KREM: A Generic Knowledge-based Framework for Problem Solving in Engineering
Proposal and Case Studies

Cecilia Zanni-Merk
INSA de Strasbourg / ICube Laboratory (BFO Team), Illkirch, France

Keywords: Knowledge Technologies, Ontologies, Reasoning, Experience, Meta-knowledge.

Abstract: This article presents a generic knowledge-based framework for problem solving in Engineering, in a broad sense. After a discussion about the drawbacks of the traditional architecture used for deploying knowledge-based systems (KBS), the KREM (Knowledge, Rules, Experience, Meta-Knowledge) architecture is presented. The novelty of the proposal comes from the inclusion of experience capitalization and of meta-knowledge use into the previously discussed traditional architecture. KREM improves the efficiency of classic KBSs, as it permits to deal with incomplete expert knowledge models, by progressively completing them, learning with experience. Also, the use of meta-knowledge can steer their execution more efficiently. This framework has been successfully used in different projects. Here, the architecture of the KREM model is presented along with some implementation issues and three case studies are discussed.

1 INTRODUCTION

Nowadays, organizations are becoming increasingly knowledge-intensive and collaborative (Ouertani et al., 2011). However, large parts of useful knowledge are hidden and not readily available. A growing number of companies have realized that a tool that effectively enables the capture, representation, retrieval, and reuse of knowledge is the key to supporting various organizational decisions (Davenport and Prusak, 2000; Leistner, 2010). And this is one of the main interests of Knowledge Technologies.

Knowledge Technologies are computer-based techniques and tools that provide a richer and more intelligent use of information. Much of their power comes from the way they combine ideas and applications from a number of fields: Psychology, Philosophy, Artificial Intelligence, Engineering, Business Studies, Computer Science and Web Technologies (Milton, 2008).

Knowledge Technologies, as the name suggests, are about doing things with knowledge. For example:

- Identifying what knowledge is important to an organisation;
- Deciding what knowledge needs to be captured to provide an appropriate solution to a real-world problem;
- Capturing and integrating knowledge from experts or repositories (such as texts or databases, for example);
- Representing and storing knowledge in ways that provide ease of access, navigation, understanding, maintenance and re-use;

According to (Milton, 2008), Knowledge Technologies include: Knowledge Based Systems, Natural Language Processing, Data Mining, Semantic Technologies, Case Based Reasoning and Intelligent Agents, among others. We are interested in a subset of the technologies presented by (Milton, 2008), but use them in a slightly different way.

One of the main advantages of using Knowledge Technologies for problem solving in Engineering is that they facilitate decision-making. Better decisions can be made when people have the right information at the right moment. When information is out-of date, incoherent, irrelevant, hard to find and hard to integrate, then the decision-making process is made more difficult. The information required to make good decisions often exists but tends to be scattered in various locations and stored in various formats (such as databases or plain text). Knowledge Technologies provide ways of finding, merging and exposing the information required to make informed decisions.

Knowledge Technologies are dealt with methods

Zanni-Merk, C..
KREM: A Generic Knowledge-based Framework for Problem Solving in Engineering - Proposal and Case Studies.
In *Proceedings of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2015) - Volume 2: KEOD*, pages 381-388
ISBN: 978-989-758-158-8

coming from the field of Knowledge Engineering, whose main goal is the development of Knowledge-Based Systems (KBSs).

The classical architecture of a KBS (discussed in Section 2) does not always provide satisfactory results when solving a problem. In fact, the knowledge models needed for their implementation are often incomplete. Therefore, we propose here a novel architecture to deal with these drawbacks by the inclusion of the capitalization of the experience learnt with the use of the KBS. The use of meta-knowledge is also integrated to deal with the whole.

As stated above, Section 2 presents an introduction to knowledge-based systems. Section 3 deals with the capitalization of experience. Sections 4 and 5 presents the KREM model with some notes about its implementation. Section 6 describes three different applications of KREM; and finally, Section 7 gives our conclusions.

2 KNOWLEDGE-BASED SYSTEMS

One of the main goals of using an approach based on Knowledge Engineering methods for problem solving is the development of KBSs. They have a computational model of some domain of interest in which symbols serve as surrogates for real world domain artefacts, such as physical objects, events, relationships, etc (Sowa, 2000).

A knowledge-based system maintains a knowledge base which stores the symbols of the computational model in form of statements about the domain, and performs reasoning by manipulating these symbols. Applications can base their decisions on domain-relevant questions posed to a knowledge base. They are programmed to solve problems in a similar way to that of an expert practitioner, e.g. make inferences based on the case at hand and adopt the right strategy to solve the problem; they can deal with incomplete information by making requests for further information or by making intelligent guesses at the answer (just as an expert has to do when there is limited information); they have a user interface that makes intelligent requests for information and can explain how it has arrived at its answers; and finally, the development can be made cost-effective by re-using generic structures, rules and problem-solving methods.

The components usually found in a KBS are: a knowledge base that contains domain-specific information, structures and rules; a working memory (also known as a blackboard) which holds case-specific data (e.g. facts about the initial problem and intermediate results); an inference engine (or reasoning engine) that controls and directs the solving of problems by making inferences, i.e. it uses the knowledge base to alter the contents of the working memory; an interface to other computer systems and/or to human users; and an editor that allows a knowledge engineer or domain expert to inspect and update the knowledge base.

The knowledge-base of a KBS needs a more explicit description. One of its components is a formal conceptual model of the domain of interest. Among the existing formal conceptual models, ontologies are formalized representations of vocabularies that are specific to a certain area. The dominating definition of an ontology is the following: *An ontology is a formal explicit specification of a shared conceptualisation of a domain of interest*, based on (Gruber, 1993).

Technically, the principal constituents of an ontology are *concepts*, *relations* and *instances*. They are commonly used with a set of rules that are chained to simulate the reasoning of a human expert. Rules come in the form of "if-then" constructs and allow to express various kinds of complex statements and allow different types of reasoning about the facts in the working memory based on the domain knowledge in the ontology.

In summary, a knowledge based system contains a conceptual yet executable model of an application domain. It is made machine-interpretable by means of knowledge representation techniques and can therefore be used by applications to base decisions on reasoning about domain knowledge.

The traditional architecture that we have just discussed has drawbacks, associated, mainly, with the difficulties that appear during the knowledge elicitation process from experts; and also with the non-completeness of the formal conceptual model obtained after the elicitation (Milton, 2008).

In fact, as the knowledge base can be incomplete, there could be problems that this traditional architecture cannot solve. Reasoning and analysis of this incomplete knowledge implies that it is needed to take advantage of the experience acquired from the interventions of human experts when the traditional system does not lead to satisfactory results; that is, some kind of *capitalization of experience* is needed.

3 ON THE CAPITALIZATION OF EXPERIENCE

Set of Experience Knowledge Structure (SOEKS) is an experience-based knowledge representation that

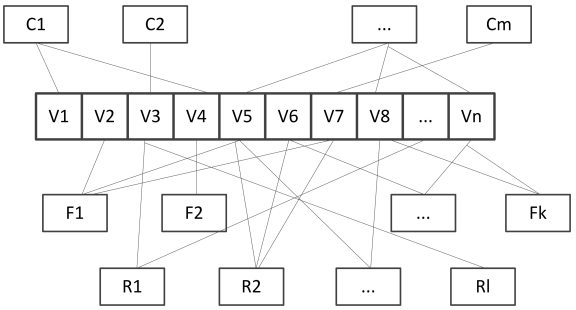

Figure 1: A SOEKS.

can store uncertain and incomplete data and make qualitative and quantitative extractions of knowledge from the available information, which can often be unstructured, semi-structured, fuzzy, and vague (Sanín and Szczerbicki, 2009; Sanín et al., 2007).

Additionally, SOEKS can be shaped in an extensive understandable and transportable language such as Extensible Markup Language (XML) or Ontology Web Language (OWL) (Grau et al., 2008). XML and OWL representation of SOE allows knowledge to be exchanged quickly and securely between applications and systems. The SOEKS has been designed to collect experiences and knowledge from multiple applications that are assembled as formal decision events (Sanín and Szczerbicki, 2009). This collected knowledge assists organizations in making precise decisions, predictions, and recommendations, and it is a dynamic structure that is dependent on the information and data that it has received.

A SOEKS has four components (Figure 1) (Sanín and Szczerbicki, 2009): variables, functions, constraints, and rules. Each formal decision event can be stored in a combined structure of those four components of the SOEKS.

Variables usually involve representing knowledge using an attribute-value language. This is a traditional approach from the origin of knowledge representation. Variables are related among them in the shape of functions. Functions, the second component, describe associations between variables. Therefore, the set of experience uses functions and establishes links among the variables constructing multi objective goals. Constraints are another form expressing relationships among the variables. A constraint is a restriction of the feasible solutions in a decision problem, and limits the performance of a system with respect to its goals. Finally, rules are suitable for representing inferences or for associating actions with conditions under which the actions should be performed. They are conditional relationships of the universe of variables.

4 THE KREM MODEL

Taking into account the points discussed in the previous sections, we propose a modular architecture, called KREM (Knowledge, Rules, Experience and Meta-Knowledge), to manage the complexity of developing KBSs, to try to solve the aforementioned problems and to incorporate the capitalization of experience with the goal of improving decision-making.

Because to be effective, decision-making must result from reasoning and analysis of this knowledge, also taking into account the experience and expertise of decision-makers. As a consequence, it is needed to capitalize them to take advantage of the experience acquired from the interventions of human experts when the traditional system does not lead to satisfactory results.

The use of meta-knowledge to steer the execution of the whole system is also necessary. Meta-knowledge is knowledge about domain knowledge, about rules or about experience. This meta-knowledge can take the form of context, culture or protocols to use this knowledge. Context is information that characterizes a situation in relation to interaction among human-beings, applications and their environment, and can be of four types: identity, place, status or time (Dey et al., 2001). Culture meta-knowledge tries to take into account the fact that decisions are made differently depending on the country or culture (Meyer, 2014). And finally protocols may include strategies or problem-solving heuristics for the task to be done (for example, in the case of medical diagnosis, the protocols used by physicians change according to the type of symptoms or the suspected illness).

Therefore, the proposed components of the architecture are (Figure 2):

- The *Knowledge* component that contains the domain knowledge to operate, by means of different domain ontologies to be developed.
- The *Rules* component that allows different types of reasoning (monotone, spatial, temporal, fuzzy, or other) depending on the application.
- The *Experience* component that allows the capitalization and reuse of prior knowledge.
- The *Meta-knowledge* component, including knowledge about the other three bricks that depends on the problem. For example, in a medical diagnosis problem, this component may include clinical protocols used by physicians.

The way the domain knowledge is formalized will shape the way the rules are expressed. Experience will come complete the available knowledge and

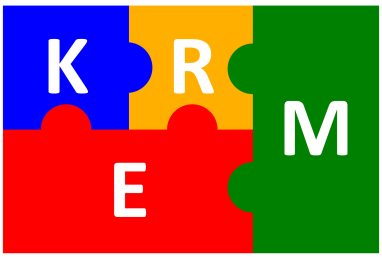

Figure 2: The KREM architecture with its four interrelated components: knowledge, rules, experience and meta-knowledge.

rules. Finally, meta-knowledge will directly interact with the rules and the experience to indicate which rules (coming from experience or from the initial rule set) need to be launched according to the context of the problem to solve.

A modular architecture, such as KREM, is one of the main architectural design pattern for large and complex systems. In this pattern, each module or component has a specific functionality providing separation of concerns that, in turn, support reuse or replacement (*i.e.* changes in a single module would not affect the others, permitting the continuous operation of the system). Moreover the communication between modules needs to be based on well defined interfaces to provide low coupling.

The KREM model has already been successfully used for very specific applications; the semantic analysis of urban satellite images, the diagnosis of small and medium enterprises or the formalization of the Lean enterprise. We will discuss these applications in Section 6.

5 SOME NOTES ABOUT THE IMPLEMENTATION

The implement the KREM architecture, it is necessary, at least, to combine rules formalisms with description logics (this is the classic architecture of a KBS). There are several possibilities to accomplish this. But our choice has been to have hybrid systems interfacing logic programming systems and description logic systems.

As explained before, the knowledge module contains the domain ontologies that have been developed for each project. These developments are made in OWL (Grau et al., 2008). The Web Ontology Language (OWL) is an integral component of the Semantic Web, as it can be used to write ontologies or formal vocabularies which form the basis for semantic web data mark-up and exchange.

The Protégé[1] editor in several versions has been our preferred choice for the design of the ontologies. Protégé is usually used to construct domain models and knowledge-based applications with ontologies because of possessing a Java-based Application Programming Interface (API) and being easily extended by a plug-in architecture.

Concerning the rules, we have also tested several approaches. The first natural choice was the use of SWRL (Semantic Web Rule Language).

SWRL is a proposal by the W3C to extend OWL with rules. It is based on a combination of some sub-languages of OWL with the Unary/Binary Datalog RuleML sub-languages of the Rule Markup Language[2]. The proposal extends the OWL abstract syntax to include the syntax of these rules and the OWL model-theoretic semantics to provide a formal meaning for ontologies that include rules written in this syntax (Golbreich, 2004).

However, some of the constraints of SWRL (mainly the lack of input/output features and the impossibility to create new individuals in the ontologies "on the fly") made us begin investigating other possibilities. And our choice went to Jess.

Jess (Java Expert System Shell) is a rule engine for the Java platform, developed by Ernest Friedman-Hill of Sandia National Labs since 1995. Jess supports the development of rule-based expert systems which can be tightly coupled to code written in Java language. Compared with the traditional matching mechanism - only one loop for all - the Jess rule engine continuously matches facts against rules to infer conclusions, which result in actions. Thus rules can modify the collection of facts (Hill, 2003).

Unfortunately, Jess also suffers from some issues, mainly regarding its licensing, so we are currently investigating some other open-source inference engines, such as Drools [3].

Concerning the implementation of experience, we are using the SOEKS in its ontology form. Figure 3 shows part of the SOEKS class hierarchy with the four components involved in decision making events: variables, functions, constraints and rules.

What is important to remark concerning the interaction between the experience module and the others is that the reasoning process in the rules component enables the evolution of the initial set of rules with experience, as it was proposed by (Toro et al., 2012). More precisely, every time the native reasoner is executed, the system will store both the output of the reasoner and the final decision, if any, made by the

[1] http://protege.stanford.edu
[2] http://www.ruleml.org.
[3] http://www.drools.org

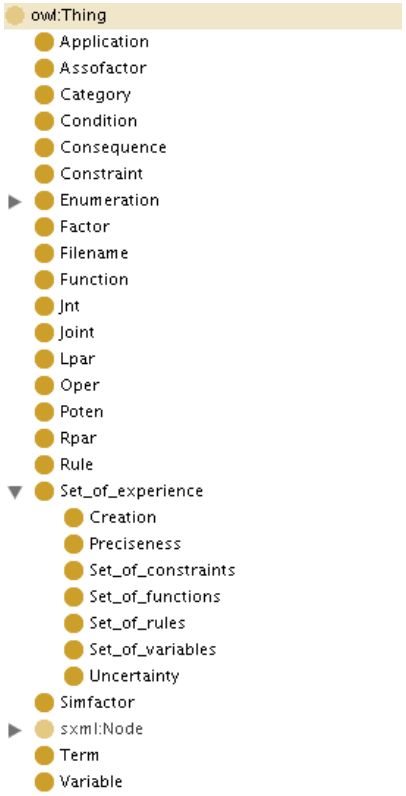

Figure 3: The SOEKS ontology.

expert. Decisions that do not follow the proposition of the reasoner produce an evolution of the set of experience rules.

Finally, concerning Meta-Knowledge, we are currently investigating the implementation of context, in a broad sense. Context has been studied carefully for a long time in areas like knowledge-based and ubiquitous systems, either for handling the complex knowledge in a dynamic manner (McCarthy, 1993; Sowa, 1995; Giunchiglia et al., 2012) or to provide smarter human interfaces (Dey et al., 2001). However, there is no agreement on a concise definition of context in these areas. (Bazire and Brézillon, 2005), for example, proposed the components of context as the user and the observer involved, as well as the items, the environment, and other related contexts. (Porzel, 2010) states that context helps when detecting semantic relations to provide extra information and correct interpretations for applications.

Diverse representations of context exist in different research areas. (McCarthy, 1993) uses a term `c` representing context, and the formal representation of a proposition `p` is true in the context `c` is represented as `ist(c, p)`. As described above, (Dey et al., 2001) define context as any information that characterizes a situation for context-aware applications, while (Porzel, 2010) refers to a context work, where a model of context contains components and the different relations of the components. The components are the user, an item, and the observer in the environment. Relations here include not only the relations between the components, but also the relations to other contexts.

We choose to represent context in the meta-knowledge module in a way similar to the one used by (McCarthy, 1993). An ontology enables the identification of the context for a certain agent in the application, and this agent will reason with a subset of all the rules in its Rules or Experience modules, that will be chosen with the help of the above mentioned predicate `ist`, once a judicious interpretation of it has been chosen.

6 SOME APPLICATIONS OF KREM

In this section, we will present some applications of KREM in diverse fields, showing the degree of development of the components and giving pointers to our publications for the interested reader.

6.1 Diagnosis of SMEs

The MAEOS project (modeling of business advice for organizational and strategic development of SMEs in Alsace) is the result of a collaboration between AEM Conseil (Alsatian SME working on business advice to small and very small companies) and our laboratory. Its main goal was to develop a software prototype for acquisition and processing of information collected by AEM Conseil from its clients to improve its business advice.

To do this, a set of knowledge bases relating to business management has been built and used to characterize the client companies. This characterization was then augmented by expert knowledge provided by AEM Conseil obtained during its professional practice. As this is a multidisciplinary project about Management Science and Knowledge Engineering, one of the main obstacles is the combined use of diverse and sometimes contradictory knowledge, this is inherent to models that come from Management Science; different authors (or the same author in different publications) may express different views on the same situation.

Several domain ontologies were developed in the Knowledge module of KREM and each ontology represents a conceptualization of the knowledge of a specific field in the context of SME (e.g. organization,

production, strategy, finance, etc.). Although an ontological study was conducted to provide the necessary theoretical foundations (Renaud et al., 2009; Renaud et al., 2010), the majority of the existing ontologies provide formal exhaustive models that could be applied to our purposes, but not many among them include reasoning rules to permit analysis or diagnosis. Consequently, our own ontologies and rules bases needed to be developed.

This Rules brick of KREM manages different types of rules to allow to reason about the instances of the classes in the domain ontologies of the knowledge module. To exploit these rules, a prototype of a multi-agent system (Zanni-Merk et al., 2011), capable of manipulating modular ontologies and reasoning with the Jess inference engine, has been developed.

Three types of rules appear in this component:

- Rules for diagnosis to obtain the analysis of the current situation of the SME.
- Recommendation rules providing hints to improve the current situation of the company.
- "Bridge" rules to ensure the semantic equivalence among concepts belonging to different ontologies.

Our latest works permitted the formalization of the first three bricks of the KREM architecture. We have also begun to explore meta-knowledge that could be included into the fourth module (Gartiser et al., 2014).

In fact, making a diagnosis of the situation of SMEs involves identifying a set of characteristics that are "present" in its current state and also identifying another set of "missing" features (the desired state). In this project, we need to effectively assess the current state and decide on actions necessary to achieve the desired state. The current and desired states are formalized as a SOEKS, and specific similarity measures to compare the current and ideal situations of the SME have been developed. The Meta-Knowledge module includes diagnosis models (e.g., about correct or incorrect behavior of companies) and criteria recommendation according to the experience of the consultant. In addition, in their daily tasks, consultants follow certain unwritten protocols (theoretical or not) to diagnose businesses that are also formalized here.

6.2 Semantic Analysis and Interpretation of Remote Sensed Images

The availability of very high resolution satellite images allows to envisage the identification of complex urban objects such as composite objects (e.g. neighborhoods, residential areas).

To extract and classify these urban objects, it is relevant to design image analysis methods based on a "thematic" modelling of these objects to extract and label them. Using formal ontologies seems a judicious choice. Our previous works (Cravero et al., 2012; di Sciascio M. et al., 2013; de Bertrand de Beuvron et al., 2013) show that this can limit the involvement of experts in time-consuming and unrewarding tasks such as labelling a large set of objects examples.

Taking advantage of those previous works, we have proposed the formalization of a new semi-automatic semantic approach for classification of urban objects from satellite images, using the KREM architecture (Zanni-Merk et al., 2015).

The Experience and Meta-Knowledge components of KREM are particularly useful for this application. The Experience component capitalizes the experience acquired from experts in the knowledge domain. After running the classification step and failing to identify some segmented regions, the statements contained in this brick should be queried in order to derive non-usual membership. For instance, if an unidentified region corresponds to a swimming pool in a backyard, then some additional knowledge based on the expert's experience could help to recognize it in this last instance. The Meta-Knowledge module includes knowledge about the use of the other components. In fact, according to the context of each image (latitude, longitude, season), different sets of rules could be launched to take into account that context. For example, the radiometric characteristics that allow the identification of vegetation in an image change according to the season.

6.3 Towards a Formal Model of the LEAN Enterprise

The goal of this project is to describe the characteristics of the Lean Enterprise and make the case for modelling it in order to reproduce its successful practices more easily.

The literature contains many good descriptions of the Toyota Production System and Lean in general (Womack et al., 1991; Byrne and Womack, 2012), but no formal model that we can build upon. We propose, then, to follow the KREM model: the K (Knowledge) component will include domain knowledge about Lean in the form of several ontologies, the R (Rules) component will be expressed by probabilistic rules, the E (Experience) component will describe the practices (Kata) and the M (Meta-knowledge) component will describe the context of the application of Lean (different types of companies or cultural environments, for example).

Our first works (Masai et al., 2015) permitted to verify the feasibility of the approach, by developing two complementary ontologies of Lean: the HOT (House Of Toyota production system or House Of TPS) ontology, which structures the core issues in Lean Management, and the ontology for Hoshin Kanri, which represents the entities necessary to model this specific process. The Hoshin Kanri process is of particular interest because it displays the behaviour of the agents (the employees in the house of Toyota production system) at various levels in the organisation going back and forth.

Concerning the Rules component, because the behaviour of enterprises experiences a high level of variability according to the current context while complying with specific management rules, they should be modelled as stochastic processes. In the previously cited work, we have chosen to model these stochastic processes with graphical representations of Markov Chains (Norris, 1998). We are working on the translation of these probabilistic graphical models into probabilistic rules (Malhotra, 2001), in order to be executed by a probabilistic inference engine.

We are also currently developing the Meta-Knowledge brick of KREM, because in the present world, it is not rare that not only two, but multiple cultures coexist in large, multinational companies and in global projects spanning multiple geographies. The misunderstandings arising from these cultural differences are responsible for many failures, which are then blamed by each nationality on the others, without trying to understand those differences first and addressing them later.

We have developed a strategic ontology of culture, based on the works of (Meyer, 2014), who identifies eight dimensions: Communicating, Evaluating, Persuading, Leading, Deciding, Trusting, Disagreeing and Scheduling.

From the specific point of view of Knowledge Engineering, this ontology is being formally aligned with the Upper Ontology of Culture or UOC (Blanchard et al., 2010) as a framework. The cultural concepts formalized in this way in the Meta-Knowledge component of the KREM model can be used in practice using the HOT ontology. Cultural aspects in the Meta-Knowledge component will also steer the behaviour of the agents that is modelled in the Rules component of KREM.

7 CONCLUSIONS

These article proposes a modular architecture called KREM (Knowledge, Rules, Experience, Meta-Knowledge) to integrate the capitalization of previous experience in the deploying of a knowledge-based system. In fact, as in general, knowledge elicitation from experts produces incomplete expertise models, sometimes, knowledge-based systems are not able to provide pertinent results. In this case, it is necessary to call in an expert to provide an answer. It is interesting, therefore, to capitalize that answer along with the cognitive process (rules or others) that the expert carries out to develop it.

Well known technologies exist to implement the knowledge and the rules components of KREM (OWL + SWRL). But the notion of "experience" has not been fully addressed in the literature. The SOEKS (set of experience knowledge structure), initially proposed by the KERT team from the University of Newcastle (Autralia)[4] is the natural choice for formalizing the experience capitalization.

Modern knowledge engineering practices support the early criteria held in the eighties, based on a modular architecture for the development of intelligent systems, which separates domain knowledge from the reasoning mechanism. This division is thus held in the proposal of the KREM architecture, where the K module corresponds to knowledge stored in the form of logic statements, and the R module contains the necessary rules to manage and exploit such knowledge. To achieve a better performance of the knowledge-based system, we have incorporated an additional brick, which represents the experience of experts in the knowledge domain (this experience capitalization permits, to a certain extent, to have access to some tacit knowledge from the experts). Finally, the use of Meta-Knowledge allows the rational use of the other three components.

The main interest of the KREM architecture is that it is flexible and modular. Hence, each project presented in the last section was described within the architecture according to the degree of its development and maturation, showing its adaptability to cope with different Engineering application domains in a broad sense.

REFERENCES

Bazire, M. and Brézillon, P. (2005). Understanding context before using it. In *Modeling and using context*, pages 29–40. Springer.

Blanchard, E. G., Mizoguchi, R., and Lajoie, S. P. (2010). Structuring the cultural domain with an upper on-

[4]http://www.newcastle.edu.au/research-and-innovation/centre/engineering-built- environment/knowledge-engineering-research-team.

tology of culture. *The Handbook of Research on Culturally-Aware Information Technology: Perspectives and Models*, pages 179–212.

Byrne, A. and Womack, J. (2012). *The Lean Turnaround: How Business Leaders Use Lean Principles to Create Value and Transform Their Company*. BusinessPro collection. McGraw-Hill Education.

Cravero, M., de Bertrand de Beuvron, F., Zanni-Merk, C., and Marc-Zwecker, S. (2012). A description logics geographical ontology for effective semantic analysis of satellite images. In *KES*, pages 1573–1582.

Davenport, T. H. and Prusak, L. (2000). *Working Knowledge: How Organizations Manage What They Know*. Harvard Business School Press, Boston, Mass.

de Bertrand de Beuvron, F., Marc-Zwecker, S., Puissant, A., and Zanni-Merk, C. (2013). From Expert Knowledge to Formal Ontologies for Semantic Interpretation of the Urban Environment from Satellite Images. *KES Journal - Innovation in Knowledge-Based and Intelligent Engineering Systems*, 17:55 – 65.

Dey, A. K., Abowd, G. D., and Salber, D. (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Hum.-Comput. Interact.*, 16(2):97–166.

di Sciascio M., Zanni-Merk, C., C., W., S., M.-Z., and de Beuvron F. (2013). Towards a semi-automatic semantic approach to satellite images analysis. In *KES*, pages 1388–1397.

Gartiser, N., Zanni-Merk, C., Boullosa, L., and Ana (2014). A semantic layered architecture for analysis and diagnosis of sme. In Elsevier, editor, *Procedia Computer Science*, pages 1165–1174. KES 2014.

Giunchiglia, F., Maltese, V., and Dutta, B. (2012). Domains and context: first steps towards managing diversity in knowledge. *Web Semantics: Science, Services and Agents on the World Wide Web*, 12:53–63.

Golbreich, C. (2004). Combining rule and ontology reasoners for the semantic web. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 3323, pages 6–22.

Grau, B. C., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., and Sattler, U. (2008). OWL 2: The next step for OWL. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(4):309 – 322. Semantic Web Challenge 2006/2007.

Gruber, T. R. (1993). Toward Principles for the design of Ontologies used for Knowledge Sharing. *Int. J. Hum.-Comput. Stud.*, 43:907 – 928.

Hill, E. F. (2003). *Jess in Action: Java Rule-Based Systems*. Manning Publications Co., Greenwich, CT, USA.

Leistner, F. (2010). *Mastering Organizational Knowledge Flow: How to Make Knowledge Sharing Work*. Wiley.

Malhotra, Y. (2001). Expert systems for knowledge management: crossing the chasm between information processing and sense making. *Expert Systems with Applications*, 20(1):7–16.

Masai, P., Parrend, P., and Zanni-Merk, C. (2015). Towards a formal model of the lean enterprise. In Elsevier, editor, *Procedia Computer Science (Proceedings of KES 2015)*, volume 60C, pages 226–235.

McCarthy, J. (1993). Notes on formalizing context.

Meyer, E. (2014). *The Culture Map: Breaking Through the Invisible Boundaries of Global Business*. PublicAffairs.

Milton, N. (2008). *Knowledge Technologies*. Polimetrica, Milano, Italy.

Norris, J. R. (1998). *Markov Chains*. Cambridge University Press.

Ouertani, M.-Z., Baïna, S., Yesilbas, L. G., and Morel, G. (2011). Traceability and management of dispersed product knowledge during design and manufacturing. *Computer-Aided Design*, 43(5):546–562.

Porzel, R. (2010). *Contextual computing: models and applications*. Springer Science & Business Media.

Renaud, D., Zanni-Merk, C., Bouche, P., Gartiser, N., and Rousselot, F. (2010). A strategy for reasoning with close knowledge bases. *Business Informatics - Data Mining and Business Intelligence*, 16(104):25–33.

Renaud, D., Zanni-Merk, C., and Rousselot, F. (2009). A compound strategy for ontologies combining. In *KEOD*, pages 200–205.

Sanín, C. and Szczerbicki, E. (2009). Experience-based knowledge representation: Soeks. *Cybernetics and Systems*, 40(2):99–122.

Sanín, C., Szczerbicki, E., and Toro, C. (2007). An owl ontology of set of experience knowledge structure. *J. UCS*, 13(2):209–223.

Sowa, J. F. (1995). Syntax, semantics, and pragmatics of contexts. In *Conceptual Structures: Applications, Implementation and Theory*, pages 1–15. Springer.

Sowa, J. F. (2000). *Knowledge Representation: Logical, Philosophical and Computational Foundations*. Brooks/Cole Publishing Co., Pacific Grove, CA, USA.

Toro, C., Sanchez, E., Carrasco, E., Mancilla-Amaya, L., Sanín, C., Szczerbicki, E., Graña, M., Bonachela, P., Parra, C., Bueno, G., et al. (2012). Using set of experience knowledge structure to extend a rule set of clinical decision support system for alzheimer's disease diagnosis. *Cybernetics and Systems*, 43(2):81–95.

Womack, J. P., Jones, D. T., and Roos, D. (1991). *The Machine That Changed the World: The Story of Lean Production*. The MIT international motor vehicle program. HarperCollins.

Zanni-Merk, C., Almiron, S., and Renaud, D. (2011). A multi-agents system for analysis and diagnosis of smes. In *KES-AMSTA*, pages 103–112.

Zanni-Merk, C., Marc-Zwecker, S., Wemmert, C., and de Beuvron, F. d. B. (2015). A layered architecture for a fuzzy semantic approach for satellite image analysis. *International Journal of Knowledge and Systems Science (IJKSS)*, 6(2):31–56.