framework presented in this paper consists of three components: i.) a universal semantic interaction manager for given abstract document types, ii.) a set of thin APIs realized as invasive extensions to particular applications, and iii.) a set of renderer components for existing semantic services. We validate the Semantic Alliance approach by instantiating it with a spreadsheet-specific interaction manager, thin APIs for LibreOffice Calc 3.4 and MS Excel’10.