EU-IST Integrated Project (IP) IST-2003-506826 SEKT

SEKT: Semantically Enabled Knowledge Technologies

D7.1.1 SEKT Methodology: Survey and Initial Framework

York Sure
Christoph Tempich
Denny Vrandečić

with contributions from Sofia Pinto
(Instituto Superior Tecnico,
Universidade Tecnica de Lisboa, Portugal)

Abstract.
EU-IST Integrated Project (IP) IST-2003-506826 SEKT
Deliverable D7.1.1 (WP 7.1)
This deliverable provides the first set of guidelines for ontology engineering and application in SEKT: DILIGENT – DIstributed , Loosely-controlled and evolvInG Engineering of oNTologies.

Keyword list:
Ontology Engineering
Methodology
Rhetorical Structure Theory (RST)

Document Id.	SEKT/2004/D7.1.1/v1.0
Project	SEKT EU-IST-2003-506826
Date	December 29, 2004
Distribution	public

SEKT Consortium

This document is part of a research project partially funded by the IST Programme of the Commission of the European Communities as project number IST-2003-506826.

British Telecommunications plc.
Orion 5/12, Adastral Park
Ipswich IP5 3RE
UK
Tel: +44 1473 609583, Fax: +44 1473 609832
Contact person: John Davies
E-mail: john.nj.davies@bt.com

Jozef Stefan Institute
Jamova 39
1000 Ljubljana
Slovenia
Tel: +386 1 4773 778, Fax: +386 1 4251 038
Contact person: Marko Grobelnik
E-mail: marko.grobelnik@ijs.si

University of Sheffield
Department of Computer Science
Regent Court, 211 Portobello St.
Sheffield S1 4DP
UK
Tel: +44 114 222 1891, Fax: +44 114 222 1810
Contact person: Hamish Cunningham
E-mail: hamish@dcs.shef.ac.uk

Intelligent Software Components S.A.
Pedro de Valdivia, 10
28006 Madrid
Spain
Tel: +34 913 349 797, Fax: +49 34 913 349 799
Contact person: Richard Benjamins
E-mail: rbenjamins@isoco.com

Ontoprise GmbH
Amalienbadstr. 36
76227 Karlsruhe
Germany
Tel: +49 721 50980912, Fax: +49 721 50980911
Contact person: Hans-Peter Schnurr
E-mail: schnurr@ontoprise.de

Vrije Universiteit Amsterdam (VUA)
Department of Computer Sciences
De Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
Tel: +31 20 444 7731, Fax: +31 84 221 4294
Contact person: Frank van Harmelen
E-mail: frank.van.harmelen@cs.vu.nl

Empolis GmbH
Europaallee 10
67657 Kaiserslautern
Germany
Tel: +49 631 303 5540, Fax: +49 631 303 5507
Contact person: Ralph Traphöner
E-mail: ralph.traphoener@empolis.com

University of Karlsruhe, Institute AIFB
Englerstr. 28
D-76128 Karlsruhe
Germany
Tel: +49 721 608 6592, Fax: +49 721 608 6580
Contact person: York Sure
E-mail: sure@aifb.uni-karlsruhe.de

University of Innsbruck
Institute of Computer Science
Techikerstraße 13
6020 Innsbruck
Austria
Tel: +43 512 507 6475, Fax: +43 512 507 9872
Contact person: Jos de Bruijn
E-mail: jos.de-bruijn@deri.ie

Kea-pro GmbH
Tal
6464 Springen
Switzerland
Tel: +41 41 879 00, Fax: 41 41 879 00 13
Contact person: Tom Bösser
E-mail: tb@keapro.net

Sirma AI EAD, Ontotext Lab
135 Tsarigradsko Shose
Sofia 1784
Bulgaria
Tel: +359 2 9768 303, Fax: +359 2 9768 311
Contact person: Atanas Kiryakov
E-mail: naso@sirma.bg

Universitat Autonoma de Barcelona
Edifici B, Campus de la UAB
08193 Bellaterra (Cerdanyola del Vallès)
Barcelona
Spain
Tel: +34 93 581 22 35, Fax: +34 93 581 29 88
Contact person: Pompeu Casanovas Romeu
E-mail: pompeu.casanovas@uab.es

Executive Summary

This deliverable provides the first set of guidelines for ontology engineering and application in SEKT: DILIGENT – DIstributed, Loosely-controlled and evolvInG Engineering of oNTologies. It consists several main building blocks: (i) a survey of relevant existing approaches, (ii) the DILIGENT process model and argumentation framework and (iii) first results from applying DILIGENT in the SEKT case studies. The DILIGENT process has already been evaluated in an in-situ experiment at the Institute AIFB, the argumentation framework is based on a thorough ex-post analysis in the biology domain.

Contents

Chapter 1

Introduction

1.1 Motivation

The SEKT project combines the topics **knowledge management** and **ontologies**. The driving force for combing the two topics is the growing importance of knowledge for societies and their economies. Both topics are now briefly introduced to motivate the contributions of this work.

Our society changed from being an industrial society to being a knowledge society. This shift of paradigms enforced enterprises to act no longer based on purely tayloristic principles but rather as *"intelligent enterprises"* (*cf.* [Qui92]). **Knowledge** became the key economic resource, as Drucker pointed out:

> *"The basic economic resource – the means of production – is no longer capital, nor natural resources, nor labor. It is and will be knowledge."*
> [Dru93]

This so-called *"post-industrial revolution"* (*cf.* [Jac96]) focused the view on knowledge as *"intellectual capital"* (*cf.* [Ste97]) that is a mission critical resource. Therefore, companies should have the same interest in managing their knowledge as in managing capital investment and working relationships (*cf.* [EM97]).

Knowledge management (KM) was born as significant corporate strategy to meet the new challenges. The history and the current status of KM is sketched by Kay:

> *"Knowledge management as an approach to business management has had a tumultuous history. It was born as a hip buzzword, was shunned as a second cousin to business process reengineering, and was for a time hijacked by software vendors. Despite this circuitous path, knowledge management is*

> *now well on the way to becoming a necessary component of every bottom-line-oriented company's long-term business strategy."*
> [Kay03]

The main goal of typical current KM initiatives is to enable a better knowledge sharing. Drivers for the introduction of knowledge management were *e.g.* the potential for reduction of (i) costs for duplication of efforts, (ii) loss of knowledge when key people leave a company and (iii) time needed to find correct answers. This has led to many efforts for capturing, storing and making knowledge accessible. But, as Davenport and Prusak mention, sharing knowledge requires a common language:

> *"People can't share knowledge if they don't speak a common language."*
> [DP98]

Successful KM strategies consist of building blocks for organization, people, technology and corporate culture (*cf.* [Alb93, Sch96a]. In such a context, knowledge sharing is not only a matter of communication between people, but also between people and technology and between technology, *e.g.*, software agents that communicate with people or between each other. More generally speaking, agents (human and software agents) need to share their knowledge and require a common language. Thus, we generalize the quotation from above to *"Agents can't share knowledge if they don't speak a common language"*.

Ontologies were exploited in Computer Science to enhance knowledge sharing and reuse (*cf.* [Gru93, Fen01]). Firstly, they provide a shared and common understanding of knowledge in a domain of interest. Secondly, they capture and formalize knowledge by connecting human understanding of symbols with their machine processability. As such, ontologies act as a common language between agents. The use of ontologies for knowledge management offers therefore great advantages. Numerous applications already exist (*cf.* [SS03]).

Common knowledge management applications make use of available technology that was originally developed for the World Wide Web, *e.g.* the now very popular corporate intranets. Similarly to the Web they provide access to a large amount of information contained in documents, databases *etc.* and suffer from the same weaknesses, *e.g.*,

(i) **searching for information** often leads to irrelevant information,

(ii) **extracting information** is left to humans since software agents are not yet equipped with common sense and domain knowledge to extract such information from textual representations and they fail to integrate information distributed over different sources,

(iii) **maintaining** weakly structured text sources is a time-consuming and difficult task when such sources become large (*cf.* introduction of [DFv02]).

To overcome such weaknesses of the current Web, Berners-Lee and others envisioned the **Semantic Web**:

> *"The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in co-operation."*
> [BLHL01]

The Semantic Web extends the Web by adding machine-processable meta-information, aka metadata, to documents. The metadata explicitly define what the document is about. Thereby, ontologies provide the schema for metadata to make them reusable and define their meaning.

1.2 The SEKT Big Picture

This report is part of the work performed in workpackage (WP) 7 on "Methodology". As shown in Figure 1.1 this work is closely related with "Usability and Benchmarking"; they both are intermediating between the research and development WPs and the case study WPs in SEKT.

The main goal of this activity is to provide a methodology for the application of Semantic Knowledge Technologies into enterprises in a goal-driven and process-oriented way. Hence, we need an integrated approach balancing the organisation and management aspects on the one hand, and the information technology and systems aspects on the other hand.

Central to Semantic Knowledge Technologies is the application of ontologies to knowledge management problems. Core aspects for the methodology therefore include the efficient and effective creation and maintenance of ontologies in settings such as provided by the case studies.

1.3 The SEKT Methodology - a fine grained View

SEKT seeks to integrate advantages of the different technologies mentioned in the Big Picture 1.1 for the overall goal to provide better technological support for knowledge management.

The work described in WP 7 is concerned with the methodological aspects of integrating the different technologies and thereby overcome weaknesses of currently available methodologies.

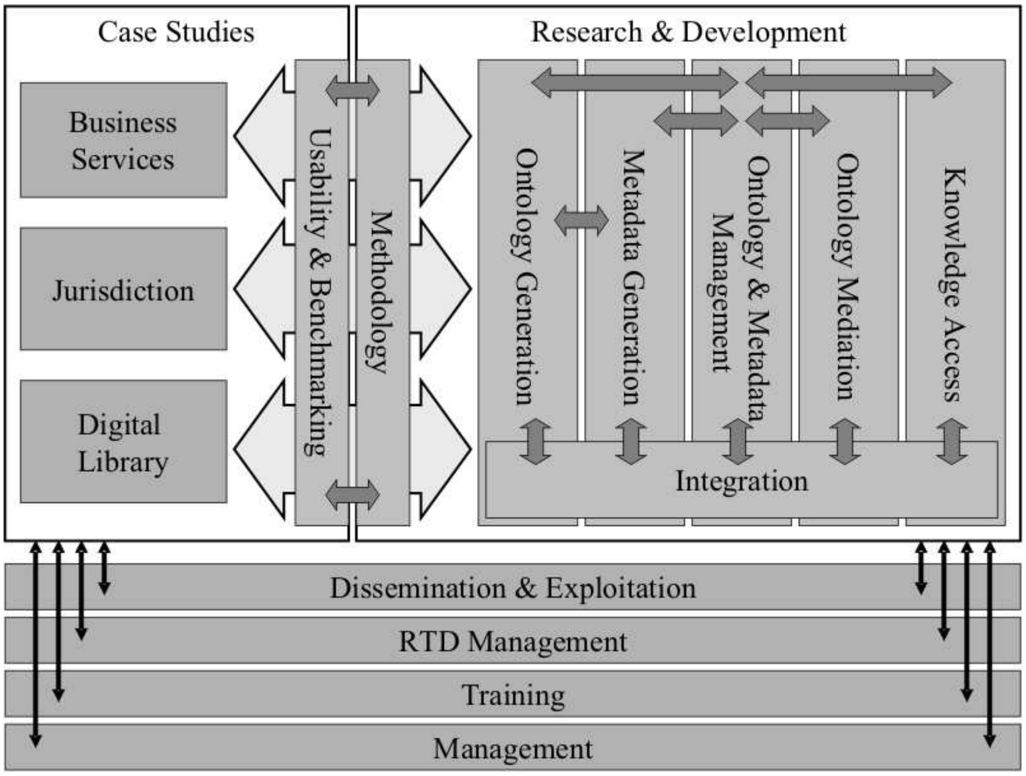


Figure 1.1: The SEKT Big Picture

1. Classical development of knowledge-based systems and of corresponding ontologies is mostly *centralized* like the targeted knowledge-based system itself. In contrast, we here consider the general tendency to support *distributed information processing with ontologies*, *e.g.* the Semantic Web, agents, web services or ontology-based peer-to-peer. Stakeholders in an ontology development process will hardly ever gather in one place. Yet they have an interest to contribute fruitfully toward the ongoing development of their ontologies:

2. Existing methodologies support knowledge engineering (KE) by using check lists that guide the engineering process. The check lists have been shaped by the needs of *knowledge engineers* to cope comprehensively with nearly arbitrarily complex processes. In contrast, in the distributed cases we consider, the participation of a knowledge engineer is often restricted to a, possibly complex, core ontology. Beyond the core, these cases involve extensive participation and, comparatively simple, concept formation by *domain experts*.

3. KE has mostly focused on an *up- and running system* with some moderate effort for maintenance. In contrast, ontologies for distributed information processing must permanently *evolve* in order to reflect the widely diverging needs of their users.

4. KE methodologies remain rather *coarse* and the gap between their description and concrete actions to be taken is filled by the KE. In contrast, for Semantic Web ontologies and comparable use cases, we ask the question whether we could provide the domain experts with *fine-grained* guidance in order to improve their effectiveness and efficiency in ontology engineering.

To account for some of the differences between classical knowledge engineering and ontology engineering methodologies derived from there, we have started to develop a methodology for DIstributed (cf. item 1 above), Loosely-controlled (cf. item 2) and evolvInG (cf. item 3) Engineering of oNTologies, the validity of which has been partially checked and is still being checked against experiences in two case studies (cf. [PSST04].

The methodology will combine aspects from different areas to integrate the different technologies in a concise way. We argue that this requires a major abstraction step which leads us to a methodology applicable in distributed environment, which can only be loosely controlled and evolves constantly. At this abstraction level we could recognize similarities between our objectives and objectives being worked on in the field of argumentation visualization. Without going into detail we here list the two areas which we will subsequently analyse.

- Methodologies for ontology engineering
- Computer supported argumentation

In the reminding report we will refer in each chapter to these areas. The overview chapter will explain the different parts and relate them to the overall objective of the deliverable.

1.4 Organization of the Deliverable

This deliverable is organized as follows. We firstly present our results of a review of related work in the area in Chapter 2. We have in particular analysed the current state of the art of ontology engineering tools and tool for argumentation visualization. On the research side we have identified the major methodologies for ontology engineering. Furthermore we have summarized the relevant research areas for argumentation visualization and conflict resolution with a focus on ontology engineering. From this review we could derive several open questions with respect to ontology engineering methodologies. Readers mainly interested in the methodology itself can skip this chapter.

In Chapter 3 we motivate and present the DILIGENT process and argumentation framework. We show a first evaluation of the process in in-situ experiments at the Institute AIFB. Finally, we elaborate on our initial studies for the argumentation framework, i.e. a thorough ex-post analysis of an evolving taxonomy in the biology domain from which we derived our argumentation framework based on the Rhetorical Structure Theory (RST).

Before concluding we illustrate in Chapter 4 how we initially applied DILIGENT in the SEKT case studies and provide ideas for further application and adaptation.

Chapter 2

Survey

2.1 Introduction

This chapter provides a survey of related work. It is organized as follows. First, we describe the focus of our methodology, provide a definition for methodology and explain how this relates to ontologies in Section 2.2. We concentrate on two major areas which have impact on our approach, viz. arguments in Section 2.3 and ontology learning in Section 2.4. Then we provide a more detailed view on existing tools (Section 2.5) and past and current research (Section 2.6).

2.2 The Methodology Focus

It has been a widespread conviction in knowledge engineering that methodologies for building knowledge-based systems help knowledge engineering projects to successfully reach their goals in time (cf. [SAA$^+$99] for one of the most widely deployed methodologies). With the arrival of ontologies in knowledge-based systems the same kind of methodological achievement for structuring the ontology-engineering process has been pursued by approaches like [GPFLC03, SSSS01, UK95] and their application has been proposed in such areas as the Semantic Web, too. At this point, however, we have found some mismatches between these proposals (including our own) and the requirements we meet in the Semantic Web.

In the next sections we will describe different areas which we currently regard as related to our work. Before that, we will now explain, why these areas are important. Therefore we recall the objective of the SEKT methodology: to provide support for the ontology engineer in the task of creating, maintaining and instantiating an ontology with the help of the SEKT technologies. Hence, it is immediately clear that existing work in the area of ontology engineering methodologies is relevant in our context. The SEKT technologies are mainly concerned with automation of the ontology creation task, *i.e.* “ontology learning". The processes which have been defined to support the ontology learning task must therefore be integrated into a common methodology. From a wider perspective an ontology learning method can be seen as an agent which proposes changes to a given ontology or creates a new one. The ontology engineer must then decide which changes are integrated in the ontology at hand. This is very similar to a setting in which many people collaboratively build an ontology for a given domain. In a collaborative environment participants propose changes, exchange arguments and finally agree on a certain approach. The capturing of arguments is analysed as part of the area of "Visualizing Argumentations". We can use the experiences made in that field to integrate ontology learning methods with the work of a human ontology engineer. Additionally we can make the ontology engineering decisions more transparent. To make the methodology more general we not only consider synchronous collaboration but also asynchronous and distributed collaboration. Our methodology should not require that the ontology is designed in one location. We explicitly want to support distributed engineering of ontologies. With this abstraction we can consequently handle the engineering of ontologies in distributed settings given in peer-to-peer systems, but also the automated engineering of ontologies using different Web services or users working with a centralised system such as Livelink where each users has its own view (extension) to an ontology.

One of the main features of automated methods is that they can be applied repeatedly with very low additional costs. By the incorporation of automated methods into the ontology engineering task, we can apply those methods continuously to available data. This data will change and so will the ontology. Evolution and change of the ontology must thus be explicitly dealt with in our methodological framework.

Compared to existing ontology design methods, introducing collaboration, distribution and evolution significantly reduces the amount of precision and control available in the process of producing the ontology. Analogously to people designing an ontology, different automated methods will propose different extensions to an ontology. An initially shared ontology may develop in different departments of a company or in various areas of the web in various directions just as organisms have found many ways to adapt to our environment through evolutions. However, from an ontological point of view it is desirable to share the conceptualization. The methodology must organize the different processes in a way that the participants can find the highest common denominator.

2.2.1 Definition of Methodology for Ontologies

> **methodology**[1] – An organised, documented set of procedures and guidelines for one or more phases of the software life cycle, such as analysis or design. Many methodologies include a diagramming notation for documenting the results of the procedure; a step-by-step "cookbook" approach for carrying out the procedure; and an objective (ideally quantified) set of criteria for

---

[1] see `http://computing-dictionary.thefreedictionary.com/Methodology`

> determining whether the results of the procedure are of acceptable quality.

Following this citation a methodology should contain the four items:

- Procedures
- Documentation standards
- Guidelines (cookbook)
- Evaluation criteria and analysis techniques

The definition of a methodology is now applied to the ontology engineering setting. In the following we separate the different parts which constitute a methodology and present the objectives in each step for the ontology engineering problem. We will subsequently analyse for the related work which part the according methodology supports, and are able to draw a picture in which the missing points emerge.

2.2.2 Set of Procedures

In the ontology building, procedures for the following three activities must be defined:

- Ontology management activities
- Ontology development oriented activities
- Ontology support activities

Ontology management activities: Procedures for ontology management activities must include definitions for the scheduling of the ontology engineering task. Further it is necessary to define control mechanism and quality assurance steps.

Ontology development oriented activities: When it comes to the development of the ontology it is important that procedures are defined for enrolling environment and feasibility studies. After the a decision to build an ontology the ontology engineer needs procedures to specify, conceptualize, formalize and implement the ontology. Finally the users and engineers need guidance for the maintenance/population, use and evolution of the ontology.

Ontology support activities: To aid the development of an ontology, a number of important supporting activities should be undertaken. Supporting activities include knowledge acquisition, evaluation, integration, merging and alignment and configuration management. These activities are performed in all steps of the development and management process. Knowledge acquisition can happen in a centralized as well as a decentralized way. Ontology learning is a way to support the manual knowledge acquisition with machine learning techniques.

2.2.3 Documenting the Results

It is important to document the results after each activity. In a later stage of the development process this helps to trace why certain modelling decisions have been undertaken. The documentation of the results can be facilitated with an appropriate tool support. Depending on the methodology the documentation level can be quite different. One methodology might require to document only the results of the ontology engineering process while others give the decision process itself quite some importance.

2.2.4 Step-by-step "Cookbook"

Each of the analysed methodologies provides some sort of step by step "cookbook". However, they differ in the requirement on the ontology engineer. It is desirable that even a "rookie" ontology engineer could refine and extend an ontology after studying the current status. However, in most cases the methodologies are not fine grained enough to enable untrained persons to engineer an ontology from scratch. For each of the activities a step-by-step "cookbook" should define the input data and output data and the exact procedures how to transform input into the desired output.

2.2.5 Set of Criteria for Evaluation

In the ontology engineering setting evaluation measures should provide means to measure the quality of the created ontology. This is particular difficult for ontologies, since modelling decisions are in most cases subjective. A general survey of evaluation measures for ontologies can be found in [GP04]. Additionally we want to refer to the evaluation measures which can be derived from statistical data (*cf.* [TV03]) and measures which are derived from philosophical principles. One of the existing approaches for ontology evaluation is OntoClean [GW02] which *e.g.* has been implemented as part of OntoEdit [SAS03]. This approach is based on philosophical principles, so far only few examples of its application to practical use cases are known. Further research on ontology evaluation is currently being carried out as part of the EU IST thematic network Knowledge Web[2].

---

[2] see `http://knowledgeweb.semanticweb.org/`

2.3 Arguments

2.3.1 Visualizing Argumentation

We can deploy research in the area of argumentation visualization [CSSS01, KSE03] to solve certain requirements on the envisioned SEKT methodology. Argumentation visualization helps its users to discuss their problems in a clearly defined way. The methodologies proposed in this field base their recommendations on different models of agreement processes. Following the methodologies various tools have been implemented. The main advantage of computer supported argumentation is the achieved traceability of decisions. In particular the traceability problem is well researched in the software engineering community (*cf.* [PB88]).

> **Traceability** – "A software requirements specification is traceable if (i) the origin of each of its requirements is clear and if (ii) it facilitates the referencing of each requirement in future development or enhancement documentation" (ANSI/IEEE Standard 830-1984) [IEE84].

The original response to argumentation visualization was the application of the IBIS methodology [KR70]. IBIS allows users to capture different design deliberations. Appropriate tools (see Section 2.5) can help to retrieve them in a sophisticated manner later on. However, the IBIS technique has received criticism due to its resilience to change and being too abstract.

In our context different actors collaborate to design an ontology and find commonalities. They will exchange arguments in favor or against certain modelling decision. We believe that by selecting the right model for argumentation we can on the one hand enhance the traceability of modelling decisions and on the other hand guide the engineering in a fine grained way towards a final shared ontology. However, simply applying for example the IBIS methodology will probably not be sufficient [PB88]. According to his findings IBIS should be enhanced with domain specific knowledge. More recently [GF97] has found that further enhancement of IBIS can be achieve by introducing an acceptance and rejection mechanism.

2.3.2 Argumentation Theory

Besides an intuitive and correct way to present the exchanged arguments to the user, the underlying argumentation theory is also a main concern. There are a number of argumentation theories ranging from informal explanations of argumentation threads to very formal specifications. To exemplify the notion of argument we introduce here the oldest model of natural argumentation. This provides an idea of the general concepts uses in this area. In the subsequent chapters we will present current theories and tools which have been developed bearing in mind these theories.

The Toulmin model [TRJ84], a natural argumentation theory that tries to explain how real people (not philosophers) argue, has its main components: **Data** (facts, data and information, the reason for the claim), **Claim** (the position on the issue, the conclusion being advocated) and **Warrant** (logical connection between the data and the claim, the reasoning process used to arrive at the claim,[3]). Other components are: **Backing** (material supporting the warrant), **Reservation** (exceptions to the claim) and **Qualifiers** (relative strength).

The following examples are taken from the ChangingMinds website[4].

Claim A claim is a statement one persons asks another one to accept. This includes information they should accept as true or actions they should accept and enact.

For example:

> You should use a hearing aid.

Many people start with a claim, but then find that it is challenged. To convince another person one must prove the claim. This is where grounds become important.

Grounds The grounds (or data) is the basis of real persuasion and is made up of data and hard facts. It is the truth on which the claim is based. The actual truth of the data may be less than 100%, as all data is based on perception and, hence, has some element of assumption about it.

It is critical to the argument that the grounds are not challenged, because if they are, they may become a claim, which must be proven with even deeper information and further argument.

For example:

> Over 70% of all people over 65 years have a hearing difficulty.

Data Data is usually a very powerful element of persuasion, although it does affect people differently. Those who are dogmatic, logical or rational will more likely to be persuaded by data. Those who argue emotionally and who are highly invested in their own position will challenge it or otherwise try to ignore it. It is often a useful test to give something factual to the other person that disproves their argument and watch how they handle it. Some will accept it without question. Some will dismiss it out of hand. Others will dig deeper, requiring more explanation. This is where the warrant comes into its own.

Warrant A warrant links data to a claim, legitimizing the claim by showing the data to be relevant. The warrant may be explicit or unspoken and implicit. It answers the question 'Why does that data mean your claim is true?'

---

[3] Authoritative, motivational, and substantive (which includes cause-effect, effect-cause, generalization based on example, classification, etc.).

[4] see `http://changingminds.org/disciplines/argument/toulmin.htm`

For example:

A hearing aid helps most people to hear better.

The warrant may be simple and it may also be a longer argument with additional sub-elements, including those described below.

Backing The backing (or support) to an argument gives additional support to the warrant by answering different questions.

For example:

Hearing aids are available locally.

Qualifier The qualifier (or modal qualifier) indicates the strength of the leap from the data to the warrant and may limit how universally the claim applies. They include words such as 'most', 'usually', 'always', 'sometimes'. Arguments may thus range from strong assertions to fuzzy statements.

For example:

Hearing aids help most people.

Reservation Another variant is the reservation, which may give the possibility of the claim being incorrect.

For example:

Unless there is evidence to the contrary, hearing aids do no harm to ears.

Qualifiers and reservations are much used by advertisers who are constrained not to lie. Thus they slip 'usually', 'virtually', 'unless' and so on into their claims.

Rebuttal Despite the careful construction of the argument, there may still be counter-arguments that can be used. These may be rebutted either through a continued dialogue, or by pre-empting the counter-argument by giving the rebuttal during the initial presentation of the argument.

For example:

There is a support desk that deals with technical problems.

Any rebuttal is an argument in itself, and thus may include a claim, warrant, backing and so on. It also, of course can have a rebuttal. Thus if you are presenting an argument, you can seek both possible rebuttals and also rebuttals to the rebuttals.

2.4 Ontology Learning Processes

Ontology learning aims at the integration of a multitude of disciplines in order to facilitate the construction of ontologies, in particular ontology engineering and machine learning. Because the fully automatic acquisition of knowledge from machines remains in the distant future, the overall process is considered to be semi-automatic, i.e. with human intervention. It relies on a coordinated interaction between human modeler and learning algorithm for the construction of ontologies.

In [Mae02] a generic ontology learning architecture is presented. The process model there builds on the principal idea of data mining as a process (*e.g.* [CKC$^+$99]) with the phases of business and data understanding, data preparation, modelling, evaluation and deployment.

Ontology learning is being recognized as an important topic and numerous people are moving to focus on the topic[5]. However, no holistic approach which is similar to or an extension of the previously mentioned one is known so far. We will continue to survey this rapidly emerging field to take into account the newest research results.

2.5 Existing Tools

2.5.1 Tools for Visualization of Arguments

Within the SEKT project argumentation visualization is not a primary research focus. We rather want to use the mature ideas from that field to enhance the ontology engineering process. Therefore we omit a complete analysis of available tools and just refer the interested reader to [GF94]. There over 100 commercial tools and research products were reviewed. However, we here summarize the main findings from there analysis. The issue raised provides a fruitful input for our own research.

A number of problems in the field of traceability are identified. Surprisingly, the inability to locate and access the sources of requirements is the most commonly cited problem across all the practitioners in there investigations. This problem was also reported to cause many others:

- An out of date RS (Requirements specification), as an RS evolves poorly when those originally responsible are not involved in its evolution, or where it is impossible to regain the original context.
- Slow realization (and deterioration as a result) of change, as the most time-consuming and erroneous part is often the identification of those to involve and

---

[5] see e.g. the workshops on (i) Mining for and from the Semantic Web (MSW) at `http://km.aifb.uni-karlsruhe.de/ws/msw2004` and (ii) Ontology Learning and Population (OLP) at `http://olp.dfki.de/ecai04/cfp.htm` for very first approaches

inform.

- Unproductive conflict resolution, decision making, and negotiation, as most tools supporting these activities do not help to identify or locate the essential participants.
- Poor collaboration, as the invisibility of changing work structures and responsibilities makes it difficult to: transfer information amongst parties; integrate work; and assign work to those with relevant knowledge and experience.
- Difficulty in dealing with the consequences when individuals leave a project and with the integration of new individuals.
- Poor reuse of requirements, as reuse is mainly successful when those initially responsible for their production are either directly involved or readily accessible.

Selected Tools for Argument Visualization

As the number of available tools in this area is huge we pick out only one commercial tool based on the most famous system to capture deliberations. From the research tools we choose the ones most cited in the research community.

QuestMap QuestMap is an award-winning product for mediating meetings through Visual Information Mapping. QuestMap originates from the pioneering hypertext system building by Jeff Conklin in the mid-1980s, whose team at MCC developed gIBIS[CB88] for capturing software design rationale, and then went onto build QuestMap.

Compendium Compendium [SSS$^+$01] builds on the gIBIS methodology. It is a semantic hypertext tool to capture arguments and visualize them. It offers a conceptual framework of argumentation, it promotes the use of a meeting facilitator and there exist a number of tools to present the exchanged arguments to different audiences. Compendium tools include *Question based templates* to facilitate the flow of the arguments. Hence, the discussion can be lead by “pre-formulated” questions which structure the discussion. The process of the discussion is visualized by different maps, interlinking and connecting the exchanged arguments. In Compendium any kind of idea can be expressed since its notation is very flexible.

ClaiMaker ClaiMaker [KSE03, LUM$^+$02] focuses on scientific debate, where scientists can express the positions and contributions in a publication through a combination of free text and structuring constructs.

Tellis The Tellis tool [BG04, GR02] is a system for structured argumentation on any topic where users progress from information sources to arguments that intermix free text and structured connectors.

2.5.2 Ontology Engineering Tools

An early overview of tools that support ontology engineering can be found in [DSW$^+$00]. However, there have been joint efforts of members of the thematic network OntoWeb[6], who provided an extensive state-of-the-art overview on ontology related tools, including Ontology Engineering Environments (OEE, *cf.* [GPAFL$^+$02]). A sign for the growing interest in Ontologies and tools that support ontology engineering is the (recently updated) published comparison of ontology editors on XML.com (*cf.* [Den02, Den04]). An evaluation of ontology engineering environments has been performed as part of the EON 2002 workshop (*cf.* [SA02]). With respect to our work in SEKT, especially the following tools are noteworthy.

APECKS [TS98] is targeted mainly for use by domain experts, possibly in the absence of a knowledge engineer, and its aim is to foster and support debate about domain ontologies. It does not enforce consistency nor correctness, and instead allows different conceptualisations of a domain to coexist.

Chimaera [MFRW00] is primarily a merging tool for ontologies. It contains only a simple editing environment and relies on the Ontolingua Server for more advanced modelling.

The **DOGMAModeler** is a set of tools for ontology engineering. It relies on ORM (*cf.* [Hal01]) as graphical notion and its cross-bonding ORM-ML to ensure easy exchange (*cf.* [DJM02]). It supports the database-inspired DOGMA ontology engineering approach and is coupled with the DOGMA Server as a backend.

KAON OImodeller [MMV02, BEH$^+$02] belongs to the KAON tool suite. The system is designed to be highly scalable and relies on an advanced conceptual modelling approach that balances some typical trade-offs to enable a more easily integration into existing enterprise information infrastructure. The extension of the KAON tool suit is part of the SEKT project.

OilEd [BHGS01]is a graphical ontology editor that initially was dedicated to modelling of DAML+OIL (now OWL) ontologies. Thus, on the one hand it is dependent on a particular representation language, but on the other hand offers strong support for modelling such ontologies. A key aspect of OilEd is the use for FaCT [Hor98] to classify ontologies and check consistency via translation from OWL to the SHIQ description logic. However, the tool is not extensible *e.g.* by plugins, nor does is provide sophisticated support for collaboration aspects.

---

[6] see `http://www.ontoweb.org/`

The **Ontolingua** [FFR96] Server is a set of tools and services that support the building of shared ontologies between distributed groups. It provides access to a library of ontologies and translators to languages such as Prolog, CLIPS and Loom. The set of tools was one of the first sophisticated ontology engineering environments with a special focus on the collaboration aspects. However, the development has not kept pace with the evolving current standards such as RDF or OWL, nor with the state-of-the-art technology.

Ontosaurus [SPKR96] consists of two modules: an ontology server, which uses Loom as knowledge representation system, and an ontology 'browser server' that dynamically creates HTML pages to display the ontology hierarchy. Translators exist from Loom to Ontolingua, KIF, KRSS and C++. Similar to the Ontolingua Server, it was a milestone in the development of OEEs, but the development has not kept pace with the evolving standards and technologies.

Protégé [NFM00] is a well established ontology editor with a large user community. The design of the tool is very similar to OntoEdit since it actually was the first editor with an extensible plugin structure and it also relies on the frame paradigm for modelling. Numerous plugins from external developers exist. It also supports current standards like RDF(S) and OWL. Recently also support for axioms was added through the "PAL tab" (Protégé axiom language, *cf.* [HNM02]).

WebODE [ACFLGP01] is an "ontology engineering workbench" that provides various service for ontology engineering. Similar to OntoEdit, it is accompanied by a sophisticated methodology of ontology engineering, *viz.* METHONTOLOGY (*cf.* [GP96, FLGPSS99]). In contrast to OntoEdit and Protégé it (both Java standalone applications) is purely web-based and is built on top of an application server. At the same time this gives WebODE an equal level of extensibility. For inferencing services it relies on Prolog. It provides translators to current standards such as RDF(S) and OWL.

WebOnto [Dom98] and the accompanying tool **Tadzebao** support graphical ontology engineering and in particular the argument between users on the ontology design, using text, GIF images and even hand drawn sketches. The strength of this approach lies in the advanced support for communication between ontology engineers and domain experts. However, the tool is not extensible nor does it provide sophisticated and specialized inferencing support.

OntoEdit [SAS03] Although OntoEdit has its roots in the research community, it has been launched into the commercial marketplace. OntoEdit supports explicitly the OTK methodology [SS02]. The open plug-in framework enables the integration of a number of extension to the basic ontology management services OntoEdit provides. In particular OntoEdit offers advanced support for collaboration and a integration of the inferencing capabilities. Noteworthy is the plug-in which implements the recommendations of the OntoClean methodology [GW02].

HCOME [KVA04] HCOME is probably the most recent development in the field of ontology management tools in the research community. The tool supports a scenario which is very similar to the objectives of the SEKT methodology. The tool supports the usual ontology management functions such as versioning and editing. However, it is not clear if they support inferencing. Besides those rather traditional functions, HCOME supports the discussion of ontological decisions with support of the IBIS methodology [KR70]. The ontology engineers may work distributively on different ontology, and are supported in collaboratively engineering a shared ontology. Their methodology does not include support for automated methods to ontology engineering, neither exists an elaborated process to reach agreement towards a shared ontology.

2.5.3 Conclusions

There exists a plethora of ontology engineering tools. Major critique points from our point of view are the following ones.

- The tools typically target manual ontology creation without integration of automatic approaches for ontology learning.
- Only very few tools provide support for distributed engineering of ontologies.
- None of the tools supports structured argumentation during the creation of ontologies.
- Apart from KAON no tool supports properly the evolution of ontologies.

2.6 Past and current Research

2.6.1 Visualization of Argumentation

As we could already see while analysing the available tools for argumentation visualization, the number of them is huge. Similarly the number of argumentation models and related research is very diverse. We attempt nevertheless to structure the area and provide references to the research we will deploy for our methodology.

Therefore we distinguish several research areas which contribute to the field in a whole. An important aspect of argumentation is the mode – **synchronous, asynchronous** – in which it is performed. Different **models** have been developed to conceptualize the way argumentation is done. Furthermore the **conceptualization of arguments** themselves is subject to investigation. In an argumentation **conflicts** can arise, thus models of conflict exist and proposals how to resolve them systematically. We recall that our objective is to deploy the findings from the selected areas to enhance ontology engineering. Hence, we finally analyse the work done in the **intersection** of argumentation and ontology engineering.

Synchronous and Asynchronous Argument Exchange

We start with separation when and how the argumentation takes place. One can distinguish synchronous and asynchronous interaction. Synchronous interaction implies that the parties discuss the *Claims* at the same time (not necessarily at the same place). Asynchronous interaction refers to discussions where *Claims* can be brought forward at various points in time. The most obvious example are discussion per email. Asynchronous interaction is typically more difficult to support than synchronous.

[SK93] analysis which kind of arguments are exchanged in a discussion depending whether it is performed asynchronous or synchronous. Therefore they distinguish arguments related to the content, meeting management and project management. Their main findings are that issues stated in one mode are continued in that mode and that the mode was chosen according to the urgency of the required decision. Alternatives to content related issues are presented five times as often than issue themselves.

The Compendium tool [SSS$^+$01] offers support for both modes. However, they assume that discussions take place in synchronous mode with the help of the facilitator. Subsequent discussions can than be performed partly asynchronously. The support comes mainly through visualizing the synchronous discussion in various ways.

Argumentation Model

The Toulmin model of argumentation was the first one presented in the literature. Today IBIS is the most often used model to describe argumentation. When it comes to the analysis of texts the Rhetorical Structure Theory is most often used. In the SEKT project a tool will developed by BT to automatically identify the underlying rhetorical structures of a text.

- **Information based information systems (IBIS):** IBIS (pronounced “eye-bis”) stands for Issue-Based Information System, and was developed by Horst Rittel and colleagues during the early 1970’s [KR70]. IBIS was developed to provide a simple yet formal structure for the discussion and exploration of “wicked” problems. Problems that are wicked, as opposed to tame, do not yield to the traditional “scientific” approach to problem solving, which is to gather data, analyse the data, formulate a solution and implement the solution. With a wicked problem your understanding of the problem is evolving as you work on a solution. One sure sign of a wicked problem is that there is no clear agreement about what the “real problem” is. Wicked problems cannot be solved in the traditional sense, because one runs out of resources (time, money, energy, people, etc.) before a perfect solution can be implemented.

  gIBIS [CB88] focuses on capturing collaborative deliberations about design in the form of graphs containing text at their nodes. It is the first graphical interface for the IBIS method.

In IBIS the following terms are used to classify different arguments.

Question / Issue States a question, raises an Issue

Idea proposes a possible resolution for the question

Argument states an opinion or judgement that either supports or objects to one or more ideas

response to Indicates a response to a question

supports Supports an argument

objects to Objects to an argument

Specializes Defines a question with more detail

Challenges Challenges an argument, an idea or a question

Justification Justifies an argument, an idea or a question

Expands-on Adds new information to an idea

Decision nodes Indicate that a decision was reached on a certain issue

- **Rhetorical Structure Theory (RST):** The aim of Rhetorical Structure Theory (RST) [MT87] is to offer an explanation of the coherence of texts. It is assumed that for every part of a coherent text there is some function. RST focuses on showing an evident role for every part of a text. A text is usually divided into structures, building blocks. These blocks are of 2 levels: nuclearity and relations. The most frequent structure is two spans of text (virtually adjacent). These are usually related such that one of them has a specific role relative to the other: the span making the claim is the nucleus (N) and the span with the evidence is the satellite (S). Thirty relations between 2 spans of text have already been identified and loosely defined. [Mar97] presents an algorithm, which is able to extract the relations from natural language text with high precision.

  For the sake of completeness we here list some of the most important relations found in RST: Elaboration, Evaluation, Justification, Contrast, Alternative, Example, Counter Example, Background knowledge, Motivation, Summary, Solutionhood, Restatement, Purpose Condition, Preparation, Circumstance, Result, Enablement, List.

In the DILIGENT methodology we will rely on RST, a brief example for our notation while using RST can be found in Section 3.5.2.

Formal Arguments

The formalization of arguments is a big topic in the AI community. Even though OWL provides us with the necessary formalism to be able to state arguments in a formal way we do not believe that ontological decisions can be discussed in a completely formal way. At least not if the ontology is to be used by humans. [GK97] for example proposes a formal model of argumentation, using IBIS as argumentation model. There, users can derive their preferred solution based on the provided arguments. Another interesting application is the selection of arguments based on the user needs. In [Hun04] a formal model is presented how formal argumentation trees can be pruned to best correspond to the users wishes.

Conflict Mediation

[Eas91] has summarized comprehensively the field of conflict mediation. He gives an introduction to economic and behavioural models to conceptualize and resolve conflicts in discussions. For our future work it is particularly interesting which kinds of conflicts can arise in the area of knowledge acquisition.

[SG89] compares the entity-attribute models of different experts, and identify four types of comparisons between conceptual systems:

- **Consensus:** experts use the same terminology to describe the same concepts
- **Correspondence:** experts use different terminology to describe the same concepts
- **Conflict:** experts use the same terminology to describe different concepts
- **Contrast:** experts use different terminology to describe different concepts

Each of these situations can be useful in capturing different perspectives, and in particular, the availability of alternative terminologies makes a knowledge-base more accessible.

Given these different types which can lead to conflict [Eas91] propose a methodology to resolve the conflicts. The first step is to establish correspondences between different conceptualizations. Afterwards conflicting issues must be identified. They can be discussed using a system like gIBIS. The conflicting issues should be explained externalizing the assumptions behind the decisions and justifying them. Thus goals and motivations become clear to all participants. For conflicting issues resolution criteria should be defined. In a next phase the participants must generate resolution options to resolve the different conflicts. Given the evaluation criteria for the different issues, one can select the best resolution criteria for each issue.

Arguments in Ontology Engineering

The application of argumentation models to ontology engineering is still in its infancy. In [SMD02] a case study in engineering an ontology from the combination of three existing ones is described. The compendium tool is used to guide the discussion in a synchronous meeting. The results of the case study show that structured argumentation in beneficial for ontology engineering. The traceability of the decisions was enhanced. However, the authors were more concerned with the evaluation of their tool than with the specific issues arising in an discussion about an ontology. The authors do not examine which kinds of arguments are exchanged and how the discussion could be made more efficient.

The authors of [ASvE04] propose and evaluate a three-phased knowledge mediation procedure which is especially conceived to integrate different perspectives and information needs into one consensual ontology. The knowledge mediation procedure consists of three main phases. In the generation phase users are jointly brainstorming about relevant concept and instances of the knowledge domain to outline the content of the ontology. During the explication phase each user independently works out a taxonomy by adding definitions and relations to the collected concepts. In the integration phase the knowledge mediator supports the users to integrate their proposed taxonomies into a shared conceptualization. They test the procedure with and without a moderator. With a moderator the participants exchange more elaborated arguments and try to structure their arguments better. They identify useful questions which can guide the actors in the ontological discussion. However, they do not analyse the dominant types of arguments which are used in the discussion.

2.6.2 Methodologies

An extensive state-of-the-art overview of methodologies for ontology engineering can be found in [GPFLC03] from where Table 2.1 has been taken. The book is partially the result of a joint efforts of the OntoWeb[7] members, who produced an extensive state-of-the-art overview of methodologies for ontology engineering (*cf.* [GPFLC+02, FLGPE+02]). There exist also deliverables on guidelines and best practices for industry (*cf.* [LAB+02, LBB+02]) with a focus on applications for E-Commerce, Information Retrieval, Portals and Web Communities.

With respect to our work, especially the following approaches from Table 2.1 are noteworthy. Where it is adequate we give pointers to tools mentioned in the previous section, whenever tool support is available for a methodology.

CommonKADS [SAA+99] 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. CommonKADS is supported by PC PACK, a knowledge elicitation tool set, that provides

---

[7] see `http://www.ontoweb.org/`

Feature			Cyc	Uschold & King	Grüninger & Fox	KACTUS	METHONTOLOGY	SENSUS	On-To-Knowledge	HCOME
Ontology management activities	Scheduling		NP	NP	NP	NP	Proposed	NP	Described	NP
	Control		NP	NP	NP	NP	Proposed	NP	Described	NP
	Quality assurance		NP	NP	NP	NP	NP	NP	Described	NP
Ontology development oriented activities	Pre development processes	Environment study	NP	NP	NP	NP	NP	NP	Proposed	NP
		Feasibility study	NP	NP	NP	NP	NP	NP	Described	NP
	Development processes	Specification	NP	Proposed	Described in detail	Proposed	Described in detail	Proposed	Described in detail	Proposed
		Conceptualization	NP	NP	Described in detail	Proposed	Described in detail	NP	Proposed	Proposed
		Formalization	NP	NP	Described in detail	Described	Described	NP	Described	Proposed
		Implementation	Proposed	Proposed	Described	Proposed	Described in detail	Described	Described	Proposed
	Post development processes	Maintenance	NP	NP	NP	NP	Proposed	NP	Proposed	Described
		Use	NP	NP	NP	NP	NP	NP	Proposed	Described
		Evolution	NP	NP	NP	NP	NP	NP	NP	Proposed
Ontology support activities	Knowledge acquisition		Proposed	Proposed	Proposed	NP	Described in detail	NP	Described	NP
	■ Distributed know. acquisition									Proposed
	■ Onto. Learning integration									
	Evaluation		NP	Proposed	Described in detail	NP	Described in detail	NP	Proposed	NP
	Integration		Proposed	Proposed	Proposed	Proposed	Proposed	NP	Proposed	NP
	Configuration management		NP	NP	NP	NP	Described	NP	Described	NP
	Documentation		Proposed	Proposed	Proposed	NP	Described in detail	NP	Described	NP
	■ Results									
	■ Decision process									Proposed
	Merging and Alignment		NP	NP	NP	NP	NP	NP	NP	Proposed

Table 2.1: Summary of ontology engineering methodologies taken from [GPFLC03]

support for the use of elicitation techniques such as interviewing, *i.e.* it supports the collaboration of knowledge engineers and domain experts.

Cyc [LG90] arose from experience of the development of the Cyc knowledge base (KB)[8], which contains a huge amount of common sense knowledge. Cyc has been used during the experimentation in the High Performance Knowledge Bases (HPKB), a research program to advance the technology of how computers acquire, represent and manipulate knowledge[9]. Until now, this methodology is only used for building the Cyc KB. However, Cyc has different micro-theories showing the knowledge of different domains from different viewpoints. In some areas, several micro-theories can be used, and each micro-theory can be seen from different perspectives and with different assumptions. The Cyc project strongly enhanced the visibility of the knowledge engineering community, but at the same time it suffered from his very high goal to model "the world". Recently this goal has been lowered and now one has divided this too complex task into smaller ones, *i.e.* the Cyc top-level ontology was separated into smaller modules.

DOGMA is one of the more recent modelling approaches [JM02, SMJ02]. The database-inspired approach relies on the explicit decomposition of ontological resources into *ontology bases* in the form of simple binary facts called lexons and into so-called ontological commitments in the form of description rules and constraints. The modelling approach is implemented in the DOGMA Server and accompanying tools such as the DOGMAModeler tool set.

The **Enterprise Ontology** [UK95] [UKMZ98] proposed three main steps to engineer ontologies: (i) to identify the purpose, (ii) to capture the concepts and relationships between these concepts, and the terms used to refer to these concepts and relationships, and (iii) to codify the ontology. In fact, the principles behind this methodology influenced many work in the ontology community and they are also reflected in the steps kickoff and refinement of the OTK Methodology and extended them. Explicit tool support is given by the Ontolingua Server, but actually these principles heavily influenced the design of most of the more advanced ontology editors.

The **KACTUS** [BLC96] approach requires an existing knowledge base for the ontology development. They propose to use means of abstraction, *i.e.* a bottom-up strategy, to extract on ontology out of the knowledge base as soon as an application in a similar domain is built. There is no specific tool support known for this methodology.

METHONTOLOGY [GP96, FLGPSS99] 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

---

[8] Cyc knowledge base, see `http://www.cyc.com`

[9] HPKB, see `http://reliant.teknowledge.com/HPKB/about/about.html`

techniques used, the products to be output and how they are to be evaluated. METHONTOLOGY is partially supported by WebODE.

SENSUS [SRKR97] is a top-down and middle-out approach for deriving domain specific ontologies from huge ontologies. The methodology is supported by Ontosaurus. The approach does not cover the engineering of ontologies as such, therefore offers a very specialized methodology.

TOVE [UG96] proposes a formalized method for building ontologies based on competency questions. The approach of using competency questions, that describe the questions that an ontology should be able to answer, is very helpful and integrated it in OTK Methodology.

HOLSAPPLE In [HJ02] a methodology for collaborative ontology engineering is proposed. The aim of their work is to support the creation of a static ontology. A knowledge engineer defines an initial ontology which is extended and changed based on the feedback from a panel of domain experts. The feedback is collected with a questionnaire. The knowledge engineer examines the questionnaires, incorporates the new requirements and a new questionnaire is send around, until all participants agree with the outcome. Their methodology does not support synchronous collaborative ontology engineering and neither the evolution of ontologies.

HCOME In [KV03, KVA04] the authors present a very recent approach to ontology development. HCONE stands for Human Centered ONtology Environment. It supports the development of ontologies in a decentralized fashion. They introduce three different spaces in which ontologies can be stored. The first one is the *Personal Space*. In this space users can create and merge ontologies, control ontology versions, map terms and word senses to concepts and consult the top ontology. The evolving personal ontologies can be shared in the *Shared Space*. The shared space can be accessed by all participants. In the shared space users can discuss ontological discission based on the IBIS [KR70] model. After a discussion and agreement the ontology is moved to the *Agreed space*.

OTK Methodology In [Sur03] the OTK Methodology is described. This methodology is the result of the EU project OnToKnowledge. The OTK Methodology divides the ontology engineering task into five main steps. Each step has numerous sub-steps, requires a main decision to be taken at the end and results in a special outcome. The phases are “Feasibility Study”, “Kickoff”, “Refinement”, “Evaluation” and “Application & Evolution”. The sub-steps of the e.g. “Refinement” are “Refine semi-formal ontology description”, “Formalize into target ontology” and “Create prototype” etc. The documents resulting from each phase are e.g. for the “Kickoff” phase an “Ontology Requirements Specification Document (ORSD)” and the “Semi-formal ontology description” etc. The documents are the basis for 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 sup- port for the decisions to be taken. The phases “Refinement - Evaluation - Application - 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. All steps of the methodology are supported by tools available for OntoEdit. In a nutshell the OTK Methodology completely describes all steps which are necessary to build an ontology for a centralized system. However the methodology does not cover scenarios where the participants are distributed in several locations. It provides no guidance for systematically evolve an ontology and it does not incorporate automated methods for ontology creation.

Misc For the sake of completeness and without a detailed description we here reference some other proposals for structured ontology engineering. Among them are [PM01] advocating an approach of ontology building by reuse. One of their major findings was that current methodologies offer only limited support for axiom building even so it is a part of ontology engineering which takes a lot of time. In [GPS98] the authors outline the ONIONS approach. ONIONS (ONtologic Integration Of Naïve Sources) creates a common framework to generalize and integrate the definitions that are used to organize a set of terminological sources. In other words, it allows to work out coherently a domain terminological ontology (a terminological ontology is usually defined as the explicit conceptualization of a vocabulary) for each source, which can be then compared with the others and mapped to an integrated ontology library.

2.6.3 Conclusions

From our point of view argumentation visualization is mature from the research perspective. First attempts were made to combine findings from argumentation visualization and ontology engineering. However, as it is argued in [PB88, dMA03] argumentation is best supported when the argumentation model such as IBIS is customized with respect to the domain which is argued about. Hence, research is moving into the following directions.

- Identify the most relevant arguments in ontological discussions.
- Support synchronous as well as asynchronous discussions.

Regarding ontology engineering methodologies, they have similar drawbacks compared to what we said about ontology engineering tools at the end of Section 2.5. Most of them address the engineering of a single ontology without considering multiple views on it and without considering the complex interactions of multiple persons working with an ontology-based application, *e.g.* in form of argumentations. Furthermore, most methodologies consider only a one-time approach without taking dynamics into account. Still an open issue is how to support best the creation and evolution of complex logical axioms, though it remains questionable whether this issue will be addressed in SEKT due to the fact that there is currently a trend for the usage of rather light-weight ontologies in the project.

There are a number of open questions which might guide the future research path beyond the work described in this deliverable.

- Should argumentation visualization and ontology visualization be integrated?
- Can ontology discussions be formalized?
- Does the tracing of arguments enhance understandability of the ontology?
- Can conflict resolution strategies be applied?
- How can automated methods be enable to provide arguments?
- To which extend can moderation be omitted?
- Does argumentation visualization facilitate the evolution of ontologies?
- Is systematic distributed ontology engineering more efficient than unsystematic?
- Which tool support is appropriate?

Chapter 3

DILIGENT Process and Argumentation Framework

We here sketch the DILIGENT process and the argumentation framework (see also [PSST04]). In Section 3.1 we describe a typical scenario for our approach which motivated our work. Following in Section 3.2 is an overview of the DILIGENT process. Each step is the presented in more detail in Section 3.3. The process has already been applied in a real world scenario. Results and lessons learned from the application are described in Section 3.4. Finally, we elaborate on a thorough analysis of a taxonomy evolution in the biology domain. The analysis motivated our extension of the DILIGENT process by an argumentation framework. The hypothesis generated from the analysis was evaluated in two in-situ experiments at the Institute AIFB. The analysis as well as the experiments are described in Section 3.5.

3.1 Motivational Scenario

In *distributed* development there are several experts, with different and complementary skills, involved in collaboratively building the same ontology. For instance, in Virtual Organizations, Open Source and Standardization efforts, experts belong to different competing organizations and are geographically dispersed. In these cases, builders typically are also users and, although some users are not directly involved in changing the ontology, they take part in the process by using the ontology.

An initial ontology is made available and users are free to use it and modify it locally for their own purposes. There is a central board that maintains and assures the quality of the shared ontology. This central board is also responsible for deciding updates, but these are based on user re-occurring changes and requests, therefore the board *loosely controls* the process. It is expected that the change rate of the ontology made available should be higher than the usual due to maintenance, therefore this is a more *evolving* process.

3.2 DILIGENT Overview

We will now describe the general process, roles and functions in the DILIGENT process. As shown in Figure 3.1 it comprises five main activities: (1) **build**, (2) **local adaptation**, (3) **analysis**, (4) **revision**, (5) **local update** (*cf.* figure 3.1). The process starts by having *domain experts*,*users*, *knowledge engineers* and *ontology engineers* **build**ing an initial ontology. In contrast to known ontology engineering methodologies available in the literature [GPS98, GPFLC03, PM01, UK95] our focus is distributed ontology development involving different stakeholders, who have different purposes and needs and who usually are not at the same location. Therefore, they require online ontology engineering support.

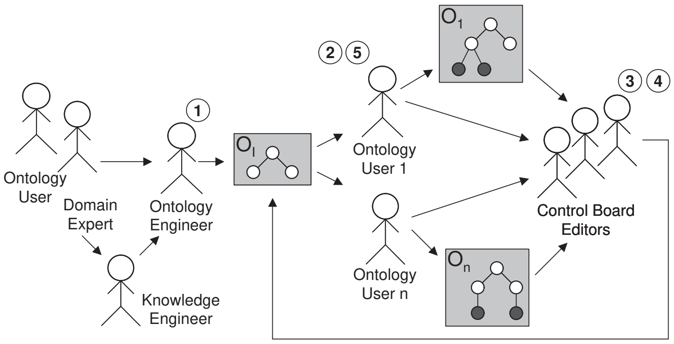


Figure 3.1: Roles and functions in distributed ontology engineering

The team involved in building the initial ontology should be relatively small, in order to more easily find a small and consensual first version of the shared ontology. Moreover, we do not require completeness of the initial shared ontology with respect to the domain.

Once the product is made available, users can start using it and **locally adapting** it for their own purposes. Typically, due to new business requirements, or user and organization changes, their local ontologies evolve in a similar way as folder hierarchies in a file system. In their local environment they are free to change the reused shared ontology. However, they are not allowed to directly change the ontology shared by all users. Furthermore, the control board collects change requests to the shared ontology.

The board **analyses** the local ontologies and the requests and tries to identify similarities in users' ontologies. Since not all of the changes introduced or requested by the users will be introduced,[1] a crucial activity of the board is deciding which changes are going to be introduced in the next version of the shared ontology. The input from users provides the necessary arguments to underline change requests. A balanced decision that takes into account the different needs of the users and meets user's evolving requirements[2] has to

---

[1] The idea in this kind of development is not to merge all user ontologies.

[2] This is actually one of the trends in modern software engineering methodologies (see Rational Unified Process).

be found. The board should regularly **revise** the shared ontology, so that local ontologies do not diverge too far from the shared ontology. Therefore, the board should have a well-balanced and representative participation of the different kinds of participants involved in the process.

In this case, users are involved in ontology development, at least through their requests and re-occurring improvements and by evaluating it, mostly from an usability point of view. Knowledge providers in the board are responsible for evaluating the ontology, mostly from a technical and domain point of view. Ontology engineers are one of the major players in the analysis of arguments and in balancing them from a technical point of view. Another possible task for the controlling board, that may not always be a requirement, is to assure some compatibility with previous versions. Revision can be regarded as a kind of ontology development guided by a carefully balanced subset of evolving user driven requirements. Ontology engineers are responsible for updating the ontology, based on the decisions of the board. Revision of the shared ontology entails its evolution.

Once a new version of the shared ontology is released, users can **update** their own **local** ontologies to better use the knowledge represented in the new version. Even if the differences are small, users may rather reuse *e.g.* the new concepts instead of using their previously locally defined concepts that correspond to the new concepts represented in the new version.

3.3 Detailed Process Description

3.3.1 Local Adaptation: Detailed view

In order to provide a detailed guidance for the participants in the process and to identify potential technical support for the single process steps we have analysed them with more detail. We exemplify the result of this analysis on the *Local adaptation* step, because here technology developed within the SEKT project will have its biggest impact.

The analysis includes the identification of (1) the major roles, (2) the input and (3) output information, (4) the decisions and (5) the actions within the process step.

1. **Roles:** The actors involved in the local adaptation step are users of the ontology. They use the ontology to retrieve e.g. documents which are related to certain topics modelled in the ontology or more structured data like the projects an employee was involved in. Information gathering though is not their main objective, but they rather need the information to fulfil their individual tasks.

2. **Input:** Besides the common shared ontology in the local adaptation step the the information available in the local information space is used. These can be existing databases, ontologies or folder structures and documents.

3. **Output:** The output of the process step is a locally changed ontology which better reflects the users needs. Each change is supported by arguments explaining the reasons for a certain change. We here emphasize that changes are not propagated to the share ontology. Only in the *analysis step* the board gathers all ontology change requests and the corresponding arguments to be able to evolve the common shared ontology in the *revision step*.

4. **Decisions:** The actors must decide which changes they want to introduce into their ontology. Hence, they must decide if and where new concepts are needed and which relations a concept should have. They must further provide reasons why they made certain decisions.

5. **Actions:** To achieve the desired output the user takes different actions namely *Understand shared ontology*, *Identify commonalities between own and shared conceptualization*, *Map equivalent conceptualizations of different actors*, *Identify missing conceptualizations*, *Change conceptualization* and finally *Organize local knowledge according to the conceptualization*.

The last three actions of the process are performed in a cyclic manner until a new common ontology is available and the entire process step starts again.

The single actions performed manually would require a grounded understanding of ontologies and their underlying formal representation. We cannon expect such knowledge from all actors participating in the process. The process should rather be integrated seamlessly in the environment the user works in. Hence we now indicate for each of the actions the available technology to support the actors.

- **Understand shared ontology:** An ontology is a conceptualization of the real world. An ontology should represent a shared conceptualization. In fact a completely shared ontology can never be engineered, since different people have varying interpretations of the real world. Therefore it is necessary as a first action to relate the own interpretation of the world to the shared conceptual model. Thus the actor must learn where the different concepts are located in the ontology and how they are interrelated with other concepts. The ontology can be very complex, thus comprehension of the ontology depends mainly on its presentation. Different technologies can be used to provide the user with a context sensitive view on the ontology which does not overwhelm him. Relevant technology to support this actions are text classification methods, natural language processing and ontology learning methods. We might also consider alternatives to technology driven teaching methods, *e.g.* handbooks or offering a new tip to users each day.

- **Identify commonalities between own and shared conceptualization:** Following the comprehension of the ontology the user can realize the communality between the own and shared conceptualization. We here point to the work of [SG89] which we introduced in section 2.6.1. He identified the different types of conflict when comparing two or more ontologies.

To support this step technically we can use the available formal conceptualization on the local machines. To identify the degree of communality we can use mapping methods to find correspondences between locally available formal models and the shared ontology.

The documents can be operationalised in part for ontology learning which than identifies concepts and relations based on the local text or the documents the user has browsed through.

- **Map equivalent conceptualizations of different actors:** After the identification of commonalities it is necessary to make them explicit. Otherwise the system will not be able to make use of the findings. The expressivity of the used ontology language may set a limit this. For example explicit formalization of mappings is only possible with OWL. RDF(S) does not support it originally. Different implementations may add specialized add-ons. Mappings have the advantage, that they leave the original structures unchanged. Of course users may also decide to change their local structures in favour of the common structure. In this case the changes must be traceable, so that the actor can retain its old version.

- **Identify missing conceptualizations:** Besides the identification of communality the same techniques can be applied in the subsequent step to support the user in identifying missing conceptualizations.

  Depending on the scenario the user might have access to other users ontologies and use their local adaptations as further input to identify missing concepts in his own conceptual model.

- **Add missing conceptualizations:** After identification of missing conceptualizations the user must be enabled to introduce the changes. This is not so much a technical challenge than a one of usability. The user should not be bother with suggestions all the time and he might not tolerate wrong suggestion. Since automated methods can not be 100% correct it depends on the user context when and how to apply the changes to the ontology.

  The board analysis the changes performed by the users. To be able to understand the change requests the actor should provide reasons for each request. Again the rationales used within the automated methods can be used as input here. To support the user further in providing reasons a part of the process model is an argumentation framework to focus the user on the relevant arguments he can provide.

- **Organize local knowledge according to ontology:** At this point the ontology should reflect the users conceptualizations. Now he can instantiate the ontology with the information available locally and hence contribute to the collective knowledge. Again text classification and natural language processing can be used to facilitate this action.

The implementation of tools to support the single actions must be done in close cooperation with the user and with respect to usability. The steps are complex so that an easy way must be found to enable the users to follow the process.

We have shown how different techniques developed in the course of the SEKT project can be used within the process to support the actors to follow the process. The output is a locally adapted ontology. Hence the board can retrieve the changes and analyse the reasons underlying each change. The reasons can either provided automatically by the supporting methods or manually following the argumentation model describe with more detail in section 3.5.

3.3.2 Analysis

Depending on the frequency and volume of changes the board will make adjustment cycles as needed.

1. **Roles:** In the analysis phase we can distinguish three roles. The domain expert will decide which changes to the common ontology are relevant for the domain and which are relevant for smaller communities, only. A representative of the users explains different requirements from the usability perspective. The control board decides about changes introduced in the shared ontology. We see the board as an integral part of DILIGENT which cannot be left out. However, we are currently thinking which of the tasks of the board might be supported by tools or even delegated to software agents.
2. **Input:** The analysis step takes as input the ontology changes proposed by the participating actors. Furthermore, the arguments underlying the proposed changes are an important input for the board to be able to make a well balanced decision which ontology changes should be introduced.
3. **Output:** The result of the analysis step is a list of the major changes introduced by the actors. Hence, all changes which should not be introduced into the shared ontology are filtered. In this step it is not required to decide the final modelling.
4. **Decisions:** The board must decide which changes should be introduced into the new shared ontology.
5. **Actions:** To achieve the desired output the board takes different actions namely *Gather locally updated ontologies and corresponding arguments*, *Analyse the introduced changes* and *Identify changes relevant for all actors*.

- **Gather locally updated ontologies and corresponding arguments:** Depending on the deployed application the gathering of the locally updated ontologies can be more or less difficult. It is important that the board has access to the local changes to be able to analyse them. In a centralized ontology based system in which users can make their changes within their workspace this task is very easy. In peer-to-peer scenarios some peers might not always be reachable, to be able to collect the local ontologies. For the board it might also be interesting not only to analyse the final changed ontology, but also the evolution process. However, with an increasing number of participants this in-depth analysis might not be feasible.

- **Analyse the introduced changes:** We expect that the number of change requests is huge and probably contradictory. As a first provision the board must identify the different areas in which changes took place. Clustering techniques which take into account the arguments underlying the proposed changes might be helpful here. Within the analysis the board should bear in mind that changes to concepts should be analysed before relations and again before axioms. Frequency of changes and overlapping changes are a good indicator for the relevance of the change to the users. Furthermore, the board should analyse the queries generated from the ontology. This should help to find out which parts of the ontology are more often used. Since actors can instantiate the ontology locally, the number of instances for the different proposed changes can also be used to determine the relevance of certain adaptations.
- **Identify changes relevant for all actors:** Having analysed the changes and grouped them according to different parts of the ontology the board can identify the most relevant changes. In the list of proposed changes there are probably also contradictions. Hence, the board must decide based on the provided arguments which changes should be introduced. Depending on the quality of the arguments the board itself might argue about different changes. The outcome of this action must be a reduced and structured list of changes which should be modelled in the ontology.

3.3.3 Revision

While we could evaluate the local update and analysis step of our process model already in small experiments the revision phase and local update phase are not well tested yet. Hence, due to the early stage of the project the revision phase and the local update phase are not yet very well elaborated. We here just sketch some general observations and point our future directions of research.

1. **Roles:** The ontology engineer judges the changes from an ontological perspective. Some changes relevant for the common ontology, but placed at the wrong place by the users. The domain experts should judge and decide whether new concepts/relations should be introduced into the common ontology even so they were not requested by the users.

2. **Input:** The input for the revision phase is a list of changes which should be included into the ontology.
3. **Output:** The revision phase ends when all changes are formalized and well documented in the common ontology.
4. **Decisions:** The main decisions in the revision phase are formal ones. All intended changes identified during the analysis phase must be included into the common ontology. In the revision phase the ontology engineer decides how the requested changes should be formalized in ontological way.
5. **Actions:** To achieve the desired output the user takes different actions namely *Formalization of the requested changes*, *Aggregation of arguments* and *Documentation*.

- **Formalization of the requested changes:** Similar to established methodologies the requested changes must be formalized with respect to the expressivity of the ontology. We will not go into detail with this step since it is already described in methodologies referred to in the related work section.
- **Aggregation of arguments:** As arguments play a major role in the decision process we expect that the changes which are eventually included into the common ontology are supported by many arguments. One of the reasons for keeping track of the arguments is to enable users to better understand why certain decisions have been made with respect to the ontology. Hence, the the user should be able to retrieve the most convincing arguments made to introduce a certain change.
- **Documentation** With the help of the arguments, the introduced changes are already well documented. However, we assume that some arguments might only be understandable for the domain expert and not for the users. Hence, we expect that the changes should be document to a certain level.

3.3.4 Local Update

1. **Roles:** The local update phase involves only the users. They perform different actions to include the new common ontology into their local system before they start a new round of local adaptation.
2. **Input:** The formalized ontology including the most relevant change request is the input for this step. We also require as an input the documentation of the changes. For a better understanding the user can request a delta to the original version.
3. **Output:** The output of the local update phase is an updated local ontology which includes all changes made to the common ontology. However, we do not require the

users to perform all changes proposed by the board. Hence, the output is not mandatory, since the actors could change the new ontology back to old one in subsequent local adaptation step.

4. **Decisions:** The user must decide which changes he will introduce locally. However this interferes with the local adaptation step.
5. **Actions:** To achieve the desired output the user takes different actions namely *Distribution of the ontology to all actors*, *Tagging of the updated version*, *Inclusion of the updated version* and *Update of local adaptations which are not included in the common ontology*.

- **Distribution of the ontology to all actors:** Analogously to step 3.3.2 the shared ontology must be distributed to the different participants. Depending on the overall system architecture different methods can be applied here.
- **Tagging of the updated version:** To ensure user satisfaction, the system must enable the user to return to his old version of the ontology at any time. The user might realize that the new updated version of the common ontology does not represent his needs anymore and thus want to leave the update cycle out. To reach a better acceptance this must be possible and is foreseen in the methodology. The user can always balance between the advantages of using a shared ontology or using his own conceptual model.
- **Inclusion of the updated version:** The system must support the user to easily integrate the new version into his local system. It must be guaranteed that all annotations made for the old version of the ontology are available in the new version.
- **Update of local adaptations which are not included in the common ontology:** The update of the local ontology can lead to different kinds of conflict. Changes proposed by the user may indeed have found their way into the common ontology. Hence, the user should be enabled to use from now on the shared model instead of his own identical model. Furthermore, the board might have included a change based on arguments the user was bringing forward, but has drawn different conclusions. Here the user can decide whether he prefers the shared interpretation. Other option might emerge in the course of the case studies.

3.4 First Application of DILIGENT

We applied the DILIGENT process in a peer-to-peer (P2P) case study of the SWAP project[3]. In the case study up to 7 organization with up to 28 peers took part. The case

---

[3] see `http://swap.semanticweb.org/`

study lasted for two weeks (cf. [TPSS04]). Ontologies were used to represent the local (folder) structures of each peer, whereby each peer represented a single user. On top of these local views a shared ontology was created and evolved according to the DILIGENT methodology to facilitate searching and querying over the P2P network.

Building. In the case study two knowledge engineers were involved in building the first version of the shared ontology with the help of two ontology engineers. In this case, the knowledge engineers were at the same time also knowledge providers. In addition they received additional training such that when the P2P network was up and running on a bigger scale, they were able to act as ontology engineers on the board – which they already are doing in later stages of this case study not reported here.

The ontology engineering process started by identifying the main concepts of the ontology through the analysis of competency questions and their answers. The most frequent queries and answers exchanged by peers were analysed. The identified concepts were divided into three main modules: “Sustainable Development Indicators”, “New Technologies” and “Quality&Hospitality Management”. From the competency questions we quickly derived a first ontology with 22 concepts and 7 relations for the “Sustainable Development Indicator” ontology. This was the domain of the then participating organizations. Recently the other modules have been further elaborated.

Based on previous experience of IBIT with the participants we could expect that users would mainly specialize the modules of the shared ontology corresponding to their domain of expertise and work. Thus, it was decided by the ontology engineers and knowledge providers involved in building the initial version that the shared ontology should only evolve by addition of new concepts, and not from other more sophisticated operations, such as restructuring or deletion of concepts.

Local Adaptation. The developed core ontology for “Sustainable Development Indicator” was distributed among the users and they were asked to extend it with their local structures. With assistance of the developers they extracted on average 14 folders. The users mainly created sub concepts of concepts in the core ontology from the folder names. In other cases they created their own concept hierarchy from their folder structure and aligned it with the core ontology. They did not create new relations. Instance assignment took place, but was not significant. We omitted the use of the automatic functions to get a better grasp of the actions the users did manually.

Analysing. The members of the board gathered the evolving structures and analysed them. The following observations were made:

- **Concepts matched:** A third of the extracted folder names was directly aligned with the core ontology. A further tenth of them was used to extend existing concepts.
- **Folder names indicate relations:** In the core ontology a relation inYear between the concept Indicator and Temporal was defined. This kind of relation is often encoded in one folder name. *e.g.* the folder name “SustInd2002” matches the

concepts Sustainable Indicator and Year[4]. It also points to a modelling problem, since Sustainable Indicator is a concept while "2002" is an instance of concept Year.

- **Missing top level concepts:** The concept project was introduced by more than half of the participants, but was not part of the initial shared ontology.
- **Refinement of concepts:** The top level concept Indicator was extended by more than half of the participants, while other concepts were not extended.
- **Concepts were not used:** Some of the originally defined concepts were never used. We identified concepts as used, when the users created instances, or aligned documents with them. A further indicator of usage was the creation of sub concepts.
- **Folder names represent instances:** The users who defined the concept project used some of their folder names to create instances of that concept *e.g.* "Sustainable indicators project".
- **Different labels:** The originally introduced concept Natural spaces was often aligned with a newly created concept Natural environments and never used itself.
- **Ontology did not fit:** One user did create his own hierarchy and could use only one of the predefined concepts. Indeed his working area was forgotten in the first ontology building workshop.

From the discussions with the domain experts we have the impression that the local extensions are a good indicator for the evolution direction of the core ontology. However, since the users made use of the possibility to extend the core ontology with their folder names, as we expected, the resulting local ontologies represent the subjects of the organized documents. Therefore, a knowledge engineer is still needed to extend the core ontology, but the basis of his work is being improved significantly. From our point of view there is only a limited potential to automate this process.

Revision. The board extended the core ontology where it was necessary and performed some renaming. More specifically the board introduced one top level concept (Project) and four sub concepts of the top level concept Indicator and one for the concept Document. The users were further pointed to the possibility to create instances of the introduced concepts. *E.g.* some folder names specified project names, thus could be enriched by such an annotation.

Local update. The extensions to the core ontology were distributed to the users. The general feedback of the users was generally positive. However, a prolonged evaluation of the user behaviour and second cycle in the ontology engineering process is still being performed.

---

[4] Year is sub class of class Temporal

LESSONS LEARNED: The case study helped us to generally better comprehend the use of ontologies in a peer-to-peer environment. First of all our users did understand the ontology mainly as a classification hierarchy for their documents. Hence, they did not create instances of the defined concepts. However, our expectation that folder structures can serve as a good input for an ontology engineer to build an ontology was met.

Currently we doubt that our manual approach to analysing local structures will scale to cases with many more users. Therefore, we are currently evaluating approaches to automatically recognizing similarities in user behaviour. Furthermore, the local update will be a problem when changes happen more often. We have so far only addressed the ontology creation task itself – we are currently measuring if users get better and faster responses with the help of DILIGENT-engineered ontologies. All this is current work.

In spite of the technical challenges, user feedback was very positive since the tool was integrated into their daily work environment and could be easily used and the tool provided very beneficial support to perform their tasks. However, it will require the introduction of some new features in order to ease ontology editing tasks by users without a knowledge engineering background.

3.5 Argumentation Framework for DILIGENT

In this section we describe how we specifically investigated whether some argumentation structures dominate the progress in the ontology engineering task and should therefore be accounted for in a fine-grained methodology.

3.5.1 Threads of Arguments

A central issue in the DILIGENT process is keeping track of threads of exchanged arguments. We can identify several stages in which arguments play an essential part:

- Ontology is defined as “a shared specification of a conceptualization” [Gru95]. Although “shared” is an essential feature, it is often neglected. In DILIGENT experts exchange arguments while **build**ing the initial shared ontology in order to reach consensus;
- When users make comments and suggestions to the control board, based on their **local adaptations**, they are requested to provide the arguments supporting them;
- while the control board **analyses** the changes introduced and requested by users, and balances the different possibilities, arguments are exchanged and balanced to decide how the shared ontology should change.

There is evidence that distributed ontology development can be rather time consuming, complex and difficult, in particular getting agreement among domain experts. Therefore, one needs an appropriate framework to assure it in a speedier and easier way. In order to provide better support, one needs to identify which kind of arguments are more relevant and effective to reach consensus. The Rhetorical Structure Theory (RST) can be used to classify the kinds of arguments most often used and identify the most effective ones.

3.5.2 RST Example

The RST has already been introduced in Section 2.6. In the examples provided within the case study section we will highlight the different elements of RST in the following way.

span nucleus ... relation indicator ...
`span satellite` **Relation**

On the one hand we have presentational relations, such as **background** that increases the ability of the reader to comprehend an element in N, **evidence**, where reader's comprehension of S increases his/her belief of N, **justify**, **restatement**, **summary**, etc. On the other hand we have subject-matter relations, such as **elaboration**, where S presents additional detail about what is presented in N, for instance set::member; abstraction::instance; whole::part, object::attribute, etc., **evaluation**, **purpose**, **solutionhood**, etc. There are also other relations that do not carry a definite selection of one nucleus, such as **contrast**, where the reader recognizes the comparability and differences in situations described in two N, etc.

The analysis process is intended to give a structured, definite way for a person to understand the text to state a part of what that understanding includes. Sometimes one may not find some structural role for every element of the text. A text may have more than one analysis, either because the observer finds ambiguity or finds that a combination of analyses best represents the author's intent. The analysis gives an account of textual coherence that is independent of the lexical and grammatical forms of the text.

The available tools are tables that explain the relations between spans of text. Therefore the analysis process is manual, intensive and requires understanding of natural language.

3.5.3 Analysis in the Biology Domain

In this section we report how the hypothesis underlying DILIGENT argumentation model has been developed. We have found in the field of biology a taxonomy that has been evolving since 1735 for over 200 years, following a DILIGENT 5-step process. A thorough analysis led to a well-defined subset of RST arguments which allows to explain most of the evolution of the biology taxonomy. In the following Section 3.5.4 we describe experiments which have been carried out to provide evidence that our approach is applicable in practice and the reduction of arguments leads to a better process instantiation and better results.

Based on the RST analysis of real arguments that are exchanged and used to support changes in this taxonomy, we formulated as hypothesis that there is a subset of arguments that can focus, speed and ease this kind of ontology engineering. In order to prove our hypothesis we performed an *in situ* experiment in two rounds. In the first one participants were not constrained. In the second one participants were requested to use the subset of arguments that had been found more effective in the first round. We show the improvements that were achieved using the restricted set of arguments, proposed in the fine-grained DILIGENT model of ontology engineering by argumentation.

The taxonomy of living things is essential for those studying, classifying and understanding life. When we analyse its evolution since 1735 one notes that it completely follows the 5-step DILIGENT process. It was initially proposed/**built** by Linnaeus based on phenetics (observable features). Each branch of the tree can have at most 26 levels, depending on how rich a taxa is, in terms of number of beings sharing a given classifying feature. Since the initial proposal, the taxonomy has changed a lot. Let us take the “highest” level: kingdom. Initially two taxa were identified: animals and plants. When microorganisms were discovered the moving ones were classified in the animals kingdom and the colored (non moving) ones in the plants kingdom. A few of them were classified in both kingdoms. Users were **locally adapting** the taxonomy for their own purposes. To more easily identify organisms in both classes, Haeckel (1894) proposed a new kingdom to more easily identify them, the Protista kingdom. This still exists today and is regarded as a “junk-basket” category.

Naming is an important issue. Lineaus binomial system (genus and species) is still in use, because it can univocally identify a given being in the taxonomy.[5] Given the difficulty and similarity of some names, the ever evolving new knowledge about ever growing number of life forms, and the difficulty of making available up-to-date knowledge to all stakeholders about so many life forms, several problems in designing and managing this complex and live/dynamic taxonomy arose. For some time, names of plants and animals have been controlled by different **boards**, that have to some extent, recorded the problems and solutions found for each kingdom. They receive requests for changes, **analyse** them, balance pros and cons, decide upon the most adequate changes to introduce and **revise** the taxonomy accordingly. Once a new version is made available users should use it/**locally update**.

After being divided for two centuries and being controlled by two different boards, there were some communication problems between the two communities. Given the availability of online information about lifeforms and the need to exchange information about new results, the need to develop a common language and a BioCode arouse. This effort

---

[5] One can reuse names in different kingdoms.

is now beginning.

So, the evolution of the taxonomy is driven by a specialized set of users, taxonomists, and the revision is loosely controlled by appropriate boards, that make new versions available for all users.

In this case the central board is the scientific community, the peers, who **analyse** the different proposals to explain new knowledge and accomodate new life forms, and once in a while **revise** the common understanding of the domain.

One can summarize the major force for reorganization of the taxonomy over time as the identification of important classifying features and gathering all beings sharing a given value for that feature into that class. For instance, the classical version by Whittaker (1969) recognizes 5 kingdoms: Monera, Protista, Plantae, Animalia and Fungi. Regarding all eukaryotic organisms, Plantae, Animalia, Fungi and Protista, the first three, classify multicellular organisms according to nourishment, autotrophic, heterotrophic and saprotrophic, respectively. Fungi were promoted from one subclass (taxa) in the Plantae kingdom to a kingdom of its own.

However, there are currently more advanced classifications, that is, several classifications coexist. Therefore, classes can be promoted, moved, folded, deleted, merged, renamed, etc. as more is known about life on earth.

Currently, given the advances in molecular biology, the tendency is to use a cladistic approach, in which the taxonomy is organized according to the evolutionary relationships between live forms based on derived similarity. In a cladogram, each split is ideally binary (two-way), and all the organisms contained in any one clade share a unique ancestor for that clade. This entails a major reorganization of the Tree of Life. The reason is that the design decisions are radically different from the previous approach.

EXAMPLE When analysing the arguments exchanged by taxonomists to change the names and organization of the taxonomy one can perceive its vast array and complexity.[6]

> ...*Acinetosporaceae*, including the genera `Acinetospora, Feldmannia, ...` **Elaboration**
> *This group* forms a well-supported clade in molecular trees based on `rbcL data`. **Evidence**
> So far, trees from nuclear ribosomal data do not reveal `them` as *a well-supported group* **Antithesis**
> but are not contradictory to *their* recognition. **Concession**
> ...

DISCUSSION The analysis of the arguments driving the evolution of the taxonomy of life on earth led to the assumption that RST could be useful to analyse arguments exchanged in ontology building process in distributed environments.

---

[6] Example taken from `http://www.ncbi.nlm.nih.gov/Taxonomy/taxonomyhome.html/index.cgi?chapter=CHANGETOCLASS`

From an arguments point of view, the focus of this paper, we can see that although elaborated, there are a few arguments in the biology case study which play a major role, such as examples/evidence, counter examples, elaboration, alternatives and comparisons to convey a certain decision.

3.5.4 Evaluation of Argumentation Framework

In order to substantiate our hypothesis that an appropriate argumentation framework can facilitate the ontology engineering process, we pursued experiments in a computer science department, viz. at the Institute AIFB[7]. Arguments in collaborative, distributed settings take place in a social environment. Therefore organizational issues are non negligible and were also taken into account.

We performed two experiments: in the first, participants were not constrained in any way; in the second, participants were asked to (1) use a subset of arguments, those that that had been found more effective in the first round,(2) and were given stricter rules, and a better environment to conduct their discussions. The task in both sessions was to build an ontology, which (1) represents the knowledge available in the research group, (2) can be used for internal knowledge management, (3) and makes the research area comprehensible for outsiders. Both experiments lasted each for one hour and a half. From the eleven participants - all from the computer science department, thus domain experts - three were unexperienced in ontology engineering. Seven of them were very active in both discussions. Concepts were only added after argumentation and some consensus was achieved.

First Experiment

The goal of the first experiment was to identify the dominant arguments used to push forward ontology development.

1. **Setting:** The participants met in a virtual chat room. Each one had their own client and all of them could see the current ontology. All arguments were exchanged via the chat room, no other forms of communication were allowed. A moderator was responsible to remind people to stay on the subject and to include the modelling decisions into the formal ontology which was visualized on a web page. At this stage very few procedural and methodological restrictions were a-priori imposed. The subjects were instructed of the high level goal of the experiment, of the procedure and of their goals.
2. **Example:**[8] An excerpt from the real dialogues taking place:

---

[7] see `http://www.aifb.uni-karlsruhe.de/`

[8] We have changed the transcripts a bit, for the sake of readability.

> ...
> **sa :** i dont care whether *someone* plays baseball or not when I am modelling `research domain`. **Evaluation**
> **cs :** **sa** just an example... **Circumstance**
> **ct :** maybe it is the purpose of *the website*, that people get also `informed about hobbys` **Purpose**
> **cs :** so we have person **Restatement**
> **jt :** what I find a bit more interesting is `the conference problem` **Motivation**
> ...

3. **Result:** In the beginning participants brought forward different kinds of arguments, like background knowledge, examples, elaboration and so on. This led to different argumentation threads where participants were discussing different topics at the same time. At some points there were 4 threads at the same time, most of the time there was more than one, including procedural and noise. Therefore, discussion was very tangled and at some points rather difficult to follow. Topics which were discussed included: the appropriate formalism to model the ontology, detailed elaboration of leaf concepts, which top level concepts to begin with, philosophical modelling decisions (roles vs. multi inheritance), which are the main modules, topic lists etc. From time to time participants called for a vote. However a decision was seldom reached. The moderator interacted only rarely in the discussion, because timely moderating multiple threads is very difficult: by the time an intervention was issued two or three other interventions from participants had already been issued. As a result, a core ontology with two concepts, Role and Topic, was agreed upon.

LESSONS LEARNED: We analysed the discussion with the help of RST. Table 3.1 lists the frequency of the different arguments exchanged during the experiment. We could identify the arguments which had most influence on the creation of the ontology, *viz.* **elaboration**, **evaluation/justification**, **examples**, **counter examples**, **alternatives**.

With respect to the experimental setup we identified the following problems: (1) Participants started too many discussion threads and lost the overview, (2) the discussion proceeded too fast, hence not everybody could follow the argumentation, (3) the moderator was too reluctant to intervene, (4) there was no explicit possibility to vote or make decisions. Even in this setting where participants shared a very similar background knowledge, the creation of a shared conceptualization without any guidance is almost impossible, at least very time consuming. We concluded, that a more controlled approach is needed with respect to the process and moderation.

Arguments	**First Round**	**Second round**
Elaboration	24	36
Eval. & Just.	14	33
Contrast & Alternative	12	17
Example	12	9
Counter Example	10	8
Background knowledge	9	3
Motivation	5	
Summary	5	3
Solutionhood	4	8
Restatement	3	6
Purpose	3	
Condition	2	
Preparation	1	
Circumstance	1	
Result	1	
Enablement	1	
List	1	1
Concepts agreed on	2	10
Relations agreed on	3	0

Table 3.1: Arguments used and outcome

Second Experiment

The goals of the second experiment, were to underline that with an appropriate argumentation framework the ontology creation proceeds faster, more effectively and the resulting ontology represents a shared view.

1. **Setting:** In the second experiment participants were asked to extend the ontology built in the first round. In this phase the formalism to represent the ontology was fixed. The most general concepts were also initially proposed, to avoid philosophical discussions. The initial ontology defined the modelling primitives for topics and the different roles people are involved in. For the second round the arguments **elaboration**, **examples**, **counter examples**, **alternatives**, **evaluation/justification** where allowed.

   The participants in the second case study joined two virtual chat rooms. One was used for providing topics for discussion, hand raising and voting. The other one served to exchange arguments. When the participants - the same as in the first experiment - wanted to discuss a certain topic *e.g.* the introduction of a new concept, they had to introduce it in the first chat room. The topics to discuss were published on a web site, and were processed sequentially. Each topic could then be expatiated with the allowed arguments. Participants could provide arguments only after hand raising and waiting for their turn. The participants decided autonomously when a topic was sufficiently discussed, called for a vote and thus decided how to model a certain aspect of the domain. The evolving ontology was again published on a web site. The moderator had the same tasks as in the first experiment, but was more restrictive. Whenever needed, the moderator called for an example of an argument to enforce the participants to express their wishes clearly.

2. **Example:** An example from the arguments window:

   ...
   cs : We have done quite a bit of research in distributed knowledge management (DKM) lately. So I suggest DKM as a topic plus a subtopic “peer to peer” (P2P) **Elaboration**
   ah : I suggest *knowledge management (KM)* as super concept of `DKM` because `every DKM` is a kind of *KM* **Elaboration, Justification**
   jt : Well I am now wondering whether *P2P is DKM*, because `File exchange` is not always KM is it? **Counterexample**
   ph : I suggest *Distributed Comp. (DC)* with `P2P and Grid` as subtopics; *DKM* as subtopic of `DC and KM` **Elaboration**
   do : PRO **ph** : because his approach *separates KM* and `distributiveness` **Justification, Evaluation**
   cs : I’d like to agree to **ph** and **do** suggestion. ...
   ad : km is a kind of kp **Elaboration**
   h : can you elaborate?

> **schm4704 :** I think we can subsume kdd and km under kp, because both are disjoint, and still both related to knowledge processing (as far as I see it). **Justification, Comparison**
> **pc :** knowledge managament is about managing knowledge, structuring, organizing etc. to organize it you need to process it so for sure KP is more special that KM **Elaboration, Justification, Comparison**
> **ad :** I did not understand your argument! Why should kp be more specal than km. If I manage something than it is also a kind of process**Counter Example**
> **pc :** KP is a part of KM so we should model it as a subpart or something **Justification**
> ...

3. **Result:** As expected the discussion was more focused, due to the stricter procedural rules. Agreement was reached quicker. A total of ten new concepts were agreed on. With the stack of topics which were to be discussed (not all due to time constraints), the focus of the group was kept. Some relations were proposed, but they were not agreed upon.

   From a methodological point of view, one can classify the ontology engineering approach followed as **middle-out**. The restricted set of arguments is easy to classify and thus the ontology engineer was able to build the ontology in a straightforward way. It is possible to explain new attendees why a certain concept was introduced and modelled in such a way. It is even possible to state the argumentation line used to justify it. The participants truly shared the conceptualization and did understand it. In particular in conflict situations when opinions diverged the restriction of arguments was helpful. In this way participants could either prove their view, or were convinced.

LESSONS LEARNED: Our experiments provide strong indication – though not yet full-fledged evidence – that a restriction of possible arguments can enhance the ontology engineering effort in a distributed environment. In addition the second experiment underlines the fact that appropriate social management procedures and tool support help to reach consensus in a smoother way. From an RST analysis perspective, the fact that the discussion was more focused eased the task enormously.

Another rather interesting conclusion is the fact that a middle-out approach comes naturally for people with knowledge engineering skills when given an appropriate work environment. Moreover, middle-out combined with appropriate argumentation and management can be used to quickly find a shared, consensual ontology even when participants must provide all and only written arguments.

The process could certainly be enhanced with better tool support. Besides the argumentation stack, an alternatives stack would be helpful. Arguments in particular **elaboration**, **evaluation & justification** and **alternatives** were discussed heavily. However, the lack of appropriate evaluation measures made it difficult, at some times, for the contradicting opinions to achieve an agreement. The argumentation should then be focused on the evaluation criteria. The evaluation can take place off-line, or can be based on modelling advices from practical experience. Discussion can proceed. As to the use of the RST to analyse real dialogues, instead of carefully written texts, one should mention, in particular in the first round where the discussion was rather tangled, that it was rather difficult to classify at some parts. However, the restricted set is easy to identify and we conjecture that the provision of template arguments will ease the task further. In both rounds one should stress the lack of tools to automatize it, although one can foresee the difficulty, since this kind of analysis requires deep NL understanding.

Chapter 4

DILIGENT in SEKT

In this chapter we will take a look at how the DILIGENT process is applied or will be applicable to the three case studies of SEKT. We will see, that different parts of DILIGENT suit best the specifics of each case study. We will identify these parts and interpret the work done so far through the DILIGENT framework view.

We will benefit threefold from applying an explicit ontology development process like DILIGENT to the case studies:

1. specific problems of the case studies are addressed and attacked with DILIGENT, for example in the legal case study, where the argumentation framework has been already applied in order to focus the discussion while creating the initial version of the legal ontology,
2. using a framework like DILIGENT helps in describing the ontology lifecycle within the use cases in a more concise and generic way, in order to gain accessibility to the experiences won in the use cases, and
3. for the further development of DILIGENT, we will be able to harvest experience with using the process and get feedback in order to identify problems and refine the steps of DILIGENT.

Besides the use cases, the development of an upper level ontology for SEKT became a further interesting application of the DILIGENT process. In the following sections, we will start with a description of the three case studies with regards to the above listed benefits, and then analyse the ontology development of the SEKT upper level ontology PROTON, to see, if we may gain the same advantages here as well.

4.1 Legal Case Study

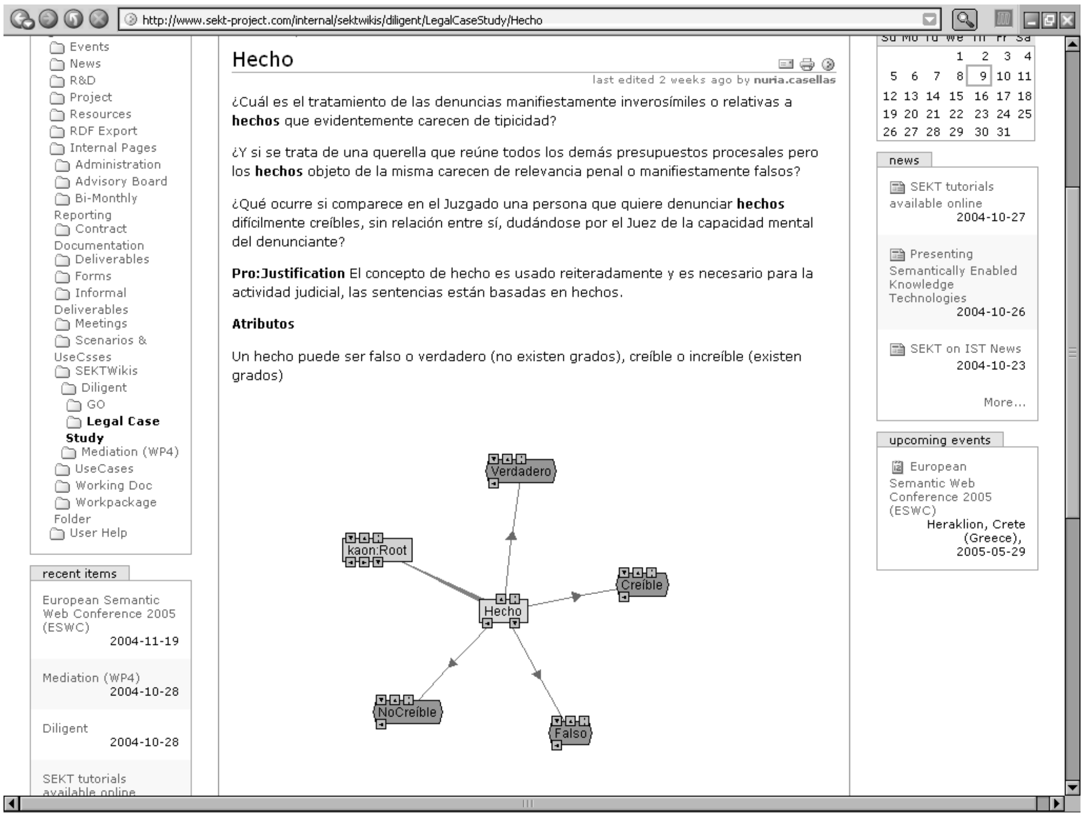
The goal of the legal case study is to provide support to professional judges. In the Spanish system one particular problem young judges face is when they are on duty and confronted with situations in which they are not sure what to do (e.g. in a voluntary confession, which process of questioning should be applied?). In such cases, they usually phone their former training tutor (experienced judges) for resolving the issue. But this is a slow and insecure procedure. In this case study, it is planned to develop an intelligent system to speed up the process and to relieve experienced judges from this effort by providing support to young judges. Only in the case that the requested knowledge is not in the system and cannot be reformulated from already stored knowledge, an experienced judge will be contacted. The result of this "expensive" consultation will be fed back into the system automatically.

In order to build a scalable and useful system, several requirements have been identified [RCCP04]. In particular, the decision as to whether a request for knowledge is covered by the knowledge stored in the system should be based on semantics of the legal domain rather than on simple word matching. An ontology can be used to perform this semantic matching. Case-based reasoning techniques will be used when considered applicable. The main difference with traditional CBR approaches is that we will use a semantic-based similarity measures based on ontologies. Moreover, cases will be - where possible - automatically extracted from information generated by judges as they perform their daily work.

The ontology used for this semantic matching, the OPLK (Ontology for Professional Legal Knowledge) [BCC$^+$05], is being developed in the SEKT project. When modelling the OPLK, the modelers soon realized that they were often discussing questions at length before making a decision. But after a few weeks, the decision was not traceable any more, and they started discussing the original question anew. The argumentation framework of DILIGENT – as presented in 3.5.2 – was identified as being able to help solving problems such as missing traceability and explicitness.

As suggested for the discussion process, we offered an easily accessible web based interface in order to allow the discussion in a traceable way. A standard wiki was used which supports seamless discussion and offers ease of use. But the users quickly extended it in combination with the KAON OIModeller [GSV04]: they modelled the ontologies agreed on, made a snapshot of part of ontology and imported it to the wiki, in order to visualize the ontology and ease the discussion and understanding of it. In figure 4.1 we see a screenshot of the wiki, running on the SEKT portal, showing the concept Hecho with its description, argumentation and attributes, as well as a graph made with the KAON OIModeller showing the concept and its attributes.

This points us to deficiencies and strengths of using standard off the shelf products. The partners of the legal case study obviously demand graphical, intuitive and easy accessible user interfaces. They want tools that help them with the specific tasks they encounter when dealing with ontology engineering. But, on the other hand, due to their general use



Hecho

last edited 2 weeks ago by **nuria.casellas**

¿Cuál es el tratamiento de las denuncias manifiestamente inverosímiles o relativas a **hechos** que evidentemente carecen de tipicidad?

¿Y si se trata de una querella que reúne todos los demás presupuestos procesales pero los **hechos** objeto de la misma carecen de relevancia penal o manifiestamente falsos?

¿Qué ocurre si comparece en el Juzgado una persona que quiere denunciar **hechos** difícilmente creíbles, sin relación entre sí, dudándose por el Juez de la capacidad mental del denunciante?

Pro:Justification El concepto de hecho es usado reiteradamente y es necesario para la actividad judicial, las sentencias están basadas en hechos.

Atributos

Un hecho puede ser falso o verdadero (no existen grados), creíble o increíble (existen grados)

Figure 4.1: Screenshot of the wiki used for discussing the legal ontology

outside of the relatively small community of ontology engineers, the software is stable, it is easy to get help on using it and it may show surprisingly flexible features.

4.2 Digital Library Case Study

BT began building its Digital Library in 1994 and over ten years has developed an online system that offers its users personalisation, linking to full text from abstracts, annotation tools, alerts for new content, and the foundations of profiling. A key driver in developing the Library has been the desire to provide a single interface to the whole collection, drawing together content from a wide variety of publishers.

The BT Library allows its user to search the library's contents. In addition, they can browse through "information spaces" that have been created on topics known to be of interest to people in the company or through the contents of journals in the Library. Information Spaces bring together content from the library's databases and details of new books into a single page in the Library. People can "join" an information space to be alerted to new articles on the topic and can create their own private information spaces for topics of particular interest to them [AO98].

We can describe the change management process with DILIGENT. After the initial **build** of the ontology, it is delivered to the users. The user profile is meant to be highly adaptive, actively - with the user creating information spaces and thus expanding the ontology - and passively - with the system logging and analysing the users actions, and thus further adapting the user profile with this usage driven changes [HV04]. This step represents the **local adaptation**. **Analysing** these adaptations by the board responsible for the quality of the Digital Library will show up the deficits of the current ontology and allow to **revise** it properly, in order to achieve a higher usability and a better performance by changing the topic relations, especially the hierarchy, and by keeping the information spaces close to the current interests of the users and easy accessible. This way the board has constant feedback on the interests of the users, who **update** their data accordingly.

In this case study the argumentation procedure is actually replaced by statistics. Users are not supposed to argue for or against new topics, but instead will simply decide by using a topic or not if it was well chosen or not.

4.3 Siemens Case Study

The objective of this case study is to investigate and verify how semantically enabled technologies can improve the productivity of IT and business consultants. It will elaborate which added values can be created and look for new forms of accessing, handling and utilising content in the IT and business services industry. Complementary to the other two case studies, it will focus on how to stipulate the emergence and creation of new knowledge and its capture, thereby providing important input for the further shaping of SEKT in subsequent phases.

We will yet have to see how DILIGENT will be applicable for this use case. We imagine the creation of new knowledge being accompanied by related changes in the local ontology of the knowledge worker, and thus allow easier access to her results, a more efficient evaluation by a supervising board and finally a better and more coherent integration of her work into the existing knowledge base.

This follows the DILIGENT process, but in a different, more active manner than the other use cases can, and may thus yield interesting results.

4.4 Support for PROTON Development

An important practical approach to generating a shareable and easier mappable ontology is the use of background knowledge in the form of an upper level ontology. Within the SEKT project, an upper level ontology called "PROTON" ([TKM04]) will be developed. In fact, PROTON can be used in several ways, from metadata to ontology generation.

Engineering an upper level ontology is a difficult task due to numerous constraints and goals, sometimes contradicting. Especially because upper level ontologies usually are domain independent and should be applicable to a big number of tools, numerous questions from epistemological considerations over computational complexity constraints and human understandability to pragmatic tool usage have to be answered.

The development of PROTON is closely following the DILIGENT process, but it should be noted, that the DILIGENT process is actually meant to happen for loosely controlled environments, where the ontologies may adapt locally as they are being used. For PROTON we therefore have to map some of the steps to the different situation. OntoText, the partner responsible for PROTON, initially created and delivered early versions of PROTON in order to foster comments from partners and to create a common ground for application level ontologies (**build** phase). The partners analyse PROTON with respect to their own expertise, be it with regards to practical constraints in the tools that will be used, with regards to the goals in their case studies or with regards to theoretical considerations to ontology engineering. These comments may be regarded as suggestions for **local adaptation**. Usually these comments are sent via the appropriate mailing lists, and then arguments are exchanged in favour or against the suggested changes. This argumentation is analysable with the Rhetorical Structure Theory (see Section 3.5.2). Even though mailing lists may have some disadvantages to other solutions for traceable and later easy accessible discussions, they still do a decent job and have the big advantage of being easy and using a very accepted and widespread technology (i.e. e-Mail).

The board, in our case this role is played by OntoText, after reading and **analysing** the arguments, makes the final decisions about the changes and **revisions** of PROTON. They then create a new release, which the partners download and review anew (this represents the **local update** phase).

It is planned to set up a dedicated community process to handle these kinds of change requests and comments, and thus the evolution of PROTON.

Chapter 5

Conclusions

It is now widely agreed that ontologies are a core enabler for the Semantic Web. The development of ontologies in centralized settings is well studied and established methodologies exist. The knowledge structures underlying today's knowledge management systems constitute a kind of ontology that may be built according to such established methodologies as *e.g.* [SAA^{+}99, Sur03]. These methodologies have a centralized approach towards engineering knowledge structures requiring *knowledge engineers*, *domain experts* and others to perform various tasks such as *requirement analysis* and *interviews*. While the user group of such an ontology may be huge, the development itself is performed by a — comparatively — small group of domain experts who *represent* the user community and ontology engineers who *help structuring*.

Current experiences from projects and the analysis of the evolution of the classification of life forms in Biology suggest that ontology engineering should be considered as a continuous improvement rather than a one time action and that ontologies promise the most benefits in decentralized rather than centralized systems. Hence, a methodology for distributed, loosely-controlled and evolving ontology engineering settings is needed. DILIGENT is a step towards such a methodology. DILIGENT comprises the steps **Build**, **Local Adaptation**, **Analysis**, **Revision** and **Local Update** and introduces a board to supervise changes to a shared core ontology.

Here we presented a fine-grained argumentation framework to be used in evolving distributed environments. We use RST to analyse the arguments exchanged when consensus is sought in evolving distributed ontology engineering processes. We have strong evidence from an *in situ* experiment, that our argumentation framework decisively contributed to speeding up process and to finding a truly shared ontology. This is a particularly important conclusion when one foresees the development of shared ontologies in distributed settings, such as the Semantic Web. The arguments which we identified as most useful in an ontology building process are: **elaboration**, **evaluation/justification**, **alternatives**, **examples** and **counter examples**. Having provided evidence for the applicability of our methodology we will now see how DILIGENT will develop with experience gained with the distributed ontology engineering efforts in the SEKT project, and how the ontology engineering processes themselves will gain from applying DILIGENT.

However, after discussing the case studies with the SEKT partners, we expect the SEKT case studies (i) to be **decentralized**, (ii) to rely on **evolving ontologies** to adapt to changing environments, and (iii) to evolve ontologies in a **loosely controlled manner**. In such decentralized settings working based on traditional, centralized knowledge management systems becomes infeasible. While there are some technical solutions toward Peer-to-Peer knowledge management systems (e.g., [BBMN03]) – and we have, *e.g.*, developed a technically sophisticated solution of our own in the finalized project SWAP [EHvH$^+$03] – traditional methodologies for creating and maintaining knowledge structures appear to become unusable like the systems they had been developed for in the first place. Here we sketch a methodology to support such scenarios, that is flexible enough to deal with the differences in the use cases and still precise enough to help them.

Furthermore, the methodology will integrate specific issues of the three core technologies that are explored within SEKT: Knowledge Discovery, Human Language Technology and Ontologies & Metadata. This initial version will be applied and evaluated within the case studies. The methodology will be extended by capturing lessons learned and best practices. In the end, the methodology will be an illustrated guidebook for implementing and applying the SEKT technology in different settings to facilitate the take-up and transfer of the technology.

Bibliography

[ACFLGP01] 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, B.C., Canada*, 2001.

[Alb93] F. Albrecht. *Strategisches Wissensmanagement der Unternehmensressource Wissen*. Verlag Peter Lang, Frankfurt am Main, 1993.

[AO98] D. Alsmeyer and F. Owston. Collaborating in information space. In *Proceedings of Online Information 98*, pages 31–37, 1998.

[ASvE04] F. Aschoff, F. Schmalhofer, and L. van Elst. Knowledge mediation: A procedure for the cooperative construction of domain ontologies. In A. Abecker, L. van Elst, and V. Dignum, editors, *Proceedings of Workshop on Agent-Mediated Knowledge Management at the 16th European Conference on Artificial Intelligence (ECAI'2004)*, pages 20–28, Valencia, Spain, August, 22-27 2004.

[BBMN03] M. Bonifacio, P. Bouquet, G. Mameli, and M. Nori. Peer-mediated distributed knowldege management. In van Elst et al. [vEDA03].

[BCC$^+$05] V. R. Benjamins, P. Casanovas, J. Contreras, J. M. Lopez-Cobo, and L. Lemus. *Iuriservice: An Intelligent Frequently Asked Questions System to Assist Newly Appointed Judges*. 2005. to appear.

[BEH$^+$02] E. Bozsak, M. Ehrig, S. Handschuh, A. Hotho, A. Maedche, B. Motik, D. Oberle, C. Schmitz, S. Staab, L. Stojanovic, N. Stojanovic, R. Studer, G. Stumme, Y. Sure, J. Tane, R. Volz, and V. Zacharias. KAON – towards a large scale semantic web. In K. Bauknecht, A. M. Tjoa, and G. Quirchmayr, editors, *Proceedings of the Third International Conference on E-Commerce and Web Technologies (EC-Web 2002)*, volume 2455 of *LNCS*, pages 304–313, Aix-en-Provence, France, 2002. Springer.

[BG04] J. Blythe and Y. Gil. Incremental formalization of document annotations through ontology-based paraphrasing. In *Proceedings of the 13th international conference on World Wide Web*, pages 455–461. ACM Press, 2004.

[BHGS01] S. Bechhofer, I. Horrocks, C. Goble, and R. Stevens. OilEd: A reason-able ontology editor for the semantic web. In *KI-2001: Advances in Artificial Intelligence*, LNAI 2174, pages 396–408. Springer, 2001.

[BLC96] A. Bernaras, I. Laresgoiti, and J. Corera. Building and reusing ontologies for electrical network applications. In *Proceedings of the European Conference on Artificial Intelligence (ECAI'96)*, 1996.

[BLHL01] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 2001(5), 2001. available at http://www.sciam.com/2001/0501issue/0501berners-lee.html.

[CB88] J. Conklin and M. L. Begeman. gibis: a hypertext tool for exploratory policy discussion. In *Proceedings of the 1988 ACM conference on Computer-supported cooperative work*, pages 140–152. ACM Press, 1988.

[CKC+99] P. Chapman, R. Kerber, J. Clinton, T. Khabaza, T. Reinartz, and R. Wirth. The CRISP-DM process model. Discussion Paper, March 1999. http://www.crisp-dm.org.

[CSSS01] J. Conklin, A. Selvin, S. Buckingham Shum, and M. Sierhuis. Facilitated hypertext for collective sensemaking: 15 years on from gibis. In *Proceedings of the twelfth ACM conference on Hypertext and Hypermedia*, pages 123–124. ACM Press, 2001.

[Den02] M. Denny. Ontology editor survey results (table), 2002. available at http://xml.com/2002/11/06/Ontology_Editor_Survey.html.

[Den04] M. Denny. Updated ontology editor survey results (table), 2004. available at http://www.xml.com/pub/a/2004/07/14/onto.html.

[DFv02] J. Davies, D. Fensel, and F. van Harmelen, editors. J. Wiley and Sons, 2002.

[DJM02] J. Demey, M. Jarrar, and R. Meersman. A conceptual markup language that supports interoperability between business rules modeling systems. In Meersman et al. [MT+02], pages 19–35.

[dMA03] A. de Moor and M. Aakhus. Argumentation support: From technologies to tools. In *Proc. of the 8th International Working Conference on the Language-Action Perspective on Communication Modelling (LAP 2003)*, Tilburg, The Netherlands, June 1-2 2003.

[Dom98] J. Domingue. Tadzebao and WebOnto: Discussing, browsing, and editing ontologies on the web. In *Proceedings of the 11th Knowledge Acquisition for Knowledge-Based Systems Workshop, April 18th-23rd. Banff, Canada*, 1998.

[DP98] T. H. Davenport and L. Prusak. *Working Knowledge – How organisations manage what they know*. Havard Business School Press, Boston, Massachusetts, 1998.

[Dru93] P. A. Drucker. *A Post Capitalist Society*. HarperCollins, New York, 1993.

[DSW$^+$00] A. J. Duineveld, R. Stoter, M. R. Weiden, B. Kenepa, and V. R. Benjamins. Wondertools? a comparative study of ontological engineering tools. *International Journal of Human-Computer Studies*, 6(52):1111–1133, 2000.

[Eas91] S. Easterbrook. Handling conflict between domain descriptions with computer-supported negotiation. *Knowlege Acquistion*, 3(3):255–289, 1991.

[EHvH$^+$03] M. Ehrig, P. Haase, F. van Harmelen, R. Siebes, S. Staab, H. Stuckenschmidt, R. Studer, and C. Tempich. The SWAP data and metadata model for semantics-based peer-to-peer systems. In *Proceedings of MATES-2003. First German Conference on Multiagent Technologies*, LNAI, Erfurt, Germany, September 22-25 2003. Springer.

[EM97] L. Edvinson and M. Malone. *Intellectual capital. Realizing your company's true value by finding its hidden brainpower*. Harper, New York, 1997.

[Fen01] D. Fensel. *Ontologies: Silver bullet for knowledge management and electronic commerce*. Springer-Verlag, Berlin, 2001.

[FFR96] A. Farquhar, R. Fickas, and J. Rice. The Ontolingua Server: A tool for collaborative ontology construction. In *Proceedings of the 10th Banff Knowledge Acquisition for KnowledgeBased System Workshop (KAW'95)*, Banff, Canada, November 1996.

[FLGPE$^+$02] 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.

[FLGPSS99] 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), January/February 1999.

[GF94] O. Gotel and A. Finkelstein. An analysis of the requirements traceability problem. In *Proceedings of International Conference on Requirements Engineering 1994*, pages 94–101. IEEE CS Press, 1994.

[GF97] O. Gotel and A. Finkelstein. Extended requirements traceability: Results of an industrial case study. In *Proceedings of the 3rd IEEE International Symposium on Requirements Engineering (RE'97)*, page 169. IEEE Computer Society, 1997.

[GK97] T. F. Gordon and N. Karacapilidis. The zeno argumentation framework. In *Proceedings of the sixth international conference on Artificial intelligence and law*, pages 10–18. ACM Press, 1997.

[GP96] A. Gómez-Pérez. A framework to verify knowledge sharing technology. *Expert Systems with Application*, 11(4):519–529, 1996.

[GP04] A. Gómez-Pérez. Ontology evaluation. In S. Staab and R. Studer, editors, *Handbook on Ontologies*, volume 10 of *International Handbooks on Information Systems*, chapter 13, pages 251–274. Springer, 2004.

[GPAFL$^+$02] A. Gómez-Pérez, J. Angele, M. Fernandéz-López, V. Christophides, A. Stutt, Y. Sure, et al. A survey on ontology tools. OntoWeb deliverable 1.3, Universidad Politecnia de Madrid, 2002.

[GPFLC$^+$02] 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.

[GPFLC03] A. Gómez-Pérez, M. Fernández-López, and O. Corcho. *Ontological Engineering*. Advanced Information and Knowlege Processing. Springer, 2003.

[GPS98] A. Gangemi, D.M. Pisanelli, and G. Steve. Ontology integration: Experiences with medical terminologies. In Nicola Guarino, editor, *Formal Ontology in Information Systems*, pages 163–178, Amsterdam, 1998. IOS Press.

[GR02] Y. Gil and V. Ratnakar. Trellis: An interactive tool for capturing information analysis and decision making. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web*, pages 37–42. Springer-Verlag, 2002.

[Gru93] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

[Gru95] 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.

[GSV04] T. Gabel, Y. Sure, and J. Voelker. KAON – ontology management infrastructure. SEKT informal deliverable 3.1.1.a, Institute AIFB, University of Karlsruhe, 2004.

[GW02] N. Guarino and C. Welty. Evaluating ontological decisions with OntoClean. *Communications of the ACM*, 45(2):61–65, February 2002.

[Hal01] T. Halpin. *Information Modelling and Relational Databases: From Conceptual Analysis to Logical Design*. Morgan-Kaufmann, 2001.

[HJ02] C. W. Holsapple and K. D. Joshi. A collaborative approach to ontology design. *Communications of the ACM*, 45(2):42–47, 2002.

[HNM02] C. J. Hou, N. F. Noy, and M. Musen. A template-based approach toward acquisition of logical sentences. In Musen et al. [MNS02], pages 77–89.

[Hol03a] C. W. Holsapple, editor. *Handbook on Knowledge Management 1 – Knowledge Matters*. International Handbooks on Information Systems. Springer, Berlin, Heidelberg, New York, 2003.

[Hol03b] C. W. Holsapple, editor. *Handbook on Knowledge Management 2 – Knowledge Directions*. International Handbooks on Information Systems. Springer, Berlin, Heidelberg, New York, 2003.

[Hor98] I. Horrocks. Using an expressive description logic: FaCT or fiction? In *Proceedings of the International Conference on Knowledge Representation (KR 1998)*, pages 636–649. Morgan Kaufmann, 1998.

[Hun04] A. Hunter. Towards higher impact argumentation. In D. L. McGuinness and G. Ferguson, editors, *AAAI2004*, pages 275–280. AAAI Press / The MIT Press, 2004.

[HV04] P. Haase and J. Voelker. Requirements analysis for usage-driven and data-driven change discovery. SEKT informal deliverable 3.3.1.a, Institute AIFB, University of Karlsruhe, 2004.

[IEE84] IEEE. Ieee guide to software requirements specifications. Technical report, IEEE, 1984. ANSI/IEEE Standard 830-1984.

[Jac96] R. Jacques. *Manufacturing the employee – Management Knowledge from the 19th to 21st Centuries*. SAGE Publications, London, Thousand Oaks, New Delhi, 1996.

[JM02] M. Jarrar and R. Meersman. Formal ontology engineering in the DOGMA approach. In Meersman et al. [MT^{+}02], pages 1238–1254.

[Kay03] A. S. Kay. The curious success of knowledge management. In Holsapple [Hol03b], pages 679–687.

[KR70] W. Kunz and H. W. J. Rittel. Issues as elements of information systems. Working Paper 131, Institute of Urban and Regional Development, University of California, Berkeley, California, 1970.

[KSE03] P. A. Kirschner, S. J. Buckingham Shum, and C. S. Carr (Eds.), editors. *Visualizing Argumentation: Software Tools for Collaborative and Educational Sense-Making*. Springer, London, 2003.

[KV03] K. Kotis and G. Vouros. Human centered ontology management with HCONE. In *ODS'03: Proceedings of the IJCAI-03 Workshop on Ontologies and Distributed Systems*, volume 71. CEUR-WS.org, 2003.

[KVA04] K. Kotis, G. A. Vouros, and Jerónimo Padilla Alonso. HCOME: tool-supported methodology for collaboratively devising living ontologies. In *SWDB'04: Second International Workshop on Semantic Web and Databases 29-30 August 2004 Co-located with VLDB*. Springer-Verlag, 2004. to appear.

[LAB$^+$02] 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.

[LBB$^+$02] 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.

[LG90] D. B. Lenat and R. V. Guha. *Building large knowledge-based systems. Representation and inference in the CYC project*. Addison-Wesley, Reading, Massachusetts, 1990.

[LUM$^+$02] G. Li, V. Uren, E. Motta, S. Buckingham Shum, and J. Domingue. Claimaker: Weaving a semantic web of research papers. In *Proceedings of the First International Semantic Web Conference on The Semantic Web*, pages 436–441. Springer-Verlag, 2002.

[Mae02] A. Maedche. *Ontology Learning for the Semantic Web*. Kluwer Academics, February 2002.

[Mar97] D. Marcu. The rhetorical parsing of natural language texts. In *The Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics, (ACL'97/EACL'97)*, pages 96–103, Madrid, Spain, July 7-10 1997.

[MFRW00] D. L. McGuinness, R. Fikes, J. Rice, and S. Wilder. An environment for merging and testing large ontologies. In *Proceedings of the International Conference on Knowledge Representation (KR 2000)*, pages 483–493. Morgan Kaufmann, 2000.

[MMV02] B. Motik, A. Maedche, and R. Volz. A conceptual modeling approach for semantics–driven enterprise applications. In Meersman et al. [MT$^+$02], pages 1082–1099.

[MNS02] M. Musen, B. Neumann, and R. Studer, editors. *Intelligent Information Processing*. Kluwer Academic Publishers, Boston, Dordrecht, London, 2002.

[MT87] W. C. Mann and S. A. Thompson. Rhetorical structure theory: A theory of text organization. In L. Polanyi, editor, *The Structure of Discourse*. Ablex Publishing Corporation, Norwood, N.J., 1987.

[MT$^+$02] R. Meersman, Z. Tari, et al., editors. *Proceedings of the Confederated International Conferences: On the Move to Meaningful Internet Systems (CoopIS, DOA, and ODBASE 2002)*, volume 2519 of *Lecture Notes in Computer Science (LNCS)*, University of California, Irvine, USA, 2002. Springer.

[NFM00] N. Noy, R. Fergerson, and M. Musen. The knowledge model of Protégé-2000: Combining interoperability and flexibility. In R. Dieng and O. Corby, editors, *Proceedings of the 12th International Conference on Knowledge Engineering and Knowledge Management: Methods, Models, and Tools (EKAW 2000)*, volume 1937 of *Lecture Notes in Artificial Intelligence (LNAI)*, pages 17–32, Juan-les-Pins, France, 2000. Springer.

[PB88] C. Potts and G. Bruns. Recording the reasons for design decisions. In *Proceedings of the 10th international conference on Software engineering*, pages 418–427. IEEE Computer Society Press, 1988.

[PM01] H. S. Pinto and J. P. Martins. A methodology for ontology integration. In *Proceedings of the First International Conference on Knowledge Capture (K-CAP2001)*, pages 131–138, New York, 2001. ACM Press.

[PSST04] S. Pinto, S. Staab, Y. Sure, and C. Tempich. OntoEdit empowering SWAP: a case study in supporting DIstributed, Loosely-controlled and evolvInG Engineering of oNTologies (DILIGENT). In C. Bussler, J. Davies, D. Fensel, and R. Studer, editors, *First European Semantic Web Symposium, ESWS 2004*, volume 3053 of *LNCS*, pages 16–30, Heraklion, Crete, Greece, May 2004. Springer.

[Qui92] J. Quinn. *Intelligent Enterprise. A knowledge and service based paradigm for industry*. Free Press, New York, 1992.

[RA+03] U. Reimer, A. Abecker, , S. Staab, and G. Stumme, editors. *Proceedings of the 2nd National Conference "Professionelles Wissensmanagement – Erfahrungen und Visionen (WM2003)"*, volume P–28 of *GI-Edition Lecture Notes in Informatics (LNI)*, Luzern, Switzerland, 2003. Gesellschaft fuer Informatik (GI).

[RCCP04] L. Rodrigo, M. Blázquez Cvico, P. Casanovas, and M. Poblet. Legal case study before analysis. SEKT deliverable 10.1.1, Intelligent Software Components S.A. and Universitat Autonoma de Barcelona, 2004.

[SA02] Y. Sure and J. Angele, editors. *Proceedings of the First International Workshop on Evaluation of Ontology based Tools (EON 2002)*, volume 62 of *CEUR Workshop Proceedings*, Siguenza, Spain, 2002. CEUR-WS Publication, available at http://CEUR-WS.org/Vol-62/.

[SAA+99] 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*. The MIT Press, Cambridge, Massachusetts; London, England, 1999.

[SAS03] Y. Sure, J. Angele, and S. Staab. OntoEdit: Multifaceted inferencing for ontology engineering. *Journal on Data Semantics*, LNCS(2800):128–152, 2003.

[Sch96a] U. Schneider. Management in der wissensbasierten unternehmung. In *Wissensmanagement* [Sch96b], pages 13–48.

[Sch96b] U. Schneider, editor. *Wissensmanagement*. Frankfurter Allgemeneine Zeitung, Frankfurt am Main, 1996.

[SG89] M.L.G. Shaw and B.R. Gaines. Comparing conceptual structures: Consensus, conflict, correspondence and contrast. *Knowledge Acquisition*, 1(4):341–363, 1989.

[SK93] Y. Sakamoto and E. Kuwana. Toward integrated support of synchronous and asynchronous communication in cooperative work: an empirical study of real group communication. In *Proceedings of the conference on Organizational computing systems*, pages 90–97. ACM Press, 1993.

[SMD02] S. Buckingham Shum, E. Motta, and J. Domingue. Augmenting design deliberation with compendium: The case of collaborative ontology design. In *HypACoM 2002: Facilitating Hypertext-Augmented Collaborative Modeling. ACM Hypertext'02 Workshop*, University Maryland, MD,

2002. Retrieved November 24, 2004 from `http://kmi.open.ac.uk/projects/compendium/SBS-HT02-Compendium.html`.

[SMJ02] P. Spyns, R. Meersman, and M. Jarrar. Data modelling versus ontology engineering. *SIGMOD Record – Web Edition*, 31(4), December '02 2002. Special Section on Semantic Web and Data Management; R. Meersman and A. Sheth (eds.); Available at http://www.acm.org/sigmod/record/.

[SPKR96] B. Swartout, R. Patil, K. Knight, and T. Russ. Toward distributed use of large-scale ontologies. In *Proceedings of the 10th Knowledge Acquisition Workshop (KAW'96)*, Banff, Canada, November 1996.

[SRKR97] B. Swartout, P. Ramesh, K. Knight, and T. Russ. Toward distributed use of largescale ontologies. In *Symposium on Ontological Engineering of AAAI*, Stanford, CA., 1997.

[SS02] Y. Sure and R. Studer. On-To-Knowledge methodology. In Davies et al. [DFv02], chapter 3, pages 33–46.

[SS03] Y. Sure and H.-P. Schnurr, editors. *Proceedings of the 1st National "Workshop Ontologie-basiertes Wissensmanagement (WOW2003)"*, 2003. April 2003, Luzern, Switzerland; held in conjunction with [RA$^+$03].

[SSS$^+$01] A. Selvin, S. Buckingham Shum, M. Sierhuis, J. Conklin, B. Zimmermann, C. Palus, W. Drath, D. Horth, J. Domingue, E. Motta, and G. Li. Compendium: Making meetings into knowledge events. In *Knowledge Technologies*, Austin, TX, March 4-7 2001.

[SSSS01] 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, January/Febrary 2001.

[Ste97] T. A. Stewart. *Intellectual Capital – The New Wealth of Organizations*. Doubleday/Currency, a division of Bantam Doubleday Dell Publishing Group, Inc., 1997.

[Sur03] Y. Sure. *Methodology, Tools and Case Studies for Ontology based Knowledge Management*. PhD thesis, University of Karlsruhe, 2003.

[TKM04] I. Terziev, A. Kiryakov, and D. Manov. Base upper-level ontology (bulo) guidance. SEKT deliverable 1.8.1, Ontotext Lab, Sirma AI EAD (Ltd.), 2004.

[TPSS04] C. Tempich, S. Pinto, S. Staab, and Y. Sure. A case study in supporting DIstributed, Loosely-controlled and evolvInG Engineering of oNTologies (DILIGENT). In K. Tochtermann and H. Maurer, editors, *Proceedings of the 4th International Conference on Knowledge Management (I-*

KNOW'04), pages 225–232, Graz, Austria, June 30 – July 02 2004. Journal of Universal Computer Science (J.UCS).

[TRJ84] S. Toulmin, R. Rieke, and A. Janik. *An introduction to reasoning*. Macmillan Publishing, 1984.

[TS98] J. Tennison and N. Shadbolt. APECKS: A tool to support living ontologies. In *Proceedings of the 11th Knowledge Acquisition Workshop (KAW'98)*, Banff, Canada, April 1998.

[TV03] C. Tempich and R. Volz. Towards a benchmark for semantic web reasoners - an analysis of the DAML ontology library. In York Sure, editor, *Evaluation of Ontology-based Tools (EON2003) at Second International Semantic Web Conference (ISWC 2003)*, October 2003.

[UG96] M. Uschold and M. Grueninger. Ontologies: Principles, methods and applications. *Knowledge Sharing and Review*, 11(2), June 1996.

[UK95] 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.

[UKMZ98] M. Uschold, M. King, S. Moralee, and Y. Zorgios. The enterprise ontology. *Knowledge Engineering Review*, 13(1):31–89, 1998.

[vEDA03] L. van Elst, V. Dignum, and A. Abecker, editors. *"Agent-Mediated Knowledge Management International Symposium AMKM 2003" Stanford, CA, USA*. Lecture Notes in Artificial Intelligence (LNAI) 2926. Springer, Berlin, 2003.