Ontology Engineering Methodology

York Sure[1], Steffen Staab[2], and Rudi Studer[3]

[1] SAP Research, CEC Karlsruhe, Karlsruhe, Germany, york.sure@sap.com
[2] Research Group ISWeb, University of Koblenz-Landau, Koblenz, Germany, staab@uni-koblenz.de
[3] Institute AIFB, University of Karlsruhe (TH) and FZI Research Center for Information Technology, Karlsruhe, Germany, studer@aifb.uni-karlsruhe.de

Summary. In this chapter we present a methodology for introducing and maintaining ontology based knowledge management applications into enterprises with a focus on Knowledge Processes and Knowledge Meta Processes. While the former process circles around the usage of ontologies, the latter process guides their initial set up. We illustrate our methodology by an example from a case study on skills management. The methodology serves as a scaffold for Part B "Ontology Engineering" of the handbook. It shows where more specific concerns of ontology engineering find their place and how they are related in the overall process.

1 Introduction

Ontologies constitute valuable assets that are slowly, but continuously gaining recognition and use throughout a set of disciplines – as becomes visible in Part C of this book. Ontologies frequently being a complex asset, their creation and management does neither come by coincidence nor does it come for free. Rather, the objectives pursued with their development as well as the development itself must be critically assessed by the organization or – rarely – the individual who is pushing for their creation and maintenance. The discipline that investigates the principles, methods and tools for initiating, developing and maintaining ontologies is "ontology engineering" which is the topic of this part of the handbook. "Ontology engineering methodology" as a part of ontology engineering deals with the process and methodological aspects of ontology engineering, i.e. with the issues of how to provide guidelines and advice to (potential) developers of ontologies.

It is the purpose of this chapter to introduce a rather generic ontology engineering methodology to the reader and to indicate where this methodology links to more specific topics discussed mostly, but obviously not completely, in the remainder of part B of this handbook. Such as software engineering methodologies cannot be described in isolation from actual software engineering activities, the purpose of ontology engineering methodologies can only be understood in the context of actual ontology engineering experiences.

The methodology presented here, has been derived from several case studies of building and using ontologies in the realm of knowledge management. Knowledge management deals with the thorough and systematic management of knowledge within an enterprise and between several cooperating enterprises. Knowledge management is a major issue for human resource management, enterprise organization and enterprise culture – nevertheless, information technology (IT) constitutes a crucial enabler for many aspects of knowledge management and ontologies frequently turn out to be valuable assets for knowledge management in order to target core knowledge management issues such as search, information integration, or mapping of knowledge assets. As a consequence, knowledge management is an inherently interdisciplinary subject and ontologies used for knowledge management play a central role, but at the same time they are by no means the single factor to determine success or failure of the overall system. Thus, we may derive our rationale that the objective of knowledge management constitutes a typical, yet comprehensive blueprint for issues that arise when developing complex ontologies. Therefore, we have chosen the knowledge management setting described below in order to report on a generic ontology engineering methodology.

IT-supported KM solutions are frequently built around some kind of organizational memory [1] that integrates informal, semi-formal and formal knowledge in order to facilitate its access, sharing and reuse by members of the organization(s) for solving their individual or collective tasks [7]. In such a context, knowledge has to be modelled, appropriately structured and interlinked for supporting its flexible integration and its personalized presentation to the consumer. Ontologies may provide such structuring and modeling of problems by providing a formal conceptualization of a particular domain that is shared by a group of people in an organization [14, 22].

There exist various proposals for methodologies that support the systematic introduction of KM solutions into enterprises and with it the construction of ontologies. A classical approach for introducing knowledge management systems – including ontologies – is CommonKADS that puts emphasis on an early feasibility study as well as on constructing several models that capture different kinds of knowledge needed for realizing a KM solution [26].

Re-engineering earlier approaches, we found that methodologies must distinguish two processes in order to achieve a clear identification of issues [27]: whereas the first process addresses aspects of introducing a new ontology-based system into an organization as well as maintaining it (the so-called "Knowledge Meta Process"), the second process addresses the management of knowledge using the developed ontology (or ontologies), i.e. the so-called "Knowledge Process" (see Fig. 1). E.g. in the approach described in [25], one may recognize the intermingling of the two aspects from the different roles that, e.g. "knowledge identification" and "knowledge creation" play.

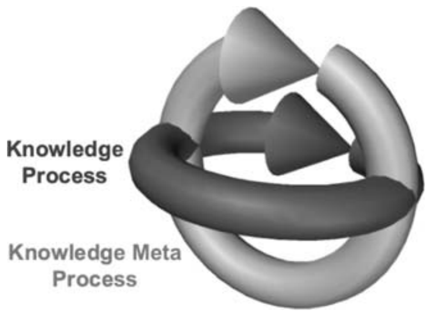


Fig. 1. Two orthogonal processes with feedback loops

The Knowledge Meta Process would certainly have its focus on knowledge identification and the Knowledge Process would rather stress knowledge creation.

The generic methodology presented here has been developed and applied in the EU project On-To-Knowledge[1] [6]. We now describe some general issues when implementing and launching ontology-based knowledge management applications. Then we focus on the knowledge meta process and the knowledge process and illustrate the instantiation of the knowledge meta process by an example from a skills management case study of the On-To-Knowledge project. During the description of the process we will point to more specific topics of ontology engineering dealt with in further chapters of this handbook.

2 Implementation and Launch of KM Applications

To implement and launch a KM application, one has to consider different processes (cf. Fig. 2). We have dealt with three major processes occurring in our case study, i.e. "Knowledge Meta Process", "Human Issues" and "Software Engineering". The processes are not completely separate but they do also overlap and interfere. As mentioned before, KM is an inherently interdisciplinary subject which should not be dominated by information technology (IT) alone, but which needs to take human and organizational issues into account. Hence, the targeted solution must trade off between problems to be solved by automated IT solutions and problems to be taken care of by human actors and through organizational processes. As a rule of thumb KM experts at a "Dagstuhl Seminar on Knowledge Management"[2] (cf. [23]) estimated that IT support cannot cover more than 10–30% of KM concerns.

Human issues (HI) and the related cultural environment of organizations heavily influence the acceptance of KM. It is often mentioned in discussions that the success of KM – and especially KM applications – strongly depends

[1] http://www.ontoknowledge.org
[2] http://dagstuhl-km-2000.aifb.uni-karlsruhe.de/

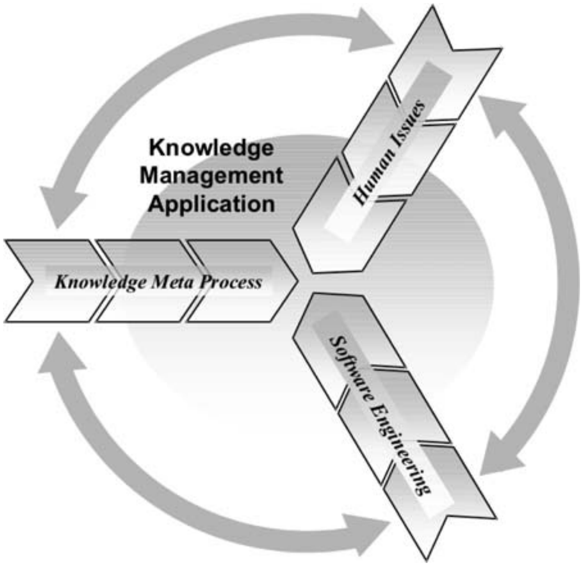


Fig. 2. Relevant processes for developing and deploying KM applications

on the acceptance by the involved people. As a consequence, "quick wins" are recommended for the initial phase of implementing any KM strategy. The aim is to quickly convince people that KM is useful for them and adds value to their daily work.

Software engineering (SE) for knowledge management applications has to accompany the other processes. The software requirements coming from the knowledge processes need to be reflected in the planning and management of the overall system design and implementation.

In the following sections we will now focus on the Knowledge Meta Process as the core process of ontology engineering and we will mention some cross-links to the other processes as well as to more specific ontology engineering issues.

3 Knowledge Meta Process

The Knowledge Meta Process (cf. Fig. 3) consists of five main steps. Each step has numerous sub-steps, requires a main decision to be taken at the end and results in a specific outcome. The main stream indicates steps (phases) that finally lead to an ontology-based KM application. The phases are "Feasibility Study", "Kickoff", "Refinement", "Evaluation" and "Application and Evolution". Below every box depicting a phase the most important sub-steps are listed, e.g. "Refinement" consists of the sub-steps "Refine semi-formal ontology description", "Formalize into target ontology" and "Create prototype", etc. Each document-flag above a phase indicates major outcomes of the step, e.g. "Kickoff" results in an "Ontology Requirements Specification Document

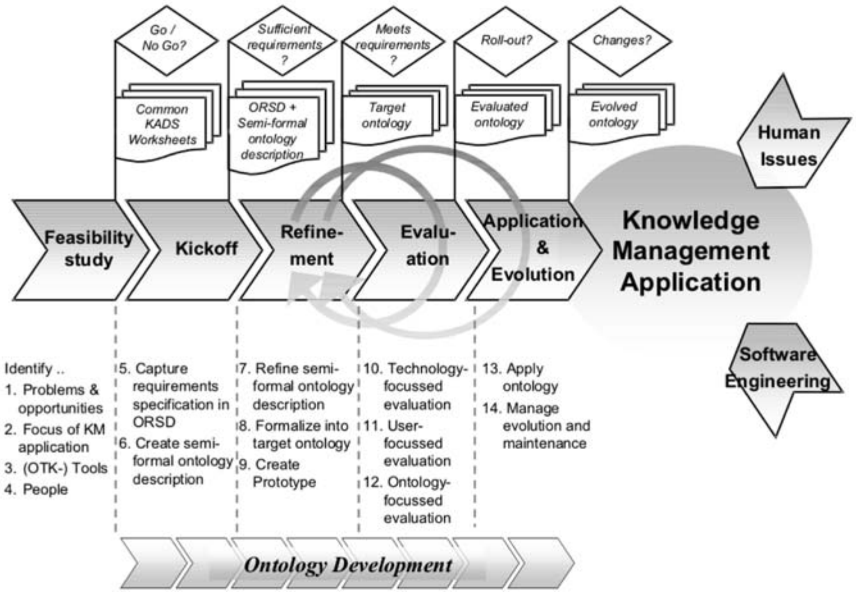

Go / No Go?
Sufficient requirements ?
Meets requirements ?
Roll-out?
Changes?
Common KADS Worksheets
ORSD + Semi-formal ontology description
Target ontology
Evaluated ontology
Evolved ontology
Human Issues
Feasibility study
Kickoff
Refine-ment
Evalu-ation
Application & Evolution
Knowledge Management Application
Software Engineering
Identify ..
1. Problems & opportunities
2. Focus of KM application
3. (OTK-) Tools
4. People
5. Capture requirements specification in ORSD
6. Create semi-formal ontology description
7. Refine semi-formal ontology description
8. Formalize into target ontology
9. Create Prototype
10. Technology-focussed evaluation
11. User-focussed evaluation
12. Ontology-focussed evaluation
13. Apply ontology
14. Manage evolution and maintenance
Ontology Development

Fig. 3. The knowledge meta process

(ORSD)" and the "Semi-formal ontology description", etc. Each node above a flag represents the major decisions that have to be taken at the end to proceed to the next phase, e.g. whether in the Kickoff phase one has captured sufficient requirements. The major outcomes typically serve as decision support for the decisions to be taken. The phases "Refinement–Evaluation–Application and Evolution" typically need to be performed in iterative cycles. One might notice that the development of such an application is also driven by other processes, e.g. software engineering and human issues. We will only briefly mention some human issues in the example section.

3.1 Feasibility Study

Any knowledge management system may function properly only if it is seamlessly integrated in the organization in which it is operational. Many factors other than technology determine success or failure of such a system. To analyze these factors, we initially start with a *feasibility study* [26], e.g. to identify problem/opportunity areas and potential solutions. In general, a feasibility study serves as a decision support for economical, technical and project feasibility, determining the most promising focus area and target solution.

Considering ontology engineering specifically, there is a need to consider the return on investment of developing ontologies as an asset. So far, the accounting of ontology as value assets has not been undertaken to our knowledge. Methods of accounting other intangible assets, such as [8] which builds on the approach of Balanced Scorecard [18], seem to be applicable, but need to be investigated more specifically. Experiences of investment needs for ontology development have been collected and are now available for broader use. They are reported in chapter "Exploring the Economical Aspects of Ontology Engineering."

3.2 Kickoff

In the kickoff phase the actual development of the ontology begins. Similar to requirements engineering and as proposed by [11] we start with an *ontology requirements specification document* (ORSD). The ORSD describes what an ontology should support, sketching the planned area of the ontology application and listing, e.g. valuable knowledge sources for the gathering of the semi-formal ontology description. The ORSD should guide an ontology engineer to decide about inclusion and exclusion of concepts and relations and the hierarchical structure of the ontology. In this early stage one should look for already developed and potentially reusable ontologies (cf. [31] on reuse).

Valuable knowledge sources may include text documents or available relational data. The knowledge contained in such data sources, and particularly in text, may be unlocked by means of ontology learning methods (cf. chapter "Ontology and the Lexicon"). A specific techniques, which is sometimes used for ontology learning, is the analysis of concept properties allowing for the derivation of hierarchical relationships by means of formal concept analysis (cf. chapter "Formal Concept Analysis").

The *outcome* of this phase is (beside the ontology requirement specification document (ORSD)) a semi-formal description of the ontology, i.e. a graph of named nodes and (un-)named, (un-)directed edges, both of which may be linked with further descriptive text, e.g. in form of mind maps (cf. [4, 32]). If the requirements are sufficiently captured, one may proceed with the next phase. The *decision* is typically taken by ontology engineers in collaboration with domain experts. "Sufficiently" in this context means, that from the current perspective there is no need to proceed with capturing or analyzing knowledge. However, it might be the case that in later stages gaps are recognized. Therefore, the ontology development process is cyclic.

3.3 Refinement

During the kick-off and refinement phase one might distinguish in general two concurrent approaches for modeling, in particular for refining the semi-formal ontology description by considering relevant knowledge sources: top–down and bottom–up. In a *top–down*-approach for modeling the domain one starts by modeling concepts and relationships on a very generic level. Subsequently these items are refined. This approach is typically done manually and leads to a high-quality engineered ontology. Available top-level ontologies (cf. chapter

"Foundational Choices in DOLCE") may here be reused and serve as a starting point to develop new ontologies. In our example scenario we encountered a *middle-out* approach, i.e. to identify the most important concepts which will then be used to obtain the remainder of the hierarchy by generalization and specialization. However, with the support of an automatic document analysis (cf. chapter "Ontology and the Lexicon"), a typical *bottom–up*-approach may be applied. There, relevant concepts are extracted semi-automatically from available documents. Based on the assumption that most concepts and conceptual structures of the domain as well the company terminology are described in documents, applying knowledge acquisition from text for ontology design helps building ontologies automatically.

To *formalize* the initial semi-formal description of the ontology into the target ontology, ontology engineers firstly form a taxonomy out of the semi-formal description of the ontology and add relations other than the "is-a" relation which forms the taxonomical structure. The ontology engineer adds different types of relations as analyzed, e.g. in the competency questions to the taxonomic hierarchy. However, this step is cyclic in itself, meaning that the ontology engineer now may start to interview domain experts again and use the already formalized ontology as a base for discussions. It might be helpful to visualize the taxonomic hierarchy and give the domain experts the task to add attributes to concepts and to draw relations between concepts (e.g. we presented them the taxonomy in form of a mind map as mentioned in the previous section). The ontology engineer should extensively document the additions and remarks to make ontological commitments made during the design explicit. The application of design patterns for ontologies (cf. chapter "Ontology Design Patterns") may greatly improve the efficiency and effectiveness of the process as well as the quality of the ontology.

The *outcome* of this phase is the "target ontology", that needs to be evaluated in the next step. The major *decision* that needs to be taken to finalize this step is whether the target ontology fulfills the requirements captured in the previous kickoff phase. Typically an ontology engineer compares the initial requirements with the current status of the ontology. This decision will typically be based on the personal experience of ontology engineers. As a good rule of thumb we discovered that the first ontology should provide enough "flesh" to build a prototypical application. This application should be able to serve as a first prototype system for evaluation.

3.4 Evaluation

We distinguish between three different types of evaluation: (1) technology-focussed evaluation, (2) user-focussed evaluation and (3) ontology-focused evaluation.

Our evaluation framework for *technology-focussed evaluation* consists of two main aspects: (1) the evaluation of properties of ontologies generated by development tools, (2) the evaluation of the technology properties, i.e. tools and applications which includes the evaluation of the evaluation tool properties themselves. In an overview these aspects are structured as follows: (1) Ontology properties (e.g. language conformity (Syntax), consistency (Semantics)) and (2) technology properties (e.g. interoperability, turn around ability, scalability etc.).

The framework shown above concentrates on the technical aspects of ontologies and related ontologies. However, the aspect of *user-focussed evaluation* remains open. The most important point from our perspective is to evaluate whether users are satisfied by the KM application. More specific, whether an ontology based application is at least as good as already existing applications that solve similar tasks.

Beside the above mentioned process oriented and pragmatic evaluation methods, one also need to *formally evaluate ontologies*. One of the most prominent approaches here is the OntoClean approach (cf. chapter "An Overview of OntoClean"), which is based on philosophical notions. Another well-known approach (cf. chapter "Ontology Engineering Environments") takes into account the normalization of an ontology. Applying such approaches helps avoiding common modelling errors and leads to more correct ontologies.

The *outcome* of this phase is an evaluated ontology, ready for the roll-out into a productive system. However, based on our own experiences we expect in most cases several iterations of "Evaluation–Refinement–Evaluation" until the outcome supports the decision to roll-out the application. The major *decision* that needs to be taken for finalizing this phase is whether the evaluated ontology fulfills all evaluation criteria relevant for the envisaged application of the ontology.

3.5 Application and Evolution

The *application* of ontologies in productive systems, or, more specifically, the usage of ontology based systems, is being described in the following Sect. 4 that illustrates the knowledge process.

The *evolution* of ontologies is primarily an organizational process. There have to be rules to the update, insert and delete processes of ontologies (cf. [29]). We recommend, that ontology engineers gather changes to the ontology and initiate the switch-over to a new version of the ontology after thoroughly testing all possible effects to the application. Most important is therefore to clarify *who* is responsible for maintenance and *how* it is performed and in *which time intervals* is the ontology maintained. However, there also exist technical approaches for the consistent evolution of ontologies (cf. [16,17,30]).

A current topic for research and practice is the use of evolutionary knowledge management technologies that frequently build on Web2.0 technology and that decentralize the responsibility of knowledge management processes and meta processes to the individuals in the (virtual) organization with a corresponding need to decentralize ontology engineering (cf. [3, 28] on decentralized and evolutionary knowledge management and chapter "Ontology

Engineering and Evolution in a Distributed World Using DILIGENT" on decentralized, evolutionary ontology engineering).

The *outcome* of an evolution cycle is an evolved ontology, i.e. typically another version of it. The major *decision* to be taken is when to initiate another evolution cycle for the ontology.

4 Knowledge Process

Once a KM application is fully implemented in an organization, knowledge processes essentially circle around the following steps (cf. Fig. 4):

- *Knowledge creation* and/or *import* of documents and meta data, i.e. contents need to be created or converted such that they fit the conventions of the company, e.g. to the knowledge management infrastructure of the organization.
- then knowledge items have to be *captured* in order to elucidate importance or interlinkage, e.g. the linkage to conventionalized vocabulary of the company by the creation of relational metadata.
- *retrieval of* and *access to knowledge* satisfies the "simple" requests for knowledge by the knowledge worker;
- typically, however, the knowledge worker will not only recall knowledge items, but she will process it for further *use* in her context.

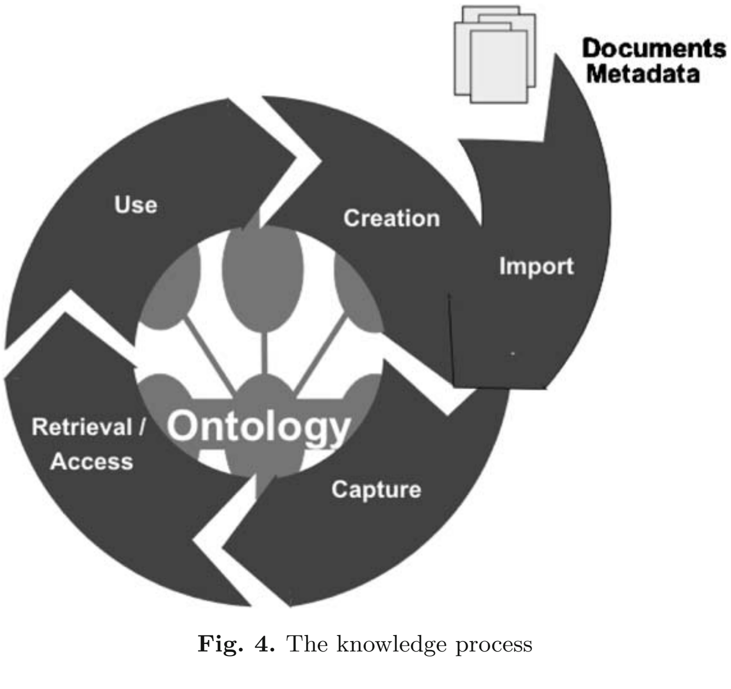
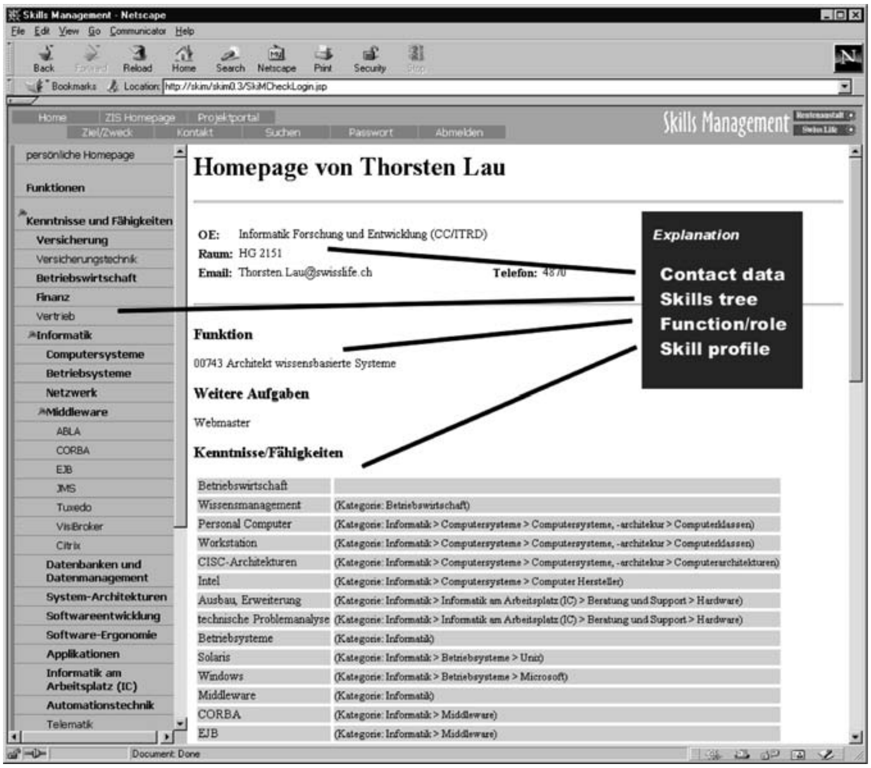


Fig. 4. The knowledge process

5 Example: Skills Management at Swiss Life

We now give an example of the Knowledge Meta Process instantiation of a skills management case study at Swiss Life (cf. [19]). Skills management makes skills of employees explicit. Within the case study existing skill databases and documents (like, e.g. personal homepages) are integrated and expanded. Two aspects are covered by the case study: first, explicit skills allow for an advanced expert search within the intranet. Second, one might explore his/her future career path by matching current skill profiles vs. job profiles. To ensure that all integrated knowledge sources are used in the same way, ontologies are used as a common mean of interchange to face two major challenges. Firstly, being an international company located in Switzerland, Swiss Life has internally four official languages, viz. German, English, French and Italian. Secondly, there exist several spellings of same concepts, e.g. "WinWord" vs. "MS Word". To tackle these problems, ontologies offer external representations for different languages and allow for representation of synonymity. Figure 5 shows a screenshot from the skills management application. The prototype enables any employee to integrate personal data from numerous distributed and heterogeneous sources into a single coherent personal homepage.



Homepage von Thorsten Lau

OE: Informatik Forschung und Entwicklung (CC/ITRD)
Raum: HG 2151
Email: Thorsten.Lau@swisslife.ch **Telefon:** 4870

Funktion

00743 Architekt wissensbasierte Systeme

Weitere Aufgaben

Webmaster

Kenntnisse/Fähigkeiten

Betriebswirtschaft	
Wissensmanagement	(Kategorie: Betriebswirtschaft)
Personal Computer	(Kategorie: Informatik > Computersysteme > Computersysteme, -architektur > Computerklassen)
Workstation	(Kategorie: Informatik > Computersysteme > Computersysteme, -architektur > Computerklassen)
CISC-Architekturen	(Kategorie: Informatik > Computersysteme > Computersysteme, -architektur > Computerarchitekturen)
Intel	(Kategorie: Informatik > Computersysteme > Computer Hersteller)
Ausbau, Erweiterung	(Kategorie: Informatik > Informatik am Arbeitsplatz (IC) > Beratung und Support > Hardware)
technische Problemanalyse	(Kategorie: Informatik > Informatik am Arbeitsplatz (IC) > Beratung und Support > Hardware)
Betriebsysteme	(Kategorie: Informatik)
Solaris	(Kategorie: Informatik > Betriebsysteme > Unix)
Windows	(Kategorie: Informatik > Betriebsysteme > Microsoft)
Middleware	(Kategorie: Informatik)
CORBA	(Kategorie: Informatik > Middleware)
EJB	(Kategorie: Informatik > Middleware)

Fig. 5. Skills management case study at Swiss life

5.1 Feasibility Study

For identifying factors which can be central for the success or failure of the ontology development and usage we made a requirement analysis of the existing skills management environment and evaluated the needs for a new skills management system. We identified mainly the human resources department and the management level of all other departments as actors and stakeholders for the skills management. After finding the actors and stakeholders in the skills management area, we named the ontology experts for each department, which are preferably from the associated training group of each department.

5.2 Kickoff

The departments private insurance, human resources and IT constitute three different domains that were the starting point for an initial prototype. The task was to develop a skills ontology for the departments containing three trees, viz. for each department one. The three trees should be combined under one root with cross-links in between. The root node is the abstract concept "skills" (which means in German "Kenntnisse/Faehigkeiten") and is the starting point to navigate through the skills tree from the top.

During the *kickoff* phase two workshops with three domain experts[3] were held. The first one introduced the domain experts to the ideas of ontologies. Additional potential knowledge sources were identified by the domain experts, that were exhaustively used for the development of the ontologies, e.g. a book of the Swiss Association of Data Processing ("Schweizerischer Verband fuer Datenverarbeitung") describing professions in the computing area in a systematic way similar to an ontology. Obviously, this was an excellent basis to manually build the skills ontology for the IT domain. First experiments with extracting an ontology semi-automatically by using information extraction tools did not satisfy the needs for a clearly structured and easily understandable model of the skills. The domain experts and potential users felt very uncomfortable with the extracted structures and rather chose to build the ontology by themselves "manually". To develop the first versions of the ontologies, we used a mind mapping tool ("MindManager"). It is typically used for brainstorming sessions and provides simple facilities for modelling hierarchies very quickly. The early modelling stages for ontologies contain elements from such brainstorming sessions (e.g. the gathering of the semi-formal ontology description).

During this stage a lot of "concept islands" were developed, which were isolated sets of related terms. These islands are subdomains of the corresponding domain and are self-contained parts like "operating systems" as sub domain in the IT domain. After developing these concept islands it was necessary to

[3] Thanks to Urs Gisler, Valentin Schoeb and Patrick Shann from Swiss Life for their efforts during the ontology modelling.

combine them into a single tree. This was a more difficult part than assembling the islands, because the islands were interlaced and for some islands it was possible to add them to more than one other island, which implies awkward skills trees that contain inconsistencies after merging. For each department one skills tree was built in separate workshops. A problem that came up very early was the question where to draw the line between concepts and instances. E.g. is the programming language Java instantiated by "jdk1.3" or is "jdk1.3" so generic that it still belongs to the concept-hierarchy? Another problem was the size of the ontology. What is the best depth and width of each skills tree? Our solution was, that it depends on the domain and should be determined by the domain expert.

As *result* of the kick-off phase we obtained the semi-formal ontology descriptions for the three skills trees, which were ready to be formalized and integrated into a single skills ontology. At this stage the skills trees reached a maturity that the combination of them caused no major changes for the single skills trees.

5.3 Refinement

During the *refinement* phase we formalized and integrated the semi-formal ontology descriptions into a single coherent skills ontology. An important aspect during the formalization was (1) to give the skills proper names that uniquely identify each skill and (2) to decide on the hierarchical structure of the skills. We discussed two different approaches for the hierarchical ordering: we discovered that categorization of skills is typically not based on an `is-a`-taxonomy, but on a much weaker hasSubtopic relationship that has implications for the inheritance of attached relations and attributes. However, for our first prototype this distinction made no difference due to missing cross-taxonomical relationships. But, according to [15], subsumption provided by `is-a` taxonomies is often misused and a later formal evaluation of the skills ontology according to the proposed OntoClean methodology possibly would have resulted in a change of the ontology.

In a second refinement cycle we added one more relation type, an "associative relation" between concepts. They express relations outside the hierarchic skills tree, e.g. a relation between "HTML" and "JSP", which occur not in the same tree, but correspond with each other, because they are based on the same content. "HTML" is in the tree "mark-up languages", while the tree "scripting languages" contains "JSP". This is based on the basic characteristics and the history of both concepts, which changed over time. But in reality they have a close relationship, which can be expressed with the associative relation.

The other task in this phase was to integrate the three skills ontologies into one skills ontology and eliminate inconsistencies in the domain ontology parts and between them. Because the domain ontologies were developed separately, the merger of them caused some overlaps, which had to be resolved. This happened for example in the computer science part of the skills trees, where the departments IT and private insurance have the same concepts like “Trofit” (which is a Swiss Life specific application). Both departments use this concept, but each uses a different view. The IT from the development and the private insurance from the users view. Additionally the personal skills of any employee are graded according to a generic scale of four levels: basic knowledge, practical experience, competency, and top specialist. The employees will grade their own skills themselves. As known from personal contacts to other companies (e.g. Credit Suisse, ABB and IBM), such an approach proved to produce highly reliable information.

As a *result* at the end of the refinement phase the “target skills ontology” consisted of about 700 concepts, which could be used by the employees to express their skill profile.

5.4 Application and Evolution

The *evaluation* of the prototype and the underlying ontology was unfortunately skipped due to internal restructuring at Swiss Life which led to a closing down of the whole case study.

Still, we considered the following aspects for the *evolution* of our skills management application: The competencies needed from employees are a moving target. Therefore the ontologies need to be constantly evaluated and maintained by experts from the human resource department. New skills might be suggested by the experts themselves, but mainly by employees. Suggestions include both, the new skill itself as well as the position in the skills tree where it should be placed. While employees are suggesting only new skills, the experts decide which skills should change in name and/or position in the skills tree and, additionally, decide which skill will be deleted. This was seen as necessary to keep the ontology consistent and to avoid that, e.g. similar if not the same concept appear even in the same branch. For each ontology (and domain) there should exist a designated ontology manager who decides if and how the suggested skill is integrated.

6 Related Work on Methodologies

A first overview on methodologies for ontology engineering can be found in [9]. Within OntoWeb[4] there have been joint efforts of members, who produced an extensive state-of-the-art overview of methodologies for ontology engineering (cf. [10, 13]). There exist also deliverables on guidelines and best practices for industry (cf. [20, 21]) with a focus on applications for E-Commerce, Information Retrieval, Portals and Web Communities.

[4] OntoWeb, a European thematic network, see http://www.ontoweb.org for further information.

CommonKADS [26] is not *per se* a methodology for ontology development. It covers aspects from corporate knowledge management, through knowledge analysis and engineering, to the design and implementation of knowledge-intensive information systems. CommonKADS has a focus on the initial phases for developing knowledge management applications, we therefore relied on CommonKADS for the early feasibility stage. E.g. a number of worksheets is proposed that guide through the process of finding potential users and scenarios for successful implementation of knowledge management.

The *Enterprise Ontology* [37] [38] proposed three main steps to engineer ontologies: (1) to identify the purpose, (2) to capture the concepts and relationships between these concepts, and the terms used to refer to these concepts and relationships, and (3) to codify the ontology. In fact, the principles behind this methodology influenced many approaches in the ontology community. These principles are also reflected and appropriately extended in the steps kickoff and refinement of our methodology.

TOVE [36] proposes a formalized method for building ontologies based on competency questions. We found the approach of using competency questions, that describe the questions that an ontology should be able to answer, very helpful and integrated it in our methodology.

METHONTOLOGY [11, 12] is a methodology for building ontologies either from scratch, reusing other ontologies as they are, or by a process of re-engineering them. The framework enables the construction of ontologies at the "knowledge level". The framework consists of: identification of the ontology development process where the main activities are identified (evaluation, configuration, management, conceptualization, integration implementation, *etc.*); a lifecycle based on evolving prototypes; and the methodology itself, which specifies the steps to be taken to perform each activity, the techniques used, the products to be output and how they are to be evaluated. METHONTOLOGY is partially supported by WebODE. Our combination of the On-To-Knowledge Methodology and OntoEdit (cf. [32, 33]) is quite similar to the combinations of METHONTOLOGY and WebODE (cf. [2]. In fact, they are the only duet that has reached a comparable level of integration of tool and methodology.

More recently, the DILIGENT methodology has been developed that addresses the decentralized engineering of ontologies [24]. The development of DILIGENT is driven by the fact that in a lot of application scenarios a geographically dispersed group of ontology engineers, domain experts, and ontology users that are often distributed across different organizations, has to develop and maintain a shared ontology for knowledge management. DILIGENT puts special emphasis on supporting the argumentation process that is needed in agreeing on updates of a shared ontology [35]. Obviously, these techniques would be a valuable support for the refinement and evolution phases of our methodology. A detailed description of DILIGENT is given in chapter "Ontology Engineering and Evolution in a Distributed World Using DILIGENT."

Currently, the NeOn methodology for engineering networked ontologies is under development as part of the NeOn [5] project [31]. This methodology supports among others the reuse of ontologies as well as of non-ontological resources as part of the engineering process. The NeOn methodology also provides detailed guidelines for executing its various activities. This includes the usage of ontology design patterns as described in chapter "Ontology Design Patterns." Thus, the NeOn methodology would provide additional methods for the kickoff and refinement phases of our methodology.

7 Conclusion

The described methodology was developed and applied in the On-To-Knowledge project and influenced work, e.g. in the SEKT and the NEON projects. One of the core contributions of the methodology that could not be shown here is the linkage of available tool support with case studies by showing when and how to use tools during the process of developing and running ontology based applications in the case studies (cf. [34]).

Lessons learned during setting up and employing the methodology in the On-To-Knowledge case studies include: (1) different processes drive KM projects, but "Human Issues" might dominate other ones (as already outlined by Davenport [5]), (2) guidelines for domain experts in industrial contexts have to be pragmatic, (3) collaborative ontology engineering requires physical presence *and* advanced tool support and (4) brainstorming is very helpful for early stages of ontology engineering, especially for domain experts not familiar with modelling (more details on be found, e.g. in [32, 33]).

In this chapter we have shown a process oriented methodology for introducing and maintaining ontology based knowledge management systems. Core to the methodology are Knowledge Processes and Knowledge Meta Processes. While Knowledge Meta Processes support the setting up of an ontology based application, Knowledge Processes support its usage. Still, there are many open issues to solve, e.g. how to handle a distributed process of emerging and aligned ontologies that is likely to be the scenario in the semantic web.

References

1. A. Abecker, A. Bernardi, K. Hinkelmann, O. Kuehn, and M. Sintek. Toward a technology for organizational memories. *IEEE Intelligent Systems*, 13(3):40–48, 1998.
2. J. C. Arpírez, O. Corcho, M. Fernández-López, and A. Gómez-Pérez. WebODE: A scalable workbench for ontological engineering. In *Proceedings of the First International Conference on Knowledge Capture (K-CAP) Oct. 21–23, 2001, Victoria, BC, Canada*, 2001.

[5] EU Integrated Project NeOn Lifecycle Support for Networked Ontologies, see www.neon-project.org for further information.

3. M. Bonifacio, T. Franz, and S. Staab. *A Four-Layer Model for Information Technology Support of Knowledge Management*, chapter 6. Advances in Management Information Systems. M.E. Sharpe, Armonk, NY, 2008.
4. T. Buzan. *Use your head.* BBC Books, 1974.
5. T. H. Davenport and L. Prusak. *Working Knowledge – How organisations manage what they know.* Havard Business School Press, Boston, MA, 1998.
6. J. Davies, D. Fensel, and F. van Harmelen, editors. *On-To-Knowledge: Semantic Web enabled Knowledge Management.* Wiley, New York, 2002.
7. R. Dieng, O. Corby, A. Giboin, and M. Ribiere. Methods and tools for corporate knowledge management. *International Journal of Human-Computer Studies*, 51(3):567–598, 1999.
8. A. M. Fairchild. Knowledge management metrics via a balanced scorecard methodology. In *HICSS. Proceedings of the 35th Annual Hawaii International Conference on System Sciences, 2002*, pages 3173–3180, 2002.
9. M. Fernández-López. Overview of methodologies for building ontologies. In *Proceedings of the IJCAI-99 Workshop on Ontologies and Problem-Solving Methods: Lessons Learned and Future Trends.* CEUR, Aachen, Germany, 1999.
10. M. Fernandéz-López, A. Gómez-Pérez, J. Euzenat, A. Gangemi, Y. Kalfoglou, D. M. Pisanelli, M. Schorlemmer, G. Steve, L. Stojanovic, G. Stumme, and Y. Sure. A survey on methodologies for developing, maintaining, integrating, evaluating and reengineering ontologies. OntoWeb deliverable 1.4, Universidad Politecnia de Madrid, 2002.
11. M. Fernández-López, A. Gómez-Pérez, J. P. Sierra, and A. P. Sierra. Building a chemical ontology using Methontology and the Ontology Design Environment. *Intelligent Systems*, 14(1), 1999.
12. A. Gómez-Pérez. A framework to verify knowledge sharing technology. *Expert Systems with Application*, 11(4):519–529, 1996.
13. A. Gómez-Pérez, M. Fernandéz-López, O. Corcho, T. T. Ahn, N. Aussenac-Gilles, S. Bernardos, V. Christophides, O. Corby, P. Crowther, Y. Ding, R. Engels, M. Esteban, F. Gandon, Y. Kalfoglou, G. Karvounarakis, M. Lama, A. López, A. Lozano, A. Magkanaraki, D. Manzano, E. Motta, N. Noy, D. Plexousakis, J. A. Ramos, and Y. Sure. Technical roadmap. OntoWeb deliverable 1.1.2, Universidad Politecnia de Madrid, 2002.
14. T. R. Gruber. Towards principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43(5/6):907–928, 1995.
15. N. Guarino and C. Welty. Evaluating ontological decisions with OntoClean. *Communications of the ACM*, 45(2):61–65, 2002.
16. Peter Haase, Frank van Harmelen, Zhisheng Huang, Heiner Stuckenschmidt, and York Sure. A framework for handling inconsistency in changing ontologies. In Y. Gil, E. Motta, V. R. Benjamins, and M. A. Musen, editors, *Proceedings of the Fourth International Semantic Web Conference (ISWC2005)*, volume 3729 of *LNCS*, pages 353–367. Springer, Berlin, 2005.
17. P. Haase, J. Vlker, and Y. Sure. Management of dynamic knowledge. *Journal of Knowledge Management*, 9(5):97–107, 2005.
18. R. S. Kaplan and D. P. Norton. The balanced scorecard – Measures that drive performance. *Harvard Business Review*, 71ff, January–February 1992.
19. T. Lau and Y. Sure. Introducing ontology-based skills management at a large insurance company. In *Proceedings of the Modellierung 2002*, pages 123–134, Tutzing, Germany, March 2002.

20. A. Léger, H. Akkermans, M. Brown, J.-M. Bouladoux, R. Dieng, Y. Ding, A. Gómez-Pérez, S. Handschuh, A. Hegarty, A. Persidis, R. Studer, Y. Sure, V. Tamma, and B. Trousse. Successful scenarios for ontology-based applications. OntoWeb deliverable 2.1, France Télécom R&D, 2002.
21. A. Léger, Y. Bouillon, M. Bryan, R. Dieng, Y. Ding, M. Fernandéz-López, A. Gómez-Pérez, P. Ecoublet, A. Persidis, and Y. Sure. Best practices and guidelines. OntoWeb deliverable 2.2, France Télécom R&D, 2002.
22. D. O'Leary. Using AI in knowledge management: Knowledge bases and ontologies. *IEEE Intelligent Systems*, 13(3):34–39, 1998.
23. D. O'Leary and R. Studer. Knowledge management: An interdisciplinary approach. *IEEE Intelligent Systems, Special Issue on Knowledge Management*, 16(1), 2001.
24. H. S. Pinto, S. Staab, and C. Tempich. Diligent: Towards a fine-grained methodology for distributed, loosely-controlled and evolving engineering of ontologies. In *ECAI*, pages 393–397, 2004.
25. G. Probst, K. Romhardt, and S. Raub. *Managing Knowledge.* Wiley, New York, 1999.
26. G. Schreiber, H. Akkermans, A. Anjewierden, R. de Hoog, N. Shadbolt, W. van de Velde, and B. Wielinga. *Knowledge Engineering and Management – The CommonKADS Methodology.* MIT, Cambridge, MA, 1999.
27. S. Staab, H.-P. Schnurr, R. Studer, and Y. Sure. Knowledge processes and ontologies. *IEEE Intelligent Systems, Special Issue on Knowledge Management*, 16(1):26–34, 2001.
28. S. Staab and H. Stuckenschmidt, editors. *Semantic Web and Peer-to-Peer.* Springer, Berlin, 2006.
29. L. Stojanovic, A. Maedche, B. Motik, and N. Stojanovic. User-driven ontology evolution management. In A. Gómez-Pérez and V. R. Benjamins, editors, *Proc. of EKAW-2002*, volume 2473 of *LNCS*, pages 285–300, Siguenza, Spain. Springer, Berlin, 2002.
30. L. Stojanovic. *Methods and Tools for Ontology Evolution.* Ph.D. thesis, Universität Karlsruhe (TH), Universität Karlsruhe (TH), Institut AIFB, D-76128 Karlsruhe, 2004.
31. M. C. Suárez-Figueroa, G. Aguado de Cea, C. Buil, K. Dellschaft, M. Fernández-López, A. García, A. Gómez-Pérez, G. Herrero, E. Montiel-Ponsoda, M. Sabou, B. Villazon-Terrazas, and Z. Yufei. Neon methodology for building contextualized ontology networks. Technical report, NeOn Deliverable D5.4.1, February 2008.
32. Y. Sure, M. Erdmann, J. Angele, S. Staab, R. Studer, and D. Wenke. OntoEdit: Collaborative ontology development for the semantic web. In I. Horrocks and J. A. Hendler, editors, *Proc. of International Semantic Web Conference 2002*, volume 2342 of *Lecture Notes in Computer Science (LNCS)*, pages 221–235, Sardinia, Italy, 2002. Springer, Berlin, 2002.
33. Y. Sure, S. Staab, and J. Angele. OntoEdit: Guiding ontology development by methodology and inferencing, volume 2519 of *Lecture Notes in Computer Science (LNCS)*, pages 1205–1222, University of California, Irvine, USA, 2002. Springer, Berlin, 2002.
34. Y. Sure and R. Studer. On-To-Knowledge Methodology – Final version. On-To-Knowledge deliverable 18, Institute AIFB, University of Karlsruhe, 2002.
35. C. Tempich, E. Simperl, M. Luczak, H. S. Pinto, and R. Studer. Argumentation-based ontology engineering. *IEEE Intelligent Systems*, 22(6):52–59, 2007.

36. M. Uschold and M. Grueninger. Ontologies: Principles, methods and applications. *Knowledge Sharing and Review*, 11(2), 1996.
37. M. Uschold and M. King. Towards a methodology for building ontologies. In *Workshop on Basic Ontological Issues in Knowledge Sharing, held in conjunction with IJCAI-95*, Montreal, Canada, 1995.
38. M. Uschold, M. King, S. Moralee, and Y. Zorgios. The enterprise ontology. *Knowledge Engineering Review*, 13(1):31–89, 1998.