D7.1.1.a Use Cases – Initial Version

Marc Ehrig, Thomas Gabel, Peter Haase, York Sure, Christoph Tempich, and Johanna Voelker (Institute AIFB, University of Karlsruhe)

Abstract.
EU-IST Integrated Project (IP) IST-2003-506826 SEKT
Deliverable D7.1.1.a (WP7.1)
This informal deliverable aims to provide use cases for SEKT. The use cases are described in natural language and serve as a first input for detailed discussion with other partners. In particular, we want to clarify the interactions of technical work packages and to understand better the synergies to be expected from combining the three core technologies of SEKT. Our focus is not on the end-user perspective, but rather on the modelling expert perspective. In future refinements it is foreseen that the most relevant use cases will be refined and serve as a basis for joint system implementations of SEKT partners.
Keyword list: use cases

Document Id.	SEKT/2004/D7.1.1.a/v0.6
Project	SEKT EU-IST-2003-506826
Date	June 24, 2004
Distribution	informal deliverable, project internal

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
Contactperson: 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
Contactperson: 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
Contactperson: Hamish Cunningham
E-mail: hamish@dcs.shef.ac.uk

Intelligent Software Components S.A.
Francisca Delgado, 11 - 2
28108 Alcobendas
Madrid
Spain
Tel: +34 913 349 797, Fax: +49 34 913 349 799
Contactperson: Richard Benjamins
E-mail: rbenjamins@isoco.com

Ontoprise GmbH
Amalienbadstr. 36
76227 Karlsruhe
Germany
Tel: +49 721 50980912, Fax: +49 721 50980911
Contactperson: 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
Contactperson: 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
Contactperson: 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
Contactperson: 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
Contactperson: 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
Contactperson: Tom Bösser
E-mail: tb@keapro.net

Sirma AI EOOD (Ltd.)
135 Tsarigradsko Shose
Sofia 1784
Bulgaria
Tel: +359 2 9768, Fax: +359 2 9768 311
Contactperson: 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
Contactperson: Pompeu Casanovas Romeu
E-mail: pompeu.casanovasquab.es

Executive Summary

This informal deliverable aims to provide use cases for SEKT. The use cases are described in natural language and serve as a first input for detailed discussion with other partners. In particular, we want to clarify the interactions of technical work packages and to understand better the synergies to be expected from combining the three core technologies of SEKT. Our focus is not on the end-user perspective, but rather on the modelling expert perspective. In future refinements it is foreseen that the most relevant use cases will be refined and serve as a basis for joint system implementations of SEKT partners.

All use cases are structured as follows:

- Description
- Required Methods and Technologies
- Application Scenarios
- Expected Benefits
- Potential Pitfalls
- Potential Assignments to SEKT Tasks

Contents

Chapter 1

Introduction

This informal deliverable aims to provide use cases for SEKT. The use cases are described in natural language and serve as a first input for detailed discussion with other partners. In particular, we want to clarify the interactions of technical work packages and to understand better the synergies to be expected from combining the three core technologies of SEKT. Our focus is not on the end-user perspective, but rather on the modelling expert perspective. In future refinements it is foreseen that the most relevant use cases will be refined and serve as a basis for joint system implementations of SEKT partners.

All use cases are structured as follows:

- **Description** – a general description of the use case in natural language, potentially accompanied by figures in UML like notation (but not required)
- **Required Methods and Technologies** – a description of all required methods and technologies which should help to understand the research to be performed, the input needed and the interactions with other partners
- **Application Scenarios** – an initial idea of how and where to apply technologies developed as a result of the identified use case
- **Expected Benefits** – a description of the expected benefits of the to be developed technology implied by the use case
- **Potential Pitfalls** – a description of the potential pitfalls which might occur during research and development potentially including in a later stage a more detailed SWOT analysis
- **Assignment to SEKT Tasks** – a list of SEKT tasks relevant for resp. covered by the use case

Chapter 2

Collection of UKARL Use Cases

2.1 Ontology Learning from Text and Semi-Structured Documents

Description

Automatic or semi-automatic learning of ontologies supports the complex process of ontology engineering, allowing even less experienced users to create huge ontologies within a relatively short period of time. The task of learning ontologies from a given corpus can be devided into three main subtasks:

- Extraction and association of concepts and instances
- Learning of taxonomic relations
- Learning of conceptual relations

Although each of these subtasks could potentially be performed in a full-automatic way, especially the results of relation learning algorithms have been proven to be significantly better, if the user is given the possibility to accept or decline single results. In fact user interaction - and how to realize it by means of visualization techniques - is of crucial importance for the whole process of ontology learning. Therefore, not only at the end, but already at the very beginning, the user should be allowed to interfere in order to give a certain direction to the learning process. For example a component for selective learning might ask him to specify interesting concepts, relation types or structural constraints on the resulting ontology. Further details regarding this aspect of user interaction and the implementation of the different subtasks are included in the following subsection.

Required Methods and Technologies

TextToOnto [MV01] is an existing open source tool suite which aims at supporting the ontology engineering process by text mining techniques. Potentially this forms a basis for further developments in the SEKT context. In particular the components provided by TextToOnto for each of the above-mentioned subtasks could be modified in order to build upon GATE [CMBT02], a framework for language engineering developed by the University of Sheffield. A detailed description of the methods already used and the advantages GATE (in particular ANNIE and JAPE) might have in that context is given below.

Learning Tasks

Learning and Association of Concepts and Instances The extraction of concepts and instances from a given text document is supported by GATE (ANNIE) in various ways. Whereas instances corresponding to named entities can be automatically identified by the gazetteer, concepts simply corresponding to nouns will be found by the Part-of-Speech tagger. The association of an instance with a certain concept is currently done by TextToOnto using patterns (e.g. ”the [instance of hotel] hotel”), which could be specified by JAPE in future implementations. Furthermore GATE’s co-reference detection might help to identify even those associations, which are not matched by any pattern yet (e.g. ”the Hilton”).

Learning of Taxonomic Relations In order to learn taxonomic relations TextToOnto uses FCA [CST03] and a combination of WordNet [Fel98], various heuristics and lexico-syntactic patterns defined by M.A. Hearst [Hea92]. The implementation of the latter is still based on Java’s support for regular expressions, but it could easily be done using JAPE, which would allow for storing the rule descriptions in a separate JAPE grammar. Unlike hard-coded rules JAPE grammars do not have to be compiled and therefore are much easier to maintain and re-use.

Learning of Conceptual Relations The extraction of conceptual relations certainly is one of the most difficult tasks related to ontology learning. In order to identify strongly related concepts and instances the current implementation of TextToOnto basically employs two approaches. The first one applies statistical data mining techniques based on association rules described by [MS00]. The second one uses NLP assuming that pairs of concepts or instances being the subject and the object of the same verb are in relationship with each other. Because this relationship is supposed to be characterized by the verb, possible names for the relation to be created can be derived from the verb and suggested to the user. Of course, it is primarily the second approach, which might benefit from GATE and its verb phrase chunking component provided by ANNIE.

Learning from HTML and XML Documents

Because the Semantic Web can be considered as one of the main application scenarios of SEKT, the corpora ontologies are to be learned from will typically contain a large number of HTML and XML documents. Although TextToOnto is already able to deal with both kinds of documents by converting them into plain text files, it does not exploit any HTML/XML tags or links. Therefore an intelligent processing of semi-structured documents and linked corpora could be a valuable extension to the current implementation. Some very promising steps towards this direction have already been made by a joint initiative of the JSI[1] and the AIFB[2], which aims at extracting F-Logic frames from HTML tables.

Selective Learning

In order to increase the effectiveness and scalability of the learning system, the user should be given the possibility to specify both semantic and structural properties of the ontology to be learned. If he wants to create a domain-specific ontology from a thematically heterogeneous corpus, for instance, the learning system might ask the user to provide a description of the concepts and relations he is interested in. Such a description can be a list of terms as well as a collection of prototype documents or an existing domain ontology to be extended.

Application Scenarios

One possible application scenario for automatic or semi-automatic learning of ontologies could be the creation of ”stream ontologies”, that is ontologies built on the fly from the data coming from a live stream. For example, if a user wants to stay up-to-date with the latest terminology in computer science, he could learn an ontology from a stream of computer news. Because in this scenario ontologies would have to be created on the fly, a fast and scalable learning system with a high degree of automation should be provided.

Expected Benefits

The creation or extension of ontologies from very large corpora is a time consuming and difficult task, especially for less experienced users. Because automatic and semi-automatic learning of ontologies supports the user in both building and maintaining an ontology it shortens and simplifies the complex process of ontology engineering.

[1] http://www.ijs.si/
[2] http://www.aifb.uni-karlsruhe.de/

Potential Pitfalls

Incremental extensions to an existing ontology, which are made in an automatic or semi-automatic way, potentially raise a lot of problems an ontology learning system has to deal with. On the one hand the user must be able to retrace and undo each of the changes, so that ontology versioning is needed. On the other hand adding new concepts, instances or relations to an existing ontology can lead to inconsistencies, which must be avoided by using appropriate evolution strategies. Furthermore there are quite a lot of technical problems and pitfalls arising from the implementation of the different subtasks. For example, the results of relation learning strongly depend on quality and quantity of the instances and concepts already given or learned, whereas the latter depends on the quality and size of the corpus. In general an appropriate way and degree of user interaction seems to be the most promising approach to improving ontology learning results.

Assignment to SEKT Tasks

T.1.9 — Simultaneous Ontologies (building multiple ontologies from a single database)
T.1.10 — Stream Ontologies (automatic creation and maintenance of an ontology from a data stream)
T.3.1 — Incremental Ontology Evolution (evolution strategies for incremental ontology learning)

2.2 Ontology Pruning

Description

A number of use cases (cf. Section 2.1) have dealt with the extraction of ontologies from given data sets. The main objective of the use case at hand can be characterized as belonging to the "opposite" task – *ontology maintenance*. So, the goal of ontology pruning is to tailor a given ontology with respect to the needs and specifics of a particular application. With tailoring we mainly refer to the deletion of concepts or entire sub-trees of the ontology under consideration. Here, the trade-off between "completeness" and "minimality" of the respective ontology will have to be considered: Any ontology ought to be complete with respect to a specific domain, application, or task, but minimal. In general, we see two main application fields for ontology pruning:

- The ontology learnt on the basis of, for example, a given text corpus, and its taxonomy of concepts may be too voluminous and not aligned to the application domain. As a result plenty of concepts may have no or only very few instances. Consequently, several concepts may be not required and the performance of the ontology-

based system may be impaired. However, in that case attention has also to be paid to the question whether certain concepts (without instances) are needed as an abstraction layer.

- Assume, a given ontology O shall be imported into the system, e.g. when intending to use that ontology as the starting point to build up a larger domain-specific ontology D. In that case usually O has to be pruned based on the given domain-specific corpus in order to make it usable as the starting point for constructing the specialized ontology D.

Thus, basically ontology pruning can be considered as the opposite of extracting lexical entries from a given corpus: Those lexical entries that do not occur in the given collection of text documents are supposed to be removed from the ontology.

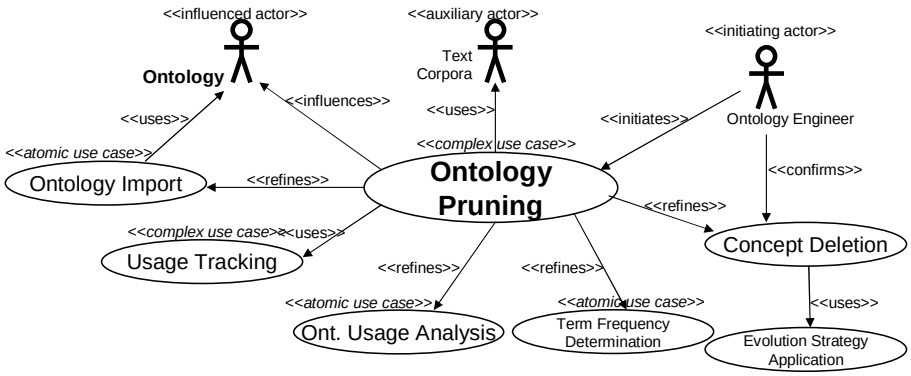

Figure 2.1: Use Case "Ontology Pruning" and Related Use Cases

Required Methods and Technologies

In [Mae02] two different approaches to ontology pruning are discussed. On the one hand, the author examines *baseline pruning* on the basis of the assumption that frequent terms in general lexicalize concepts. Inversely, concepts corresponding to unfrequent terms may be removed from the ontology. On the other hand, *relative pruning* is introduced. Here, the decision whether to eliminate a concept from the ontology or not is made on the basis of a term's frequency in some independent generic reference corpus R. So, a lexical entry is not only considered to be domain-relevant if it occurs very frequently in a given corpus D, but if it occurs significantly more frequent in D than in R.

Within this use case new approaches to ontology pruning shall be examined. For example, instead of considering the lexical term occurrence frequencies only, also the amount of existing relations in the existing ontology may be regarded. A further idea is to take the previous usage of the ontology into consideration. Knowing that certain parts

of the ontology have been visited rather sparsely, one may infer that the corresponding concepts and/or relations are of little significance.

Summarizing, the following methods/technologies will be needed in the scope of this use case:

- appropriate, probably new sophisticated measures that determine the relevance of certain concepts/relations for a specific application domain
- a facility to allow the ontology engineer to interact with the pruning process
- access to usage data and maybe usage patterns in the current ontology in combination with adequate criteria that allow for pruning based on ontology usage

In comparison to ontology management ontology pruning primarily needs to delete entities from an existing ontology. Those changes to the ontology of course require appropriate handling by the ontology management system (middleware). Here, for example, questions have to be tackled that deal with the deletion of concepts and problems such as handling those concepts' instances, relations, and/or subconcepts.

An interesting extension of this use case might arise if it is combined with Ontology Learning 2.1 so that successive learn and prune cycles iteratively create the final ontology.

Application Scenarios

Imagine the Medline corpus, consisting of a large biomedical index with access to content of articles from almost 4,000 journals, and an ontology build upon that large data set []. When aiming at reusing the constructed ontology O (or at least some part of it) as a base-upper level ontology for some other, yet somehow similar medical text data set D, the application of some pruning mechanisms seems to be useful. Then, the resulting ontology for D will feature maximal overlap with O. Moreover, using O as a starting point the ontology generation process may be sped up.

Expected Benefits

The main advantage of a pruned ontology is that it excludes irrelevant information, makes the ontology's use easier, carries information in a more compact manner, and consequently may improve the quality of the ontology-based system significantly. It is plausible that users may find interaction with the system more straightforward if they are not overwhelmed by plenty of information that is only marginally relevant.

To evaluate gained advantages it is necessary to compare the unpruned and pruned version of the ontology not only by syntactic comparison measures, but also regarding their actual use and acceptance by end users. Similarity measures comparing ontologies

and usage patterns might help to facilitate the evaluation. At this point it must be referred to an appropriate evaluation use scenario.

Potential Pitfalls

The crucial problem in ontology pruning – at least when intending to perform it with as few human interaction as possible – is represented by the definition of adequate measures (e.g. term frequency measures for text corpora or ontology usage measures). Finding accurate measures represents an enormous challenge for the system developer. Furthermore, it must be emphasized that the actual choice of those measures is supposed to explicitly bias the pruning process and hence the quality of the resulting ontology.

Assignment to SEKT Tasks

T.1.6 — Ontology Evaluation
T.1.8 — Base-Upper Level Ontology Guidance
T.3.1 — Incremental Ontology Evolution

2.3 Case-Based Ontology Reuse

Description

It is well-known that ontology engineering represents a complex and time-consuming synthesis task. However, the quality of the employed ontology greatly influences the performance of any ontology-based knowledge management system.

The target of this use case is to develop strategies to reuse the expertise that some (human) ontology engineer has put into the development of an existing ontology. If it is possible to recycle at least some fragments of existing ontologies, a great deal of engineering effort may be avoided and thus the ontology development process be sped up.

In [MMS+03] an infrastructure for searching, reusing, and evolving distributed ontologies is described. One of its key components, the Open Registry, provides the functionality to store, search for and browse a set of registered ontologies in order to reuse them. The effort put into the development of the Open Registry may be used as a starting point for this use case.

Required Methods and Technologies

Of course, the problem of reusing existing ontologies has been addressed by many researchers, e.g. [UCH+98, AVGPLTP98, DF01]. A core idea of the use case at hand is to apply some of the techniques known from the research field of Case-Based Reasoning to improve reuse. From an abstract point of view, ontologies and/or ontology fragments may be regarded as cases that are stored in a repository (case base). According to the CBR paradigm [AP94] similar problems have similar solutions. Thus, to be more exact the ontologies stored in the case base represent potential *solutions* that can be reused for upcoming ontology engineering tasks[3].

So, the intention of this use case is to provide facilities to treat ontologies in a case-based manner. In particular this will involve

- the collection of a larger number of ontologies whose reutilization appears to be advantageous (building up of an ontology base),
- the definition of an appropriate case characterization for ontologies,
- the creation of an infrastructure to retrieve entire ontologies for some specific query.

The crucial question, however, is what are the cases' *problem* parts and what are typical queries (i.e. new problems). This issue is in particular striking as the similarity between a query and old cases (during case retrieval) is computed between the new problem and the cases' problem parts.

Imagine, for example, a larger collection of text documents D for which a corresponding ontology shall be generated. Then, this collection represents the new problem from which to define a query and for which to retrieve an adequate old solution. A possible way to formulate a query q for D might be to perform a lexical analysis on D to extract relevant lexical terms that may denote potential concepts. At this, known techniques for ontology learning [Mae02] might be used or further developed. On the other hand, the user may supply a list of terms (referring to concepts and/or relations) that he/she wants to be present in the ontology searched. Anyway, the resulting query will depict an ordered list of terms or concept suggestions (maybe ordered with respect to relevance). Moreover, a query might be enriched by additional meta information about the requested ontology. This approach is pursued by [MMS+03] where desired ontologies are described using an ontology meta ontology.

Employing WordNet As follows, some ideas concerning the employment of WordNet to improve the ontology retrieval are given:

[3]Note that cases usually consist of a problem part and a solution part.

- The current proposal for the formulation of queries plans to specify a query as a vector of terms which ought to be reflected in the ontology to be created. That vector might be constructed (i) manually by the user providing a collection of (eventually weighted) concepts and/or relations, or (ii) automatically as the result of a lexical analysis of a given collection of text documents.

- In WordNet [Fel98] notions are not represented and ordered with respect to their word form, but rather on the level of word *meaning*. To be exact, a word meaning is determined by a collection of synonyms. Consequently, the actual meaning of a specific word in WordNet is determined by its position relative to other words in the WordNet structure.

- The idea of the matching process is to use WordNet as some kind of common denominator: On the one hand, a query as described above may be mapped to WordNet. On the other hand, each ontology contained in the case base may be mapped to WordNet as well, depending on the concepts and relations it includes. On that basis the appropriateness of some ontology O for a particular query q may be determined in terms of similarity.

One of the above-mentioned Open Registry's main drawbacks is its commitment to reuse existing ontology as a whole only (via inclusion). Here, a natural extension would be to relax that constraint and to allow for reusing ontology fragments, also. Then, of course, methods for extracting and reusing subparts of existing ontologies as well as their integration need to be examined.

Application Scenarios

The DAML Ontology Library[4] comprises 282 different ontologies covering various application fields. As a starting point, those might be employed to build up an ontology base from which elements (or parts of) might be retrieved and reused. In a first step, that repository has to be analyzed to determine which of the ontologies are valuable enough to be stored in a ontology base and potentially reused.

Then, given a set of data D, e.g. text documents from a (standard) text corpus, one might determine the most appropriate ontology from the ontology base for D and examine how well it fits the current problem situation. When, in a later stage, extending this reuse scenario to ontology fragments and ontologies composed of fragments from several other ontologies, it should be interesting to compare the effort for building up as well as the quality of the resulting ontology between the case-based and fully manual ontology creation.

[4] `http://www.daml.org/ontologies`

Expected Benefits

Without any doubt, this use case's benefit would be the reuse of taxonomic and non-taxonomic relations when given a set of concept/terms and meta-information circumscribing a desired ontology.

The utilization of existing ontologies is expected to result in a reduction of time needed for ontology development as well as in a higher quality of created ontologies as the responsible engineer can revert to existing ontologies (or parts of those) into whose development already enormous effort had been put and hence indirectly access the expertise contained. The reuse of entire ontologies will in general not lead to optimal results, i.e. usually the knowledge engineer will have to further modify and adapt a retrieved ontology in order to make it suitable for the respective application domain. However, the process of ontology generation may be realized faster and thus cheaper. In particular, a fast start of the ontology engineering process is imaginable as basic design decisions do not have to be made again, but may be reused. This effect can even be boosted when the constraint for reusing entire ontologies is relaxed.

Potential Pitfalls

The availability of existing ontologies is an indispensable prerequisite for this use case. Moreover, it is clear that the main focus for reusing ontologies must be laid upon task- and domain-specific ontologies. Those (existing) ontologies were created for a particular task or for modelling a certain domain of interest. Hence, their reuse makes sense when intended for a similar context, i.e. a similar task or a related domain. This implies, that it must be possible to infer the actual purpose an existing ontology serves for. As already denoted, this can be done by one of the following approaches (or by a combination of both):

- Each ontology contained in the case base may be described by appropriate annotations telling explicitly in which context or for which domain the respective ontology is used.
- The ontology's actual purpose may be inferred from the names of concepts and relations used in the ontology. So, that collection of identifiers serves as the basis to determine the application field of the ontology and hence the usability of reusing that ontology for the query/problem at hand.

As the first approach requires additional knowledge acquisition effort, the second one seems more inviting. Hence, we calculate the degree of overlap between the query's terms and ontological terms taking into regard relationships between words on the basis of WordNet.

The ideas behind the use case at hand feature a number of potential problems that have to be thoroughly examined and solved appropriately:

- availability of a high-quality ontology base, i.e. a set of ontologies whose quality is guaranteed and whose reutilization is advantageous
- comprehensiveness of problem description (Only if it is possible to give a very concise description of the problem at hand (query), it is useful to reuse solutions to former problems. So, the preprocessing of data, i.e. creating a query from the given data is a crucial issue.)
- handling of relations and instances (When reusing an ontology, it is likely that instances must be disregarded, as the appearance of those is highly dependent on the respective application scenario.)

Assignment to SEKT Tasks

T1.5. — Extracting Human Expertise from Existing Ontologies
T1.6. — Ontology Evaluation
T2.5. — Quantitative Evaluation Tools and Corpora
T3.1. — Incremental Ontology Evolution
T3.2. — Usage Tracking for Ontologies and Metadata

2.4 Experience-Based Ontology Creation

Description

As mentioned in Section 2.3 the used ontology's appropriateness for the respective task is of crucial importance for each ontology-based knowledge management system. As the construction of an ontology from scratch represents a complex, time-consuming, and probably error-prone task, the idea of reusing some of the experience, that an ontology engineer has put into the development of an ontology, seems appealing.

Although the use case ontology reuse addresses the issue of finding the most appropriate existing ontology for an upcoming task, it does not take into account the advantages that may arise when only fragments of former ontologies are combined and reused. In other words, just finding an ontology that was used in a similar context before is in many cases not a commensurate condition for its reuse. So, experience-based ontology creation is intended to build on top of the results of use case ontology reuse and aims at its extension into the following directions:

- selecting sub-parts (fragments) of ontologies whose reutilization is highly advisable, while omitting the reuse of ontology parts that are not relevant in the current context
- examining existing (e.g. [PM01]) and developing new strategies to integrate ontological fragments
- comparing the quality and engineering effort of ontologies that were created on the basis of reusing human experiences with ontologies that were build manually or by use of machine-learning approaches

The use case at hand is part of the Karlsruhe technology road map insofar as it builds upon and represents an extension of the case-based ontology reuse – hence, the ideas presented herewith will be targeted subsequently. Accordingly, the ideas presented in the following do not correspond to intended next research steps, but rather to perspectives that we want to investigate.

Required Methods and Technologies

In addition to the methods and technologies required for case-based ontology reuse (Section 2.3), several further problems will have to be tackled.

- It is imaginable to dynamically divide queries as well as cases into smaller units. This would allow for a more fine-grained retrieval, i.e. searching ontology fragments for sub-queries, As a consequence, matchings with higher degrees of similarity are likely to be found.
- Having found a set of ontological fragments, these have to be appropriately integrated into one ontology which is returned to the user and which ought to depict an acceptable solution to the user's query. At this, methods from the research field of ontology aligning and merging may have to be utilized.

  Note, that ontological fragments do not necessarily have to arise from pruning an existing ontology. In fact, also more or less arbitrary connected subgraphs from an ontological structure can be considered as ontological fragments.
- It is of overall importance to empirically verify that recycling human expertise (concealed in an ontological structure) is really beneficial compared to conventional methods of ontology creation. Hence, techniques from the field of ontology evaluation will have to be addressed and used as well.

Application Scenarios

In general the application scenario from Section 2.3 may serve as the basis to apply and evaluate the ideas of experience-based ontology creation.

Expected Benefits

Reusing ontologies usually refers to recycling the entire ontology. Here, an extension to relax that constraint and to allow for reusing ontology fragments is supposed to improve the appropriateness of suggested solutions (i.e. reused and/or combined ontology fragments) for their particular task. Consequently, the effort the ontology engineer has to invest into adapting an existing ontology to his/her task or domain, should be decreased.

Potential Pitfalls

In addition to the potential pitfalls mentioned in the context of the underlying use case case-based ontology reuse, the following problem may arise:

- It may be unfeasible to integrate semantically too different ontological fragments. Moreover, it is unclear according to which strategy relations may be handled that do not fully belong to one fragment.
- The striking question regarding the "quality" of existing ontologies is even more predominant, when only small excerpts of ontologies are considered (which eventually are only meaningful in a broader context, i.e. within their ontology).

Assignment to SEKT Tasks

T1.5. — Extracting Human Expertise from Existing Ontologies
T1.6. — Ontology Evaluation
T2.5. — Quantitative Evaluation Tools and Corpora
T3.1. — Incremental Ontology Evolution
T3.2. — Usage Tracking for Ontologies and Metadata

2.5 Use Case Ontology Usage Mining

Description

The actual usage of an existing ontology may reveal a lot of information about its appropriateness for the respective application. For example, it might be possible to conclude from the user's interaction (e.g. querying or browsing) with the ontology-based knowledge management system, that the underlying ontology's structure is suboptimal. Consequently, certain changes might be applied to that ontology, based on some optimality criteria, in order to improve the overall performance of the system. A lot of work into that

direction, that may be used as a starting point for the use case at hand, has already been done by [SSGS03].

Required Methods and Technologies

First of all, means are required to track and store the actual usage of the system and of the underlying ontology employed. Moreover, it is indispensable to also store changes that are made to the ontology in case the system under consideration employs an evolving ontology.

Second, adequate optimality criteria are needed which allow for an estimation of the quality of the user's interaction with the system. Such measures ought to reflect the users' actual information needs and how those are fulfilled by the system used. When considering the improvement of system usage as an optimization problem, it may be feasible to apply machine learning techniques to optimize the measures mentioned and thus to refine and improve the underlying ontology.

Third, in order to realize these ideas, of course it is necessary to dispose of methods to adapt the system's structure (i.e. the ontology) with respect to optimality of use. Another required technique concerns the analysis of different users and user groups, i.e. the determination user profiles, for which the underlying ontology may be tailored.

Application Scenarios

One possible application scenario for ontology usage mining and subsequent ontology refinement/optimization emerges from a possible analysis of the access to the project web site[5]. As described in [GSV04], it is planned to extend the current web site so that it provides semantically annotated content and is based on an ontological structure. Hence, when storing web site usage data in appropriate log files (click streaming), that collected data may be used to apply the ideas suggested with this use case.

Analogously, further data sources might be employed, e.g. usage data from bibliographic databases such as CiteSeer.

As the collection of a sufficient amount of usage data of course requires a longer period of time, and because the existence of data represents an indispensable and critical prerequisite to start working in the context of this use case, we may also employ usage data collected from the semantic AIFB Portal[6] which is based on an ontology. This data source currently comprises 700.000 semantic log entries that may be utilized to perform first analyses.

[5] `http://sekt.semanticweb.org`
[6] `http://www.aifb.de`

Expected Benefits

This use case's main benefit is represented by the system's ability to implicitly induce useful changes with respect to user behavior, thus enabling the continual improvement of the application. Furthermore, it might be possible to make the system adaptable to certain users or user groups (user profiling).

Potential Pitfalls

The successful implementation of the ideas sketched so far depends on several inalienable prerequisites:

- A sufficient amount of usage data to be analyzed must be available.
- It is important that the data is of high quality, i.e. features as few noise as possible. Hence, appropriate techniques for data pre-processing are necessary.
- The applicability presumes the presence of similar or identical users or user groups who make use of the system repeatedly.

Assignment to SEKT Tasks

T1.6. — Ontology Evaluation
T1.10. — Stream Ontologies
T3.1. — Incremental Ontology Evolution
T3.2 — Usage Tracking for Ontologies and Metadata

2.6 Data-driven Change Discovery

Description

The World Wide Web as one of the most important application scenarios for Ontology Learning is a corpus of semi-structured documents which is not only very big but also extremely dynamic. Whereas its enormous size can be considered as a pure scalability issue, being more-the-less solved by automatic or semi-automatic approaches, the highly dynamic character of the WWW raises even more difficult problems. An ontology which has been extracted from the Web or an intranet with similar characteristics represents a snapshot of the domain knowledge and terminology encoded by the documents at a specific point of time, and it becomes out-dated as soon as the corpus changes. Data-driven Change Discovery tries to identify relevant changes within a specific domain in

order to suggest potential modifications improving the ontology - and consequently all the annotations which are based on this ontology. Some of the most crucial questions arising from this task are

- How to detect changes to a huge document base, such as the WWW
- How to judge the relevance of the changes with respect to the domain knowledge modelled by the ontology
- How to map the changes to potential modifications to the ontology and
- How to present suggestions for these modifications to the user by means of visualization techniques.

Although not all of these questions can be answered without taking into account the regarding application scenario and individual user preferences, the following subsection tries to suggest a few solutions by considering different technical approaches for each problem.

Required Methods and Technologies

An important precondition for Data-driven Change Discovery are immediate notifications about all relevant changes to a certain corpus. Whereas this is not a very challenging task for a small and locally stored corpus where appropriate listeners can be registered to each document, it is quite difficult for huge distributed corpora such as the WWW. Continuous background scanning of a limited number of domains as one possible way deal with this problem would, of course, result in an enormous reduction of the frequency of ontology updates.

Whenever a modification to the corpus is detected the system has to decide whether it involves concepts, instances or relations which are relevant with respect to the domain knowledge modelled by the ontology or specified by the user (see section 2.1). The type of this modification determines the way it is mapped to ontology changes:

- If a new document is added to the corpus or new content is added to a document, new concepts, instances and relations will be extracted in order to add them to the ontology.
- If a document is removed from the corpus or content is removed from a document, all the concepts, instances and relations which are not contained in any other document might be removed as well.
- Since modifying the content of a document can be simulated by removing the original version of a modified sentence and adding the new one, this case can be treated by simply applying the two above-mentioned strategies.

Of course, in all cases existing ontology learning techniques (see chapter 2.1) can be used for the extraction of concepts, instances and relations from the text. But since these machine learning approaches usually need a large document base in order to produce reliable results, their application to single documents or text fragments is rather problematic. Therefore, in order to avoid repeating the ontology extraction process on the basis of the whole corpus, a small subset of *representative* documents might be identified, which - together with the modified documents - provides a sufficient amount of data for ontology learning. These representative documents could be found, for instance, by applying clustering algorithms such as k-Means to the corpus. Only if there are *too many* simultaneous changes to the corpus a repetition of the whole learning process should be considered. In this case the newly created (preliminary) ontology will be compared to the original one by means of appropriate tools, like OntoComparison included in the TextToOnto tool suite [MV01]. The user will then be asked to approve or decline each of the changes.

A graphical visualization of the whole ontology could support him in this decision by providing probability values for the correctness of each modification proposed by the learning system. Furthermore each change should be highlighted and linked with the regarding parts of the corpus.

Application Scenarios

One of the most interesting application scenarios for Data-driven Change Discovery is the detection and visualization of terminological or conceptual changes in a specific domain. For example, if the user has extracted an ontology of various research topics from the CiteSeer repository[7], Data-driven Change Discovery will (semi-)automatically update this ontology every time the common knowledge about a certain topic changes or new topics arise.

Expected Benefits

Data-driven Change Discovery simplifies the time-consuming task of ontology maintenance and it reduces the need for expensive domain experts who spend much time observing a specific domain in order to keep the ontology up-to-date.

Potential Pitfalls

The implementation of Data-driven Change Discovery by applying algorithms developed for ontology learning (as described in subsection 2.6) involves a lot of specific technical problems not yet mentioned in chapter 2.1 such as the need for ontology evolution and versioning.

[7] `http://citeseer.nj.nec.com/cs`

In particular, the extraction of concepts, instances and relations from a partial corpus might require the modification of existing algorithms which consider term weights based on information retrieval measures such as *tf*idf*. Since modifications to the corpus always bring about changes to the term weights, any reduction of the corpus size will result in a lower quality of ontology extraction results, which has to be avoided by a very careful choice of *representative* documents to be considered in addition to those actually changed.

Moreover, since Data-driven Change Discovery for big corpora tends to be a very time-consuming task, the user should not only be given the possibility to define, which changes he considers to be necessary (e.g. which concepts or relations are of current interest), but also which parts of the corpus are to be monitored and how often updates of the ontology have to be performed.

Assignment to SEKT Tasks

T.3.3 — Data-driven Change Discovery

2.7 Ontology Merging

Description

As more and more approaches for ontology creation and learning become common, one can observe an increase of ontologies. To ensure one of the main goals of the Semantic Web, the interoperability, it becomes necessary to align these ontologies. In specific, if they are suppose to work for one application it is necessary to integrate and merge them. From a slightly different perspective one can also interpret the incremental merging of one ontology to another at a time as a kind of evolution of a central ontology.

Required Methods and Technologies

Several methods are required to achieve the alignments or its extension the merging for ontologies.

Ontology Feature Identification The only way to extract additional information such as two entities being identical from ontologies is to interpret their features. We have to identify the specific features which help us identify e.g. similar, subsuming, or redundant entities. These features can be general for any ontology or domain specific.

Heuristic Application for Relation Identification The features have to be evaluated according to some heuristics which returns a quantifiable result. Depending on which

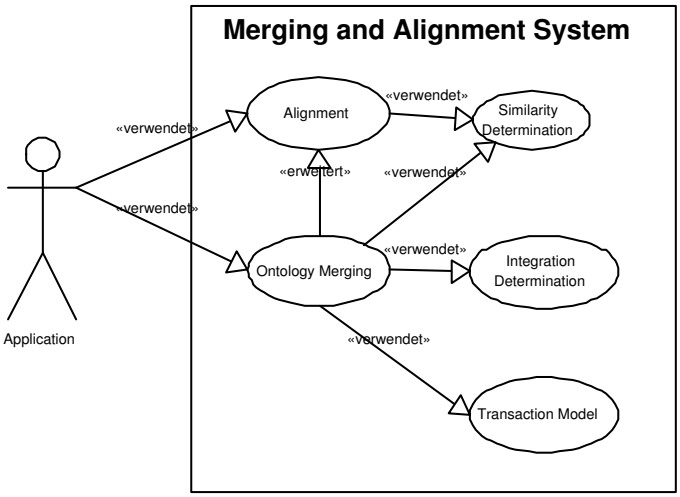

Figure 2.2: Use Case “Ontology Alignment and Merging”

relation one wants to identify one has to use different heuristics e.g. syntactic string similarity. Features combined with heuristics can be described as rules. We cite [**?**] for an overview of possible in this case only similarity identifying approaches.

Interpretation and Decision After a value has been determined a decision has to be taken, on what to interpret from the calculated value. [**?**] present ideas on how determine thresholds for mappings. The decision can also be more complex e.g. on deciding to actually create one new concept from two former concepts.

Action After specific relations have been identified the last step is to actually do the corresponding transaction. In the easiest case only statements are added, in other cases statements are removed - with all its implications. We refer to [SMMS02] for a model on how to deal with different types of changes.

Application Scenarios

Two main application scenarios can be thought of for ontology merging or more specific the use of alignments and merged ontologies.

Use for Structuring One can assume that in many cases one does not have to deal with only one ontology, but several. This is especially true when ontologies are created from different applications, at different times, or on different peers. As the user wants to work with one structure rather than several at the same time, he can rely on the merged ontologies. This structure will then be used for e.g. classification of documents or the knowledge of users in general. This corresponds to the personal knowledge management system as envisioned in SEKT.

Querying The querying scenario is another scenario where aligned ontologies have a major role. Having in mind that several ontologies exist in parallel it is important to have a possibility to query all of them at once. For querying across ontologies one needs to determine alignments. The user will then hopefully receive only meaningful results.

Browsing Somewhat related but an own scenario is an application which allows the user to browse a knowledge base. Again the user will prefer one structure rather than many not interconnected ones.

Expected Benefits

We expect several benefits from aligning or merging ontologies:

Interoperability As mentioned before this is a core idea of the semantic web. Having alignments or even integrated, merged ontologies allows applications to use the different parts of knowledge.

Integrated Complete Ontology An integrated complete ontology allows the user to access an ontology-based knowledge system easier. On the other hand it also allows applications to do more powerful procedures e.g. querying and inferencing over previously distributed and independent knowledge bases.

Potential Pitfalls

Two main pitfalls endanger the usefulness of ontology alignment and merging.

Efficiency A problem which always occurs when trying to determine similarity between a big number of objects is efficiency. For ontology merging many similarity comparisons are required. The danger could be that the actual good determination of possible aligned entities for merging simply takes too long for applications.

Quality of Merged Ontologies Another pitfall is the possible low quality of results. The quality is especially important if automatic postprocessing occurs e.g. further inferencing on the merged ontologies is necessary. A mistake will have effects for all later applications. If quality is too low or applications too sensitive, merging will not be of value.

Assignment to SEKT Tasks

T3.1. — Incremental Ontology Evolution
T4.1. — Ontology Merging
T4.2. — Ontology Alignment

2.8 Evolving Ontology Mappings

Description

In many cases where one has to deal with a multitude of ontologies, ontology mapping is required. The scenarios may be manifold: Just to name one example, in a distributed knowledge management scenario, users may want to maintain their own view on the domain, but want to express queries against other user's ontologies. Ontology mapping has been addressed in many research projects and existing applications (at least prototypical ones) already exist.

An open issue is, how to deal with the the fact that the mapped ontologies change over time, i.e. evolve. Obviously, it is not desired to recreate the mappings between the ontologies from scratch. Instead, it may be useful to analyze the semantics of the change to adapt the mapping using adequate evolution strategies. Depending on the kind of change, it may of course not be possible to adapt the mappings. In these cases, it may be possible to propose changes to the mapped ontology. For example, if a concept is added to one of the ontologies that does not exist yet in the other, the creation of the concept may be proposed.

Required Methods and Technologies

It is assumed that the methods for specifying mappings between ontologies are given. In the most simple case these mappings can simply be the identification of equivalent concepts. Analogously it is assumed that change opererations and the semantics of change for a formal ontology model are defined.

In combination, it is needed to consider:

- *Mapping Formalism:* The mappings between ontologies need to be expressed using an adequate formalism. We assume that this formalism is provided as part of the workpackage Mediation.
- *Semantics of Mapping Evolution:* The semantics of how changes to the ontology affect the mapping need to be formally defined.
- *Compositionality:* Mapping may be defined compositional, i.e. a mapping from ontology A to C may be defined using a mapping A to B and B to C. The evolution of composed mappings may be
- *Versioning and Mappings* When an ontology evolves, it is desirable to be able the keep track of the various versions of the ontology. Here we want to be able to express the relationship between two ontologies as a mapping between ontologies. As for versioning of ontologies, one may want to version mappings between ontologies accordingly.

Application Scenarios

Personal Information Management A typical application scenario may be distributed Personal Information Management (PIM), in which the users maintain their own ontology, but want to query against other ontologies. The users may have specified local mappings against other users ontologies, either manually or automatically. If one user changes his ontology, the mappings may become invalid und thus need to be updated according to the change.

Information Integretion To integrate the schemas of multiple heterogeneous sources, either a global ontology is derived which integrates local schemas, or each information source has its own ontology and the different ontologies are linked directly. Hybrid solutions combine the two approaches by building ontologies of single information sources using elements from a shared vocabulary. Again, mappings are used to express the relationship between the ontologies. The mappings can either be defined manually or automatically by using lexical relations, top-level groundings and semantic correspondences.

One typical application scenario is the integration of bibliographic databases, such as Citeseer, DBLP, Compuscience, etc. The aspects of static integration are already addressed in projects such as SemiPort. A first interesting scenario might be the dynamic integration of these well-established databases with personal sources of bibliographic information.

Expected Benefits

Dealing with mappings of evolving ontologies allows a much more dynamic integration in distributed knowledge management scenarios. Because of the complexity of combined mappings, current distributed systems can only either deal with a single ontology (thus no need for mappings) or static ontologies (thus no need for evolution). Allowing individual, evolving ontologies will be great benefit.

Potential Pitfalls

Ontology mappings and ontology evolution are for themselves already rather complex. The combined complexity is hard to estimate. It will also depend much on the underlying ontology model.

Assignment to SEKT Tasks

T1.9 — Simultaneous Ontologies
T3.1. — Incremental Ontology Evolution

T4.2. — Ontology Alignment

2.9 Ontology Versioning

Description

As ontologies evolve over time, one often needs to keep the current state of the ontology as a version. Versioning can be defined as the ability to manage ontology changes and their effects by creating and maintaining different variants of the ontology. Maintaining variants means that multiple variants of an ontology need to persist, be accessible, etc.

The effects of change include changes on

- the instance data that the ontologies describe,
- other ontologies that are built from or import the ontology, and
- applications that use the ontology

Required Methods and Technologies

Version management requires a set of specific methods:

- *Distinguish and recognize Versions*: this included first of all the ability to uniquely identify a variant of the ontology, e.g. via a versioning scheme. Based on this, operations in the versions (retrieve, undo, etc.) may be defined.
- *Change Detection*: One way to describe the relationship between two versions is de describe a set of changes that transform one version to another. If this set of changes is not provided explicitly, e.g. via an evolution log, the changes need to be detected, e.g. via a diff.
- *Mappings between versions*: A more advanced way to describe the relationship between two ontologies is to specify a mapping from ontology version to another. This mapping can for example be generated from the set of change operations. The mappings can then be used for query rewriting or instance transformations, and thus allow to access a knowledge base through a version of an ontology that is different from the version according to which the knowledge base is organized.
- *Compatibility of versions*: A next task is to determine whether two versions of an ontology are compatible, where the meaning of compatibility may depend on a certain context of use.

- *Multi-version reasoning*: Reasoning as one core task of ontology management becomes a challenge with different, possibly incompatible or inconsistent versions.

Application Scenarios

Versions of Classification Schemes In bibliographic databases such as CiteSeer, the content (knowledge base) is classified according to a classification scheme, for example the ACM topic hierarchy. Over time, the requirements to the classification scheme change and new versions of the classification schemes are created. This results in the need to reclassify parts of the knowledge base. This reclassification can be done fairly easily, if a mapping between the versions is provided. For some changes, e.g. the change from the ACM91 to the ACM98 classification scheme, this is somewhat straightforward, as the core properties of the classification scheme were not changed and only certain types of change operations were performed[8]. For other changes, such as from ACM64 to ACM91, other techniques are required.

In addition, this application scenario requires methods for ontology engineering (changes to the classification scheme by experts, ontology learning (changes in the knowledge base) and usage tracking. Specific required methods are:

- Support for manual changes to the classification scheme: Evolution strategies are important to keep the content base consistent with the changing classification scheme.
- Usage tracking of the classification scheme: A usage model for classification schemes is needed. Based on this usage model, it must be possible to propose changes to the classification scheme.
- Change discovery in the knowledge base: As the content base evolves, e.g. new documents are added, the new knowledge may not be reflected by the classification scheme. Based on these changes, potential extensions to the classification scheme need to be proposed. Similarly, potential areas for reducing the classification scheme may be required.

Expected Benefits

Support for versioning will enable to cope with the changes in specifications and conceptualizations of a domain over time. In addition to evolution, the variants of ontologies are maintained such that it will be possible to access the knowledge via these different variants.

[8] See http://www.acm.org/class/1998/ccs98-intro.html

Potential Pitfalls

The complexity of the ontology model may be a challenge for various aspects of the versioning process: Changes may be hard to detect on the proper level (complex change operations), the compatibility of versions may be hard to define.

Furthermore, there is an additional complexity for the user to deal with versioned ontologies.

Assignment to SEKT Tasks

T3.1 — Incremental Ontology Evolution
T3.4 — Multi-Version Reasoning

Bibliography

[AP94] A. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59, 1994.

[AVGPLTP98] Julio Csar Arprez-Vega, Asuncin Gmez-Prez, Adolfo Lozano-Tello, and Helena Sofia Pinto. (onto)2 agent: An ontology-based www broker to select ontologies. In *Proceedings of ECAI98's Workshop on Application of Ontologies and Problem Solving methods*, pages 16–24, 1998.

[CMBT02] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, Philadelphia, USA, July 2002.

[CST03] P. Cimiano, S.Staab, and J. Tane. Automatic acquisition of taxonomies from text: Fca meets nlp. In *Proceedings of the PKDD/ECML'03 International Workshop on Adaptive Text Extraction and Mining*, 2003.

[DF01] Y. Ding and D. Fensel. Ontology library systems: The key for successful ontology reuse. In *The First Semantic Web Working Symposium (SWWS1)*, Stanford, USA, July 2001.

[Fel98] C. Fellbaum. *WordNet, an electronic lexical database*. MIT Press, 1998.

[GSV04] T. Gabel, Y. Sure, and J. Voelker. D12.1.1. web site and mailing lists. SEKT Deliverable, March 2004.

[Hea92] M.A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*, 1992.

[Mae02] A. Maedche. *Ontology Learning for the Semantic Web*. Kluwer Academic Publishers, Boston, 2002.

[MMS+03] A. Maedche, B. Motik, L. Stojanovic, R. Studer, and R. Volz. An infrastructure for searching, reusing and evolving distributed ontologies. In *Proceedings of the twelfth international conference on World Wide Web*, pages 439–448, Budapest, Hungary, 2003. ACM Press.

[MS00] A. Maedche and S. Staab. Discovering conceptual relations from text. In W. Horn, editor, *Proceedings of the 14th European Conference on Artificial Intellignece (ECAI'2000)*, 2000.

[MV01] A. Maedche and R. Volz. The ontology extraction and maintenance framework text-to-onto. In *Proceedings of the ICDM'01 Workshop on Integrating Data Mining and Knowledge Management*, 2001.

[PM01] H. S. Pinto and Joao P. Martins. Ontology integration: How to perform the process. In A. Gmez-Prez, M. Gruninger, H. Stuckenschmidt, and M. Uschold, editors, *Proceedings of the IJCAI-01 Workshop on Ontologies and Information Sharing*, pages 71–80, Seattle, USA, August 2001.

[SMMS02] L. Stojanovic, A. Maedche, B. Motik, and N. Stojanovic. User-driven ontology evolution management. In *Proceedings of the 13th European Conference on Knowledge Engineering and Knowledge Management EKAW*, volume 2473 of *Lecture Notes in Computer Science*, pages 285 – 300, Siguenza, Spain, October 1-4 2002. Springer.

[SSGS03] L. Stojanovic, N. Stojanovic, J. Gonzalez, and R. Studer. Ontomanager - a system for usage-based ontology management. In *Proceedings of CoopIS/DOA/ODBASE 2003*, Lecture Notes in Computer Science, pages 858–875, Catania, Italy, November 2003. Springer.

[UCH+98] M. Uschold, P. Clark, M. Healy, K. Williamson, and S. Woods. An experiment in ontology reuse. In *Proceedings of the 11th Workshop on Knowledge Acquisition, Modeling, and Management*, Banff, Canada, April 1998.