DILIGENT: Towards a fine-grained methodology for DIstributed, Loosely-controlled and evolvInG Engineering of oNTologies

H.Sofia Pinto[1] and **Steffen Staab**[2] and **Christoph Tempich**[2]

Abstract.

Ontology engineering processes in truly distributed settings like the Semantic Web or global peer-to-peer systems may not be adequately supported by conventional, centralized ontology engineering methodologies. In this paper, we present our work towards the DILIGENT methodology, which is intended to support *domain experts* in a *distributed* setting to *engineer and evolve* ontologies with the help of a fine-grained methodological approach based on Rhetorical Structure Theory, viz. the DILIGENT model of ontology engineering by argumentation.

1 INTRODUCTION AND MOTIVATION

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. [12] 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 [3, 13, 15] 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:

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 the ontologies being developed in these cases will hardly ever gather in one place. Yet they have an interest to fruitfully contribute 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 comprehensively cope 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. [11]).

In this paper, we focus on the last consideration (cf. item 4): Could ontology engineering benefit from a more fine-grained methodological support? To answer this question, we develop a methodological hypothesis from the retrospective analysis of a historical ontology engineering task, viz. the engineering and evolution of the classification of life forms in biology. Specifically, we investigate whether some argumentation structures dominate the progress in the ontology engineering task and should therefore be accounted for in a fine-grained methodology. Then, we test the methodological hypothesis by an *in vivo* experiment of collaborative ontology engineering — once with and once without fine-grained methodological guidance.

In the remainder of this paper, we start by explaining the methodological framework of DILIGENT (Section 2). Then, we present Rhetorical Structure Theory (RST), which constitutes the theoretical underpinning of our fine-grained methodology (Section 3). In Section 4, we go through the different cases/experiments that help to shape and test DILIGENT. Before concluding, we compare to some related work not contrasted in the introduction here.

2 DILIGENT PROCESS

In order to provide enough background knowledge about the DILIGENT argumentation model, which we present in this paper, we here sketch the overall framework, in which it is embedded, i.e. the overall DILIGENT process (cf. [11]).

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,

[1] Instituto Superior Tecnico, Universidade Tecnica de Lisboa, Portugal email: sofia.pinto@dei.ist.utl.pt

[2] AIFB, University of Karlsruhe, Germany email: {staab,tempich}@aifb.uni-karlsruhe.de

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.

Process We will now describe the general process, roles and functions in the DILIGENT process. It comprises five main activities: (1) **build**, (2) **local adaptation**, (3) **analysis**, (4) **revision**, (5) **local update** (*cf.* figure 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 [2, 3, 10, 15] 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. 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.

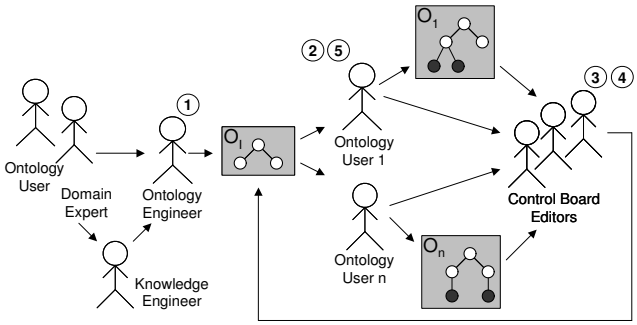

Figure 1. Roles and functions in distributed ontology engineering

The board **analyzes** 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,[3] 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[4] has to 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.

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.

We have applied this process model to the case of folder sharing via a Peer-to-Peer setting with centralized core folder structures and individual specific folder structures. Our experiences there have substantiated the validity of DILIGENT(cf. [11]).

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" [4]. 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 **analyzes** 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 can be used to classify the kinds of arguments most often used and identify the most effective ones.

3 RHETORICAL STRUCTURE THEORY

The aim of Rhetorical Structure Theory (RST) [8] 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.

In the examples provided within the case study section we will highlight the different elements 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.

[3] The idea in this kind of development is not to merge all user ontologies.
[4] This is actually one of the trends in modern software engineering methodologies (see Rational Unified Process).

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 analyzes 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 full NL-understanding.

4 CASE STUDIES

In this section we report how the DILIGENT argumentation model has been developed. We have found in the Biology area a taxonomy that has been evolving for over 200 years, following a DILIGENT-like approach. 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 vivo* 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.

4.1 Case study from the Biology domain

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/**build** 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.[5] Another issue is naming. Lineaus binomial system (genus and species) is still in use, because it can univocally identify a given being in the taxonomy.[6] 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**. 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**.

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. 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 analyzing the arguments exchanged by taxonomists to change the names and organization of the taxonomy one can perceive its vast array and complexity.[7]

> ...*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 analyze arguments exchanged in ontology building process in distributed environments.

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.

4.2 Case study in a computer science department

In order to substantiate our hypothesis, that an appropriate argumentation framework can facilitate the ontology engineering process, we pursued an experiment in a computer science department. 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 use a subset of arguments, and were given stricter rules, 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

[5] We do not provide more examples due to the lack of space
[6] One can reuse names in different kingdoms.
[7] Example from `http://www.ncbi.nlm.nih.gov/Taxonomy/taxonomyhome.html/index.cgi?chapter=CHANGETOCLASS`

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.

4.2.1 First experiment

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

SETTING The participants met in a virtual chat room. 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.

EXAMPLE [8] An excerpt from the real dialogues taking place:

...

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**

...

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, seldom a decision was reached. The moderator interacted only rarely in the discussion. As a result, a core ontology with two concepts, Role and Topic, was agreed upon.

CONCLUSION We analyzed the discussion with the help of RST. Table 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.

4.2.2 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.

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

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 1. Arguments used and outcome

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.

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. ...

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.

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. As to the use of the RST to analyze 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.

5 RELATED WORK

Collaborative ontology engineering has been examined in recent years from different perspectives. We see three research areas as related to our work. The first deals with ontology engineering in general, the second is the work done on methodologies for collaborative ontology engineering, and the third deals with remote collaboration. We have outlined the differences between the general ontology engineering methodologies and our approach already in sections 1 and 2 and do not refer to them here. In [5] 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.In contrast to their work, we support online discussion for a distributed group, thus the participants can interact. Moreover, our process deals with evolving ontologies rather than static ones. [14] describes a system to collect feedback on different ontological discission. However, the feedback is not analysed in the structured way as we do.

There are a number of technical solutions to tackle problems of remote collaboration, *e.g.* ontology editing with mutual exclusion [1], inconsistency detection with a voting mechanism [9] or evolution of ontologies by different means [7]. All these solutions address the issue of keeping an ontology consistent. Obviously, none supports (and do not intend to) the work process of the ontology engineers by way of a methodology. Our process could also benefit from the incorporation of appropriate argument visualization (cf. [6]).

6 CONCLUSION

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. However, current experiences from projects and the analysis of the evolution of the classification of life forms in Biology, suggest that ontology engineering should be subject to continuous improvement rather than a one time action. Hence, a methodology for distributed, loosely-controlled and evolving ontology engineering settings is needed. In this paper we present a process model and a fine-grained methodological approach driven by argumentation for evolving distributed environments. We use RST to analyze the arguments exchanged when consensus is sought in evolving distributed ontology engineering processes. We have strong evidence from an *in vivo* experiment, that our argumentation framework decisively contributed for speeding the process and for 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 it would be interesting to assess if other distributed ontology engineering efforts like the IEEE SUO will benefit from our findings.

Acknowledgements. Research reported in this paper has been partially financed by EU in the IST project SWAP (IST-2001-34103) and Fundação Calouste Gulbenkian (21-63057-B). In particular we want to thank participants of our experiments for the fruitful discussions. We would like to thank the referees for their comments which helped improve this paper.

REFERENCES

[1] A. Farquhar et al., 'The ontolingua server: A tool for collaborative ontology construction', Technical report KSL 96-26, Stanford, (1996).
[2] A. Gangemi et al., 'Ontology integration: Experiences with medical terminologies', in *Formal Ontology in Information Systems*, ed., Nicola Guarino, pp. 163–178, Amsterdam, (1998). IOS Press.
[3] A. Gómez-Pérez et al., *Ontological Engineering*, Advanced Information and Knowlege Processing, Springer, 2003.
[4] T. R. Gruber, 'Towards Principles for the Design of Ontologies Used for Knowledge Sharing', in *Formal Ontol. in Conc. Analysis and Knowl. Rep.*, eds., N. Guarino and R. Poli. Kluwer Acad. Pub., (1993).
[5] C. W. Holsapple and K. D. Joshi, 'A collaborative approach to ontology design', *Commun. ACM*, **45**(2), 42–47, (2002).
[6] *Visualizing Argumentation: Software Tools for Collaborative and Educational Sense-Making*, eds., P. A. Kirschner et al., Springer, 2003.
[7] A. Maedche et al., 'Managing multiple and distributed ontologies on the semantic web', *The VLDB Journal*, **12**(4), 286–302, (Nov 2003).
[8] W. C. Mann and S. A. Thompson, 'Rhetorical structure theory: A theory of text organization', in *The Structure of Discourse*, ed., L. Polanyi, Ablex Pub. Corp., Norwood, N.J., (1987).
[9] A. Pease and J. Li, 'Agent-mediated knowledge engineering collaboration', in *"Agent-Mediated Knowledge Management" International Symposium AMKM 2003*, eds., L. van Elst et al., Springer, (2003).
[10] H. S. Pinto and J.P. Martins, 'A Methodology for Ontology Integration', in *K-CAP2001*, pp. 131–138, New York, (2001). ACM Press.
[11] S. Pinto et al., 'OntoEdit empowering SWAP: a case study in supporting DIstributed, Loosely-controlled and evolvInG Engineering of oNTologies (DILIGENT)', in *1st ESWS 2004*. Springer, (2004).
[12] G. Schreiber et al., *Knowledge Engineering and Management — The CommonKADS Methodology*, The MIT Press, Cambridge, Massachusetts; London, England, 1999.
[13] S. Staab et al., 'Knowledge processes and ontologies', *IEEE Intelligent Systems*, **16**(1), (2001).
[14] J. Tennison and N. R. Shadbolt, 'APECKS: a Tool to Support Living Ontologies', in *11th KAW*, eds., BR Gaines and MA Musen, (1998).
[15] M. Uschold and M. King, 'Towards a methodology for building ontologies', in *Proc. of IJCAI95 WS*, Montreal, Canada, (1995).