D2.3.8v1 Report and Prototype of Dynamics in the Ontology Lifecycle

Coordinator: Vít Nováček[1,2]
Siegfried Handschuh[1]**, Diana Maynard**[4]**, Loredana Laera**[3]**,**
Sebastian Ryszard Kruk[1]**, Max Völkel**[5]**, Tudor Groza**[1]**,**
Valentina Tamma[3]

[1]**DERI, National University of Ireland, Galway;** [2]**Faculty of Informatics, Masaryk University, Czech Republic;**

[3]**University of Liverpool, Liverpool, UK;** [4]**The University of Sheffield, Sheffield, UK;** [5]**FZI Forschungszentrum**

Informatik, Karlsruhe, Germany

Abstract.
EU-IST Network of Excellence (NoE) IST-2004-507482 KWEB
Deliverable D2.3.8v1 (WP2.3) proposes the dynamic ontology lifecycle schema and discusses its implementation. It provides concrete solutions to ontology development and evolution issues in highly dynamic and data-intensive environments. Particularly, it deals with proper placement of ontology learning, evaluation and negotiation methods and with integration of learned and collaborative ontologies in a novel way. The transfer possibilities of the framework are justified by elaborated application scenarios from the medicine domain.

Keyword list: ontology, ontology lifecycle, ontology development, dynamic ontology evolution, ontology integration, meaning negotiation, ontology versioning

Document Identifier	KWEB/2007/D2.3.8/v1.0
Project	KWEB EU-IST-2004-507482
Version	v1.1
Date	January 29, 2007
State	final
Distribution	public

Knowledge Web Consortium

University of Innsbruck (UIBK) - Coordinator
Institute of Computer Science
Technikerstrasse 13
A-6020 Innsbruck
Austria
Contact person: Dieter Fensel
E-mail address: dieter.fensel@uibk.ac.at

France Telecom (FT)
4 Rue du Clos Courtel
35512 Cesson Sévigné
France. PO Box 91226
Contact person : Alain Leger
E-mail address: alain.leger@rd.francetelecom.com

Free University of Bozen-Bolzano (FUB)
Piazza Domenicani 3
39100 Bolzano
Italy
Contact person: Enrico Franconi
E-mail address: franconi@inf.unibz.it

**Centre for Research and Technology Hellas /
Informatics and Telematics Institute (ITI-CERTH)**
1st km Thermi - Panorama road
57001 Thermi-Thessaloniki
Greece. Po Box 361
Contact person: Michael G. Strintzis
E-mail address: strintzi@iti.gr

National University of Ireland Galway (NUIG)
National University of Ireland
Science and Technology Building
University Road
Galway
Ireland
Contact person: Christoph Bussler
E-mail address: chris.bussler@deri.ie

École Polytechnique Fédérale de Lausanne (EPFL)
Computer Science Department
Swiss Federal Institute of Technology
IN (Ecublens), CH-1015 Lausanne
Switzerland
Contact person: Boi Faltings
E-mail address: boi.faltings@epfl.ch

Freie Universität Berlin (FU Berlin)
Takustrasse 9
14195 Berlin
Germany
Contact person: Robert Tolksdorf
E-mail address: tolk@inf.fu-berlin.de

**Institut National de Recherche en
Informatique et en Automatique (INRIA)**
ZIRST - 655 avenue de l'Europe -
Montbonnot Saint Martin
38334 Saint-Ismier
France
Contact person: Jérôme Euzenat
E-mail address: Jerome.Euzenat@inrialpes.fr

Learning Lab Lower Saxony (L3S)
Expo Plaza 1
30539 Hannover
Germany
Contact person: Wolfgang Nejdl
E-mail address: nejdl@learninglab.de

The Open University (OU)
Knowledge Media Institute
The Open University
Milton Keynes, MK7 6AA
United Kingdom
Contact person: Enrico Motta
E-mail address: e.motta@open.ac.uk

Universidad Politécnica de Madrid (UPM)
Campus de Montegancedo sn
28660 Boadilla del Monte
Spain
Contact person: Asunción Gómez Pérez
E-mail address: asun@fi.upm.es

University of Liverpool (UniLiv)
Chadwick Building, Peach Street
L697ZF Liverpool
United Kingdom
Contact person: Michael Wooldridge
E-mail address: M.J.Wooldridge@csc.liv.ac.uk

University of Sheffield (USFD)
Regent Court, 211 Portobello street
S14DP Sheffield
United Kingdom
Contact person: Hamish Cunningham
E-mail address: hamish@dcs.shef.ac.uk

Vrije Universiteit Amsterdam (VUA)
De Boelelaan 1081a
1081HV. Amsterdam
The Netherlands
Contact person: Frank van Harmelen
E-mail address: Frank.van.Harmelen@cs.vu.nl

University of Aberdeen (UNIABDN)
Kings College
AB24 3FX Aberdeen
United Kingdom
Contact person: Jeff Pan
E-mail address: jpan@csd.abdn.ac.uk

University of Karlsruhe (UKARL)
Institut für Angewandte Informatik und Formale
Beschreibungsverfahren - AIFB
Universität Karlsruhe
D-76128 Karlsruhe
Germany
Contact person: Rudi Studer
E-mail address: studer@aifb.uni-karlsruhe.de

University of Manchester (UoM)
Room 2.32. Kilburn Building, Department of Computer
Science, University of Manchester, Oxford Road
Manchester, M13 9PL
United Kingdom
Contact person: Carole Goble
E-mail address: carole@cs.man.ac.uk

University of Trento (UniTn)
Via Sommarive 14
38050 Trento
Italy
Contact person: Fausto Giunchiglia
E-mail address: fausto@dit.unitn.it

Vrije Universiteit Brussel (VUB)
Pleinlaan 2, Building G10
1050 Brussels
Belgium
Contact person: Robert Meersman
E-mail address: robert.meersman@vub.ac.be

Work package participants

The following partners have taken an active part in the work leading to the elaboration of this document, even if they might not have directly contributed to writing parts of this document:

National University of Ireland Galway
University of Karlsruhe
University of Liverpool
University of Sheffield
Institut National de Recherche en Informatique et en Automatique

Changes

Version	Date	Author	Changes
0.1	03.11.06	Vít Nováček	draft created
0.2	17.11.06	Diana Maynard	added content to Chapter 2
0.3	19.11.06	Loredana Laera	added content to Chapter 2 and 3
0.4	01.12.06	Vít Nováček	general changes, Chapter 4 completely rewritten, Chapter 5 added, Chapter 7 written
0.5	04.12.06	Loredana Laera	provided corrections for Chapter 2
0.6	05.12.06	Vít Nováček	general changes, added screenshots of the MarcOnt portal to Chapter 2
0.7	06.12.06	Sebastian Kruk	added content to Chapter 2
0.8	12.12.06	Diana Maynard	language corrections, minor changes
0.9	12.12.06	Vít Nováček	figures changed, final changes before QC
1.0	11.01.07	Vít Nováček	QC (Holger Wache) incorporated
1.1	29.01.07	Vít Nováček	QC (Jérome Euzenat) incorporated, final version created (links to D.1.2.4)

Executive Summary

We present an integral scenario of the ontology lifecycle in this report. The phases of ontology evolution (*creation*, *versioning*, *evaluation* and *negotiation*) are introduced. Concrete methods to be employed for the realisation of the particular phases are selected and discussed properly.

Consequently, we propose coherent implementation of the lifecycle and describe a conceptual vision of the dynamics in the scenario that reflects the dynamic flow of knowledge in data-intensive real world applications. We focus on solutions to the following problems that have not been sufficiently covered by the current approaches:

1. specific roles of the evaluation in the different phases of the lifecycle and respective appropriate methods of ontology quality assessment;

2. integration of the results of ontology learning and manual ontology development;

3. proper placement and utilisation of ontology alignment and/or meaning negotiation services within the lifecycle according to the previous two points.

Various methods of evaluation applied in different phases of the lifecycle allow us to assess and improve the ontology quality dynamically within its development and not only at the time of its deployment. The alignment/negotiation techniques are employed not only in mediation of delivered ontologies, but even in the development cycle itself (as a part of a proposed ontology integration process). Since we aim particularly at the technical integration of all the phases, we identify the requirements on the lifecycle implementation in this respect. We have devised and partially implemented a solution to the trickiest part concerning the connection of the phases – a method of semi-automatic learned and collaborative ontology integration. Potential applicability of the whole lifecycle scenario within an industry transfer is demonstrated with respect to suggested use cases from the biomedicine domain.

Contents

Chapter 1

Introduction

Ontologies in real world applications are very likely subject to change, as the domain knowledge is usually changing and/or growing. This holds especially for scientific domains – we have to incorporate the newly discovered facts and possibly change the inappropriate old ones in the respective ontology as the research evolves further. However, even virtually any industrial domain is dynamic – changes typically occur in product portfolios, personal structure or industrial processes, which can all be reflected by an ontology in a knowledge management policy.

Moreover, the data-sets to be utilised in the task of ontology evolution can be very large (e.g. in domains like medicine). It is not always feasible to process all the relevant data and extract the knowledge from them manually, since we need not have a sufficiently large committee of ontology engineers and/or dedicated experts at hand in order to process new data data anytime they occur. This implies a need for (partial) automation of ontology extraction and management processes in a dynamic and data-intensive environments.

Therefore, an ontology lifecycle scenario apt for a universal application in scientific and/or industrial domains should support appropriate mechanisms for dealing with the dynamics in large amounts of knowledge. In this deliverable, we present a simple methodology and coherent framework for practical handling of dynamics and possibly large data-sets in the ontology lifecycle. It offers solutions to the shortcomings of the current ontology development methodologies, which are summed up in the next section of the introduction. The deliverable builds on the relevant achievements that have been deployed within the Knowledge Web NoE (and elsewhere) so far. Two use cases showing the possible straightforward transition of the respective research results to the industry are given, outlining the practical significance of the proposed scenario.

1.1   Motivation

Basically any current generic methodology for ontology evolution takes into account the dynamic nature of the knowledge. A way of dealing with dynamics is usually proposed, typically using a versioning component. Furthermore, the fact that the knowledge cannot be always acquired purely manually due to extensive size of respective resources is acknowledged among the community as well. Thus, means for automated knowledge extraction are usually also included in standard ontology evolution scenarios.

The placement of the version management among other components of ontology lifecycle (such as creation, update, evaluation or alignment) is straightforward – it is naturally bound to the creation and update (extension or retraction) phases of the ontology evolution. However, even though the ontology evolution methodologies usually mention automatic methods of knowledge acquisition, there has been little effort aimed at mechanisms integrating the results of the automatic and manual ontology extraction techniques in a coherent lifecycle scenario.

Another problem is related to the evaluation of the ontologies – in order to continually improve the results of the creation process on the fly, the evaluation should be used as a part of an ontology lifecycle and not only as a measure to assess the quality of an ontology before its final delivery. However, once we deal with manually designed and automatically extracted knowledge within one lifecycle scenario, we have to employ different methods of evaluation of the respective ontologies since they differ in structure and partially also in content. The manually designed ontologies are usually rich in structure (e.g. in the sense of number of syntactic features used) but restricted to a small sub-domain or an abstract level of detail. On the other hand, the automatically extracted ontologies are rather shallow and simple in terms of structure, but specific and large in the sense of coverage of the domain.

Following the motivations above, we propose a very simple methodology in this deliverable. It builds on four basic phases of an ontology lifecycle: *creation* (comprises both manual and automatic ontology development and update approaches), *versioning*, *evaluation* and *negotiation* (comprises ontology alignment and merging as well as negotiation among different possible alignments). The methodology suggests a particular combination of the basic phases into a coherent and dynamic ontology lifecycle scenario. Emphasis is put on the integration of manually designed and automatically extracted (learned) ontologies and proper placement of the respective evaluation measures. An implementation of the scenario is also proposed, using existing tools for the particular phases and newly designed techniques for the integration of learned and manually designed ontologies.

1.2   Position within WP 2.3

This deliverable puts various existing technologies into one coherent and methodologically sound scenario of a dynamic ontology lifecycle. Within WP 2.3, this is related to the versioning methodology and its implementation. The specification of the problem and possible ways of its solution has been solved by the task T2.3.1. The task T2.3.3.3 has dealt with practical implementation of the RDF-based versioning in the scope of the Sem-Version [VG06] system development.

For the manual ontology development, we propose utilisation of a collaborative development framework. This is related to the task T2.3.5, which covers the consensus making environment topic. Application of the negotiation techniques in ontology alignment and merging is an outcome of the task T2.3.7.

1.3   Relation to other Workpackages

As we utilise argumentation-based negotiation and ontology alignment techniques within one of the essential components of our lifecycle, we relate to the research in WP 2.2 (Heterogeneity). Furthermore, we propose concrete scenario use cases from the bio-medicine domain. Therefore we also refer to industrial WP 1.1 – namely to the business cases 2.12 (Hospital Information System) and 2.16 (Integration of Biological Data) presented in [NM04]. This topic is further developed in Section 4.3. Since we inherently aim at implementation of several parts of the Semantic Web framework (as proposed within D1.2.4), our work is related to industry WP 1.2. In Chapter 6, we give an overview of the Semantic Web framework functionalities covered by the work presented here.

1.4   Structure of the Deliverable

The rest of the paper is organised as follows. Firstly, we introduce the overall scheme of the lifecycle and make an overview of the related work in Chapter 2. We continue with presentation of the applicable techniques and methods developed within Knowledge Web WP2.3 or elsewhere, listing and discussing the available techniques for particular phases of the lifecycle. Consequently, we describe the inherent dynamics of the proposed ontology lifecycle and develop the roles of its particular components in Chapter 3. Biomedicine use cases are discussed in Chapter 4. A study on seamless and application-oriented integration and a possible solution is introduced in Chapter 5. The conclusion and discussion on the industry impact are given in Chapter 6.

Chapter 2

Lifecycle Scheme and Its Components

In this chapter, we provide a simple lifecycle methodology designed according to our motivations. Notes on the implementation of its particular phases are given, too. We discuss our approach after presentation of the related work in each of the respective sections.

2.1 Applied Methodology

Here we develop our motivations in more detail within an overview of relevant approaches in the literature and presentation of a scheme of our approach to the implementation of the lifecycle.

2.1.1 Related Work

There has been focused research in some recent EU projects on tasks such as collaborative ontology development, ontology alignment and conflict resolution, and evaluation of ontology content, but little work until now has focused on the lifecycle as a whole. Recent overviews of the state-of-the-art in ontologies and related methodologies can be found in [SS04] and [GPFLC04]. However, none of them offers a direct solution to the problems mentioned in 1.1.

Methontology [FLGPJ97] is a methodology developed in the *Esperonto* project for designing ontologies to serve as a base for extending it towards evolving ontologies. The methodology proposed there is primarily based on the IEEE standard for software development [Sch97]. *Methontology* is provided with an ontology lifecycle based on evolving prototypes [FLGPR00]. It defines stages from specification and knowledge acquisition to configuration management, basically following the software management lifecycle. The particular stages and their requirements are characterised, but rather generally. The automatic ontology acquisition and evaluation methods are considered in *Methontology*, however, no distinction is made in their placement within the lifecycle.

The ODESeW and WebODE suite [CLCGP06] developed in the *Esperonto* and *Knowledge Web* projects provide an infrastructure and tools for semantic application development/management, which is in the process of being extended for networked and evolving ontologies. However, they focus rather on the application development part of the problem than on the ontology evolution parts. Aspects of the ontology lifecycle have also been addressed in the EU projects *SEKT*, *Dot.Kom*, *AKT*, *Esperonto*, *OntoWeb*, as well as in *Knowledge Web*, and will be the main focus of the recently started project *NeOn*[1].

2.1.2   Presentation of our Approach

These projects have all focused on either a single part of ontology development, or on a rather abstract study of the knowledge management cycle. When a methodology is delivered, it usually provides generic obligatory and/or optional requirements on the lifecycle's components and their combination. However, mechanisms that would provide a clue on how to incorporate the dynamics into the lifecycle are typically put off only by introduction of the version management, which we find insufficient in the scope of the remarks in Section 1.1. Moreover, the need for automatic methods of ontology acquisition in data-intensive environments is acknowledged but the place of the automatic techniques is usually not distinguished in the dynamic lifecycle settings. Summing up, it has been unclear so far how to tackle these particular problems within an extension of the current approaches to ontology evolution:

1. specific roles of the evaluation in the different phases of the lifecycle and respective appropriate methods of ontology quality assessment;

2. integration of the results of ontology learning and manual ontology development;

3. proper placement and utilisation of ontology alignment and/or meaning negotiation services within the lifecycle according to the previous two points.

Figure 2.1 below depicts the scheme of the proposed dynamic and application-oriented ontology lifecycle that deals with the problems mentioned above. The four main phases (*creation*, *versioning*, *evaluation* and *negotiation*) are indicated by the boxes annotated by respective names. The *creation* phase of the ontology lifecycle has two major parts as it consists of the automatic ontology extraction (ontology learning) and community-driven manual (collaborative) development. These directly correspond to the sliced component of the chart in Figure 2.1. Ontologies or their instances in time are represented by circles, with arrows expressing various kinds of information flow. The A boxes present actors (institutions, companies, research teams etc.) involved in ontology development, where A_1 is zoomed-in in order to show the lifecycle's components in detail.
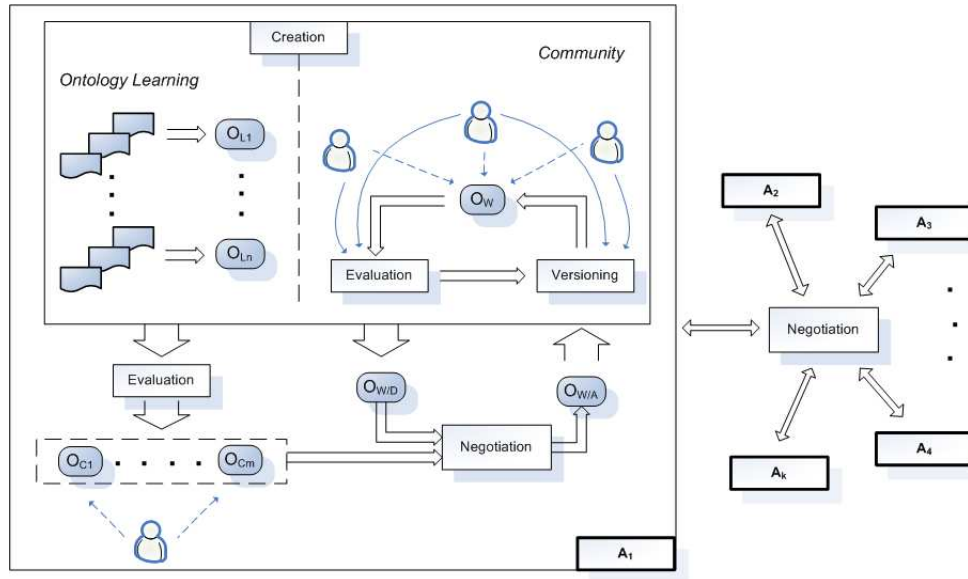
---

[1]http://www.neon-project.org

Figure 2.1: Dynamics in the ontology lifecycle

The general dynamics of the lifecycle goes as follows. The community experts (or dedicated ontology engineers) develop the (relatively precise and complex) domain ontology (the *Community* part of the *Creation* component). They use means for continuous ontology *evaluation* and *versioning* to maintain high quality and manage changes during the development process. If the amount of data suitable for knowledge extraction is too large to be managed by the community, *ontology learning* takes its place. Its results are *evaluated* and partially (we take only the results with quality above a certain threshold into account) integrated into the more precise (but typically relatively small) reference community ontology. The integration is based on alignment and merging covered by the *negotiation* component. The *negotiation* component takes its place also when interchanging or sharing the knowledge with other independent actors in the field.

In the following sections of this chapter, we give an overview of the components of the lifecycle, methods of their concrete realisation and summaries of related work for each particular phase. Further description of the dynamics in the lifecycle that universally reflects the dynamics of the knowledge flow in real application scenarios is in Chapter 3. The proposal of integration of the learned and collaborative ontologies is elaborated in detail in Chapter 5.

2.2 *Creation* **Component – Collaborative Techniques**

The first sub-component presents *manual* ontology creation. Within this phase of the ontology lifecycle, people from a community can directly create and update an ontology

using their expert knowledge. They may edit an ontology from scratch or extend a suitable top–level ontology by means of a collaborative user interface. The community people are presumably experts in their field, however, they are not generally proficient in knowledge engineering. Therefore the tools and interfaces employed within this part of the lifecycle should be as user friendly and intuitive as possible. The following set of requirements has been defined by [CK03] for supporting the development of large-scale collaborative ontology creation:

- easy to use, easy to communicate graphical representation;

- multi-user development environment;

- configuration management capabilities, including integration with version control;

- ability to view, analyse, compare and compose multiple models at a time;

- automated import/export of traditional ontology languages such as OWL or RDFS

There are currently very few existing tools which even come close to fulfilling all these requirements.

2.2.1   Available Methods and Tools

Collaborative ontology development can be realised in a number of ways. Some attempts have been made to develop ontology editors and environments which include facilities for collaborative ontology creation, such as Sandpiper Software's Visual Ontology Modeler (VOM) [KDM02]. This is an extension to Rational Rose[2], a visual modelling and development tool used widely in industry, enabling the development of ontologies through user-friendly wizards which create the necessary code automatically. The *Protégé* environment [GMF+03], one of the most well known ontology editing tools, presents a frame–based ontology engineering application with many plug-ins available. The Ontolingua server [FFR96] consists of a set of tools and services to support the process of achieving consensus on common shared ontologies by geographically distributed groups. OntoEdit [SEA+02] is an ontology editing environment combining methodology-based ontology development with capabilities for collaboration and inferencing, including support for collaborative generation of requirements specifications for the ontology (Onto-Kick) and support for collaborative brainstorming about the development process (Mind-2Onto).

Another method of collaborative ontology development involves collaborative portals [KBD05, ZKHF05]. A first version of the MarcOnt Portal [KBD05] has been recently published[3]. This hosts a collaborative ontology development framework created in DERI

---

[2]http://www.ibm.com/software/rational
[3]Preview access at http://portal.marcont.org/, the default credentials for a test user are login: *marcont*, password: *marcont*.

Galway. The project delivers a web-based environment where domain experts can cooperate on the ontology. The collaboration is achieved through the concept of suggestions of changes to the ontology. MarcOnt Portal uses SemVersion [VG06] to maintain suggestions (represented as changed instances of the ontology) together with main versions of ontology. Each suggestion can be evaluated/ranked and commented in the portal by the community of domain experts. Based on the comments and rankings, some of the suggestions are selected to be merged into the next version of the main ontology. Domain experts can work on suggestions with a web interface (similar to Protege) or upload their ontology in the form of an OWL file. Both ontologies and suggestions can also be accessed in a RDF/XML or N-TRIPLES format, or visualized using touchgraph. Users can also track changes between versions of ontologies using a semantic diff feature directly on the portal. The MarcOnt Portal also accepts suggestions from the community of users (in this case librarians) of the JeromeDL system [KDZ05]. Those suggestions are created from the RDF properties introduced by the librarians during annotation of the resources. This solution extends the folksonomies paradigm from keywords (objects) to named properties.

There have been various proposals also for using Wikipedia as a collaborative ontology editing tool, for example by using Wikipedia entries as ontology elements [HBS06], by upgrading the current Wikipedias Wiki software to create an ontology-editing module, where people can create their own ontologies associated with particular subjects in the same way that they can currently create and edit topics [Gia05].

Finally, a Wiki-like editing environment can be used such as Wiki@nt [BH03]. This consists of ontology modules, each composed of one or more wiki pages that can edited by multiple users, with version control and transaction management. Ontologies are then loaded into or uploaded from these wiki pages and managed by an ontology repository.

Alternatively, rather than have users create the ontology manually in a Wiki, they can use a controlled language such as CLONE[4] [TPCB06, PCTB06] or GINO [BK06] to create natural language text in a wiki, which is translated into an ontology by a special information extraction system (in the case of CLIE) or other means.

2.2.2 Discussion

Although widely used tools such as Protégé are good for a single user generating an ontology, they are not so good for collaborative ontology generation where many people may assist in the creation process and where many changes may be made to the ontology as it develops. This is due to several factors, such as the lack of an inbuilt mechanism for handling modular representation, and the lack of support for communication and cooperation among the multiple (human) ontology editors, in particular where modules may

---

[4]Note that CLONE (Controlled Language for ONtology Editing) is the new name for the actual controlled language used in CLIE, while CLIE (Controlled Language Information Extraction) remains the term used for the tool itself

be independently developed and may partially overlap when "glued together" [BH03]. At the time of Ceccaroni et al's report, VOM had some major drawbacks such as the inability to import certain existing ontologies and incomplete management of versioning.

We therefore look to the collaborative portals for our solution, and incorporate the ontology management services of the *MarcOnt* initiative [KBD05] for community-driven ontology development in the lifecycle scenario. We pick this solution for its clear design with universal mechanism of community collaboration, supported by incorporated versioning system SemVersion [VG06][5]. The portal-based architecture is to be enhanced by the controlled language (CLONE) interface introduced in [TPCB06]. The CLONE NL generation interface can also help us with development of a fancy mechanism of ontology integration (see Section 5.2 for details).

2.3 *Creation* Component – Automatic Techniques

The second component exploits methods of *ontology learning* from textual resources, using NLP (Natural Language Processing) and ML (Machine Learning) techniques. Methods for automated ontology acquisition are extremely useful when there is a large amount of relevant textual resources available.

It is tedious and expensive to process the knowledge in all the resources manually. Ontology learning can help significantly, although it suffers from significant precision/recall tradeoff and does not provide a "silver bullet" for ontological engineering at all [MS04]. However, when combined with human supervision, the results can be much better. We argue that the scheme of the integration of knowledge in learned and manually designed ontologies (further developed in Sections 3 and 5.2) can serve as a good example of this fruitful combination.

2.3.1 Available Methods and Tools

A comprehensive overview of other methods and tools for ontology learning from text can be found for example in [BCM05] and in [MS04].

An example of an ontology learning suite is Text2Onto [CV05], which builds on *GATE's* NLP and IE (Information Extraction) framework [CMBT02]. It employs several term extraction, taxonomy discovery and relation acquisition techniques and learns a statistical model of an ontology from text.

Text2Onto contains quite a generic set of tools for ontology learning, and therefore requires a certain amount of human intervention in order to achieve high quality results, working best in an incremental process combining human assistance with machine learning. One alternative is to use a tool for *focused* ontology learning such as that proposed

---

[5]See Section 2.4 for the discussion of this particular versioning system.

by [Sab05], which is designed for use in a narrow domain with a restricted sublanguage. Instead of the Machine Learning (ML) techniques used by most ontology learning approaches, this also uses GATE (like Text2Onto) but relies on the developer defining linguistic patterns to be matched, which lead to the identification of important terms for the hierarchy and relational information helping to position them correctly. This can be combined with visualisation tools to help the expert then analyse the generated ontology and correct it where necessary, as reported in the previous Knowledge Web deliverable D2.3.6 [MPSC06].

OntoLearn [GNV03] is another application for ontology learning. This finds terms in the corpus and tries to extract a definition of these terms using NLP methods or by asking an expert. The terms are arranged into hierarchical trees and linked to a core ontology suitable for the domain. An external ontology (e.g. DOLCE [MBG+02] or WordNet [Fel98]) is used for annotation with general semantic relations.

2.3.2 Discussion

The Text2Onto tool is the most suitable for this part of the lifecycle, for a variety of reasons. First, it is being continuously developed as an open source stand-alone application. Second, the probabilistic internal model supports well the presentation of plausible learned ontologies to human experts. Third, the tool uses an incremental learning process as described above, which supports the idea of ontology change. Finally, the tool comprises a suite of applications and therefore has quite a generic architecture compared with other, more specific tools, which is a very important criterion for a Human Language Technology tool (see the Knowledge Web deliverable D1.4.2v2 for further description of Best Practices in Human Language Technology).

2.4 *Versioning* Component

Versioning (or change management) is no doubt considered as a must in the area of collaborative software development. The same need arises even in ontology development, as stated in [VEK+05, KFKO02]. Versioning is especially important in our lifecycle scenario, as the knowledge is processed dynamically and continuously and thus we need to represent and handle several different versions of the same ontology.

2.4.1 Available Methods and Tools

[KFKO02] introduces theoretical specification, studies the versioning task and also mentions a RDF-based OntoView system inspired by CVS – the concurrent version system [Ber90] – a software management framework. CVS was initially a collection of scripts to simplify the handling of the revision control system (RCS). RCS operates in a

file-centric way by using a "lock-modify-unlock"-style. However, CVS works on the syntactic level, not on the conceptual level, i.e. it is not capable of versioning objects and in particular not capable of versioning ontological entities and their complex structure. The underlying diff operation is only capable of showing the syntactic differences between two files (based on the differences of text lines).

Following terminology from the database community (cf. [Rod95]), we mainly distinguish between ontology versioning and ontology evolution.

Ontology versioning is accommodated when an ontology management system allows for handling of ontology changes by creating and managing different versions of it. Ontology evolution is accommodated when an ontology management system facilitates the modification of an ontology by preserving its consistency (e.g. [Sto04]).

A first survey on causes and consequences of changes in an ontology is presented in [KF01], followed by an OntoView implementation for ontology versioning [KFKO02] that is based on the comparison of two ontology versions in order to detect changes. Basically, the system compares ontological classes, displays them side-by-side in RDF/XML and leaves it to the user to state whether it is an "identical" or "conceptual change".

Another example of an ontology versioning tool is KPOntology[6]. This is a library for versioning ontologies and allowing the use of different triple stores. A similar tool, SemVersion, is presented in [VG06]. This aims at a cost-efficient ontology versioning, identifying reusable parts of a versioning system and explaining where the system needs to be tailored to a particular ontology language. The re-use of existing components (such as the RDF-S reasoning offered by libraries like Jena) is emphasised. This keeps the code in SemVersion small and manageable.

2.4.2 Discussion

SemVersion [VG06] is a robust RDF-centric and efficient application of universal ontology versioning. It supports not only syntactic diffs, but also semantic ones (which is an improvement over the OntoView system, besides the higher level of automation of diff computation). The latter diffs are discovered using formal semantic underpinning of various RDF extensions like RDF Schema or OWL. The main difference when compared with the similar KPOntology system is given by the fact that KPOntology provides a separate library for each triple store, while the SemVersion approach is unified. Only the programmer using SemVersion will be aware of the triple store underneath (by changing one line of code), but to the end user, this will be transparent.

This SemVersion versioning system takes its place in the manual sub-component of the ontology *creation* phase. It supports commits by different users working on the same ontology, diffs between various ontology versions and other important operations that allow us to store, compare and process different instances of the ontology dynamically

---

[6]http://kpontology.isoco.com

changing in time.

2.5 *Evaluation* **Component**

The role of the *evaluation* component in our scenario is twofold, following the inner split structure of the *creation* component:

1. assessment of the ontology quality within the cycle of *collaborative ontology development* – can be used for example when selecting the preferred version of the ontology for a release dump; typically, there is no suitable gold standard to use in this lifecycle's phase, therefore evaluation methods that do not require a reference source for comparison must be applied;

2. assessment of the *learned ontology* quality in the *ontology learning* sub-component – aimed mainly at determination of appropriate candidate ontology to be passed further in the lifecycle (read more in Section 3.2 on this topic); in certain cases, the relatively precise ontology developed within the collaborative phase of the lifecycle can be used as a gold standard for NLP and ML based evaluation methods, which is the main difference compared with the conditions that hold for the previous point.

2.5.1 **Available Methods and Tools**

A survey of evaluation techniques is given in [BGM05]. Moreover, there is a Knowledge Web Deliverable D1.2.3 [HSG$^+$05] that presents a more complex and application–oriented overview.

   We can divide the evaluation methods into various categories. First, we have methods measuring *formal properties of ontologies*, such as OntoClean [GW00], which looks at properties such as rigidity and unity, and tools examining the *structure of ontologies*, such as ODEval [O. 04]. Second, we have methods which measure the *content* of ontologies, such as measuring how well an ontology has been populated, e.g. the Balanced Distance Metric (BDM) [MPL06, HSG$^+$05] and Learning Accuracy (LA) [HS98] which compare the populated ontology with a gold standard, taking into account the similarity between key and response in the two ontologies. An alternative method dealing with data-driven ontology evaluation is the measure of *ontology fit* [BADW04], based on the tennis measure [Ste02]. This is a measure of quality that can be easily computed without the need for human assistance: it captures (using cluster analysis) the "fit" between an ontology and the resources from which it has been created.

2.5.2   Discussion

The kind of evaluation tool needed depends on a variety of criteria such as the task under evaluation, the tool or process to be evaluated, and the availability of gold standards against which to form a comparison. Therefore, a multilevel evaluation concerning an ontology's lexical, syntactic and semantic properties combined with some kind of assessment of ontology suitability for given task and/or domain is needed for our lifecycle.

As stated in the first point in Section 2.5, the application of methods using a gold standard is not appropriate for the *collaborative creation* sub-component. The *OntoManager* tool presented in [HSG+05] is partially suitable for this task. However, data-driven and/or statistical evaluation methods [BADW04, BGM05] can also be employed in order to obtain an objective quality measure, when the resources from which the knowledge was acquired are easily identifiable and machine–readable.

On the other hand, more or less autonomous NLP, machine learning metrics and data driven evaluation techniques [HSG+05, BADW04] are suitable and even desirable for the quality assessment of different levels of *learned ontologies*. Namely, the metrics of precision, recall, F-measure or their extensions LA and BDM are employed. The ontology developed within the *collaborative creation* phase could present an applicable gold standard for these automated evaluation methods.

However, it is inappropriate to use the *collaboratively created ontology* as a reference in some cases, because its content may differ from the scope of the textual resources used for automatic ontology creation. In this case we should use the tennis measure described earlier. Note, however, that the tennis-measure by itself is not appropriate when a learned ontology was created using clustering. The tennis-measure is based on the overlap of clusters in an ontology and clusters in the data it was extracted from, therefore we should at least combine it with another method from the above-mentioned if we want to be honest when assessing the ontology quality.

2.6   *Negotiation* Component

When different versions or independently developed ontologies co-exist, or when existing ontologies are adapted for a new purpose, questions about the combined use of these ontologies throughout the ontology lifecycle become important. Especially the former situation is encountered within our lifecycle as we deal with learned and collaboratively designed ontologies within the *creation* component. We need to merge [PGPM99] them in this case, as seen in Figure 2.1, Section 2.1.2. Furthermore, if we want to exchange the knowledge between different institution in the general perspective of the Figure 2.1, it is very convenient to have a mapping between the ontologies in question that would help us to resolve possible similarities in their structure and lexical content.

Both tasks of merging and mapping of ontologies can be tackled by ontology align-

ment [EBB$^+$04] services. However, the nature of learned and manually designed ontologies slightly differs (as recalled in our motivations), thus the requirements on the preferred alignment may differ, too. We could for example prefer lexical alignments for the relatively simple learned ontologies and syntactic alignments for the collaboratively created ones. The different actors in the field may also have different preferences concerning the ontology alignment when exchanging knowledge among them.

Therefore, we need an agreement that would select an alignment (among many possible ones) that is as close as possible to different preferences related to the ontologies in question. Reaching this agreement can only come through a sort of negotiation process [ACMO04]. Thus, the *negotiation* component has the specific role of applying negotiation approaches for reconciling different ontologies as well as for the merging and alignment in the particular phases of the lifecycle.

The *negotiation* component can be applied in two different stages of our ontology lifecycle:

- intra-actor – when different ontologies (e. g. manually created and learned) need to be aligned and possibly merged within one actor (institution, research team, etc.);

- inter-actor – when ontologies released by different actors need to be aligned.

In contrast with the previous component (*evaluation*), we apply exactly the same techniques in both cases. However, we use them in different tasks with respect to the whole lifecycle (see Chapter 3), therefore we make this distinction.

2.6.1   Available Methods and Tools

Many ontology alignment tools have been proposed in the area of the Semantic Web. QOM [ES04] and its extension Foam [ES05] are based on heuristically calculated similarity of the individual ontology entities, and are characterised by their focus on the computational efficiency of the alignment. OLA [ELTV04] is dedicated to the alignment of OWL-Lite ontologies, and aims to use all the available information (i.e. lexical, internal and external structure, extensional, and data-types) extracted from two given ontologies. Falcon [NJQ05] is an automatic tool for aligning ontologies that employs three distinct matchers in combination: a string-similarity matcher, a vector space model of domain terms, and an RDF graph matcher. Anchor-PROMPT [NM01] is able to produce new concept mappings by analyzing similar paths between a set of anchor matches, which have already been identified (manually or automatically). oMap [ST06] is a method and tool for automatically aligning OWL ontologies that uses different classifiers to estimate the quality of a mapping. [DMDH04] describes machine learning utilisation in ontology mapping. More details about these and other approaches can be found in [EBB$^+$04].

The following approaches address the use of negotiation between agents/people with respect to ontology alignments, which is not very common, though they present certain

advantages we cannot exploit within the above approaches. An ontology mapping negotiation [SMR05] has been proposed to establish a consensus between different agents using the MAFRA alignment framework [SR03]. It is based on the utility and meta-utility functions used by the agents to establish if a mapping is accepted, rejected or negotiated, but is highly dependent on the MAFRA framework and cannot be flexibly applied in other environments.

Van Diggelen et al. [vDBD$^+$05] present an approach for agents to agree on a common ontology in a decentralised way. Rather than being the goal of any one agent, the ontology mapping is a common goal for every agent in the system. Beun et al. [BvEP05] present a computational framework for the detection of ontological discrepancies in multiagent systems by using feedback utterances. Bailin and Truszkowski [BT02] present an ontology negotiation protocol that enables agents to exchange parts of their ontology, by a process of successive interpretations, clarifications, and explanations. The end result of this process is that all the agents will converge on a single, shared ontology.

In [MSR06] the authors propose an argumentation framework for inter-agent dialogue to reach an agreement on terminology, which formalizes a debate in which the divergent representations (expressed in Description Logic) are discussed. The proposed framework is stated as being able to manage conflicts between claims, with different relevances for different audiences, in order to compute their acceptance, but no details are given about how agents will generate such claims.

There is a recent implementation of agent-inspired approach proposed in [LTE$^+$06]. Provided with a set of possible mappings between ontologies, the agents can reach a consensus on the terminology they use. In order to compute the preferred ontology alignments, the approach uses a specific argumentation framework – the Value-based argumentation framework [BC03] – allowing each agent to express its preferences between the categories of arguments that are clearly identified in the context of ontology alignment. The potential alignments are generated externally, e.g. using an *Ontology Alignment Service (OAS)* [EV04]. An alignment consists of a set of correspondences between two ontologies. Moreover, it assumes that for each correspondence, an OAS is able to provide a set of justifications, that explain why it has generated a candidate mapping. Agents use such information to exchange arguments supplying the reasons for their mapping choices. Every agent also has a private threshold value. This is compared with the degree of confidence that an OAS associates with each mapping. An agent can refuse a mapping with confidence lower than the threshold without negotiating. Although few approaches for ontology alignment provide such justification [SGPM05, DLD$^+$04], tools like [ES04] combine different similarity metrics, and these measures can be used to extend the system and provide the required justifications.

The consequent process reaching the alignment is fully automatic and does not require any involvement from human users. The framework contains means, by which the ontology engineers can express their preferred choices over candidate correspondences. This is achieved by adapting argumentation theory. Argumentation is based on the exchange

of arguments, against or in favour of a correspondence. The arguments interact with each other using an attack relation. Each argument instantiates an argumentation schema, and utilises the domain knowledge, extracted from extensional and intensional ontology definitions. When the full set of arguments and counter-arguments has been produced, the ontology developers consider which of them should be accepted. The acceptability of an argument depends on a ranking, represented by a particular preference ordering on the type of arguments.

2.6.2 Discussion

We utilise the recent work [LTE$^+$06] that deals with automatically *agreed* alignment between different ontologies. The acceptability of a partial mapping provided by an ontology alignment tool cannot always be taken for granted, as more of the possible alignments may be rationally acceptable by the agents in the lifecycle.

The approach based on [LTE$^+$06] is motivated by the fact that different and sometimes independent ontology engineer groups have their own objectives. For example, an ontology engineer may be interested in accepting only those mappings that have linguistic similarities, since its ontology is too *structurally simple* to realise any other type of mismatch[7]. In addition, any decision on the acceptability of these mappings has to be made dynamically, as the knowledge flow in the lifecycle is dynamic from its very nature. The argumentation based framework in [LTE$^+$06] has been recently implemented and is in the process of further improvement. Thus it can be very naturally incorporated even in the emerging lifecycle's implementation. Moreover, it fits nicely into the twofold role of the *negotiation* component in the proposed scenario, as described further in Section 3.

In particular phases of the lifecycle, the layman-oriented negotiation interfaces of the MarcOnt Initiative [KBD05] are also applied. See Sections 3 and 4 for further information on the concrete realisation of alignment within the ontology lifecycle's dynamics.

---

[7]The example corresponds also to the integration of simple learned ontologies with complex manually designed ones, which is very important for our scenario.

Chapter 3

Connection between the Components and its Dynamics

In the previous chapter we presented a proposal for a lifecycle chart and described its components, namely all the interwoven phases and technologies developed and/or suggested in this work package. We now develop the dynamics of the lifecycle itself and comment further on particular roles of the components within the lifecycle.

The dynamics of the whole ontology lifecycle that reflects the dynamics of the knowledge flow in typical application scenarios can be divided into three levels of different granularity (from finest to coarsest):

1. the ontology development cycle inside the collaborative part of the *creation* component (Section 3.1);

2. the cycle of integration of automatically learned ontologies with manually developed ones (Section 3.2);

3. mediation of ontology releases from different actors in the field (Section 3.3).

The first dimension reflects the process of iterative creation and improvement of a community ontology. However, since the knowledge is dynamic, it is therefore sometimes impossible to process all relevant knowledge by the collaborative tools, when there are extensive resources available and new ones are emerging over time. The second dimension reflects this kind of dynamics via the incorporation of automatically extracted knowledge into the master collaborative ontology. The third dimension conforms to the dynamic need for interchange and mediation of the knowledge between different actors with the same interests, so that the ontologies can be mutually improved and/or extended.

17

3.1 Community Sub-cycle

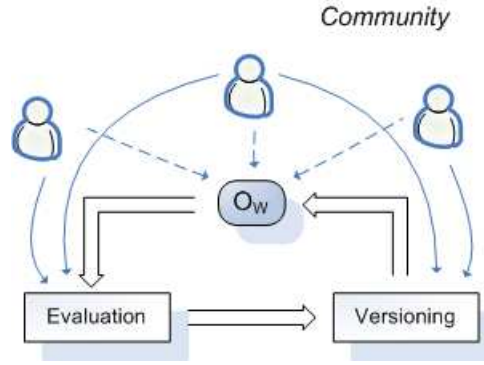The cycle of community driven ontology development is recalled in Figure 3.1.

Figure 3.1: Community ontology creation sub-cycle

Domain experts (denoted by the "people" icons) belonging to one actor (institution, research team, etc.) collaborate in a single domain ontology development, using a dedicated portal [KBD05] (see Figure 3.2 for the screenshot of the MarcOnt Portal ontology editor) and/or a controlled language interface such as CLIE [TPCB06, PCTB06]. These experts perform operations (expressed by the dashed arrows in Figure 3.1) such as insertions, editing or deletions on a master working ontology (the O_W circle).

The quality of the ontology as it changes over time is assessed by the *evaluation* component. Evaluation by community peers is mainly applied in this phase. Manual or machine-assisted evaluation is generally preferred here, since there is currently no proper method of automatic and objective evaluation specifically for a collaboratively created ontology. As discussed in Section 2.5.2, the *OntoManager* [HSG+05] tool combined with data-driven evaluation could be suitable here.

The *versioning* component only takes into account changes preserving or improving previously defined quality criteria. Figure 3.3 depicts a screenshot of the SemVersion interface in the MarcOnt portal. Both *evaluation* and *versioning* phases are supervised by human agents (designated domain experts), indicated by full arrows.

3.2 Intra-actor Cycle

The second level of the ontology lifecycle is realised by one actor when integrating results of ontology learning with a manually designed ontology, as shown in Figure 3.4.

The ontologies learned from different document sets (depicted by the $O_{L1}, \ldots, O_{Ln}$ circles in the diagram) are assessed by the *evaluation* component. The $O_{C1}, \ldots, O_{Cm}$ candidate ontologies ($m \leq n$) are selected following previously defined quality criteria.
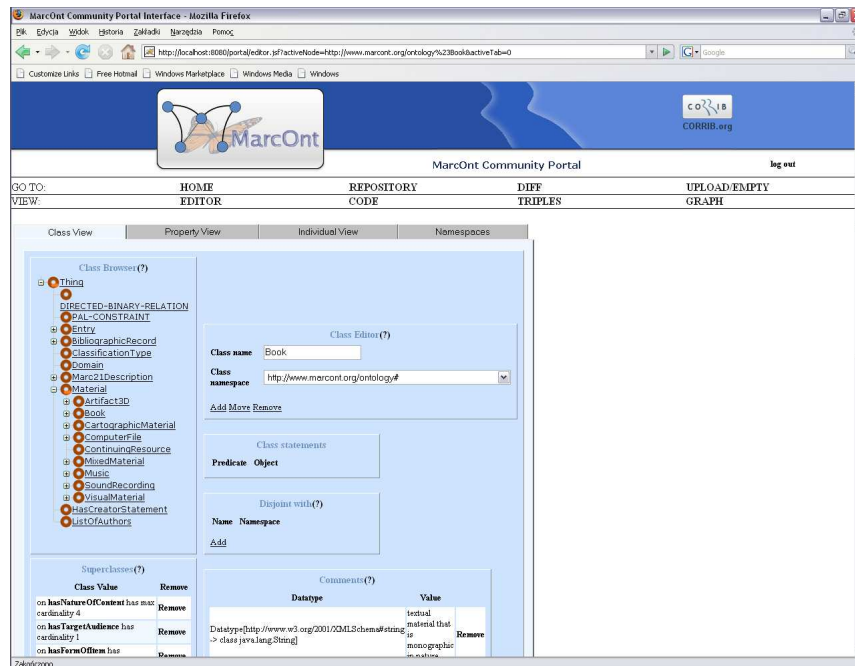
Figure 3.2: Editing an ontology in the MarcOnt portal

This process is supervised by designated human expert(s) again. Quality assessment of this phase utilises automated evaluation methods, as stated in Section 2.5.2. The O_W ontology serves as a gold standard here.

The m candidate ontologies are then used for population or general extension of the working ontology dump (depicted by the $O_{W/D}$ circle) by means of a *negotiation* component. Human intervention is also needed here, as shown in the diagram, for tasks such as selection of candidate ontologies, guided definition of argument sets and preference relations. Each ontology $O_{C1}, \ldots, O_{Cm}$ and $O_{W/D}$ is associated with an agent abstraction in the form of corresponding preference and argument sets and the agreed alignment is sought as described in [LTE$^+$06].

The agreed mapping found by the *negotiation* component in this phase is used to populate/expand [VPKV04] the dump of working ontology ($O_{W/D}$) and thus produce the resulting augmented ontology $O_{W/A}$. Expansion is performed as a side–effect of the negotiation process in this way: supposing the algorithm described in [LTE$^+$06] has found an agreed match between two or more ontologies, we can naturally use the *equivalent* and/or *subsumed* mapping relations to merge these ontologies.

The enriched ontology consequently replaces the former instance of the working O_W ontology (managed by the *versioning* component). The collaborative negotiation and also novel integration tools that are being developed within the MarcOnt Initiative [KBD05] can support the guided adoption of automatically proposed ontology extension $O_{W/A}$. The integration process is based on the technique described in Section 5.2. Thus we can

Figure 3.3: Version management of an ontology in the MarcOnt Portal
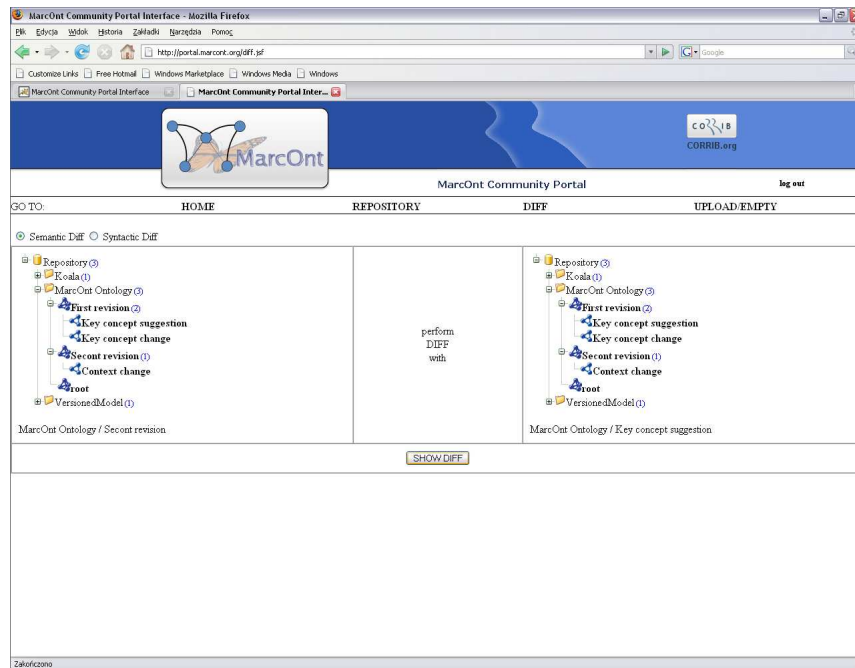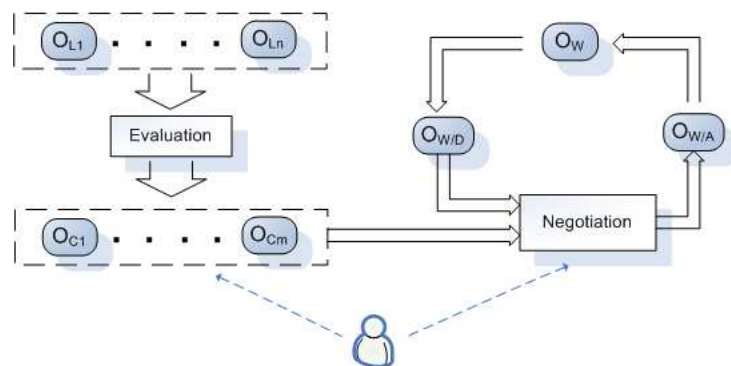

Figure 3.4: Intra-actor cycle

prevent possibly incorrectly matched or extracted parts from being added into the master working O_W ontology.

3.3 Mediation among Different Actors

When different actors involved in the same or similar domain create their own ontologies, the *negotiation* component needs to come to an agreement, as depicted in Figure 3.5. As
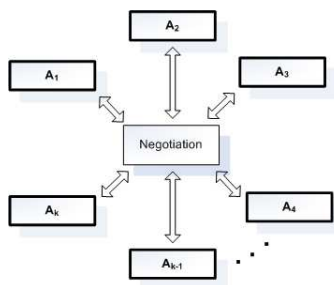
Figure 3.5: Inter–actor mediation

already mentioned, this is achieved by arguing about the meaning of the terms for different ontologies, described in [LTE$^+$06]. The approach has been implemented and offers three core functionalities, which we describe below.

User can (1) load the ontology to be negotiated, (2) set the preferences and (3) set the threshold, as shown in Figure 3.6. As has been already discussed, the setting can depend on several factors, e. g. the nature of the ontology or the context.
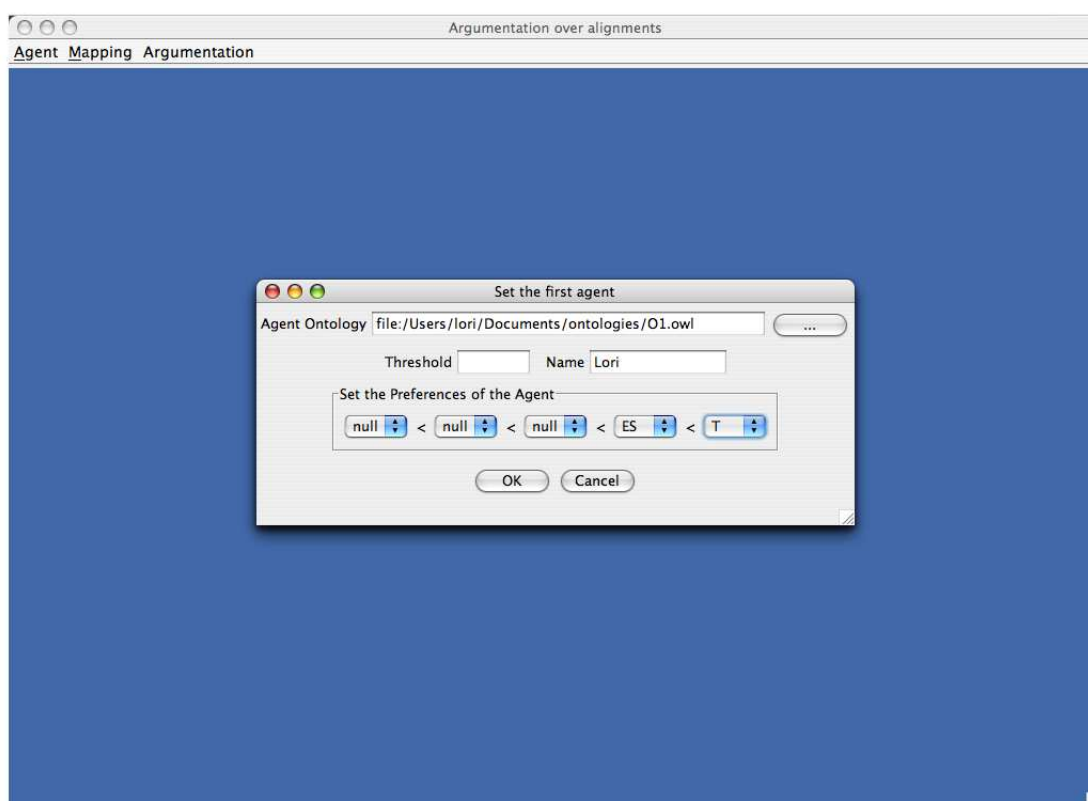
Figure 3.6: Screenshot – setting the preferences

The user is then provided with a set of potential ontology mappings. Each mapping

provides the justifications that explain why it has been generated (see Figure 3.7).
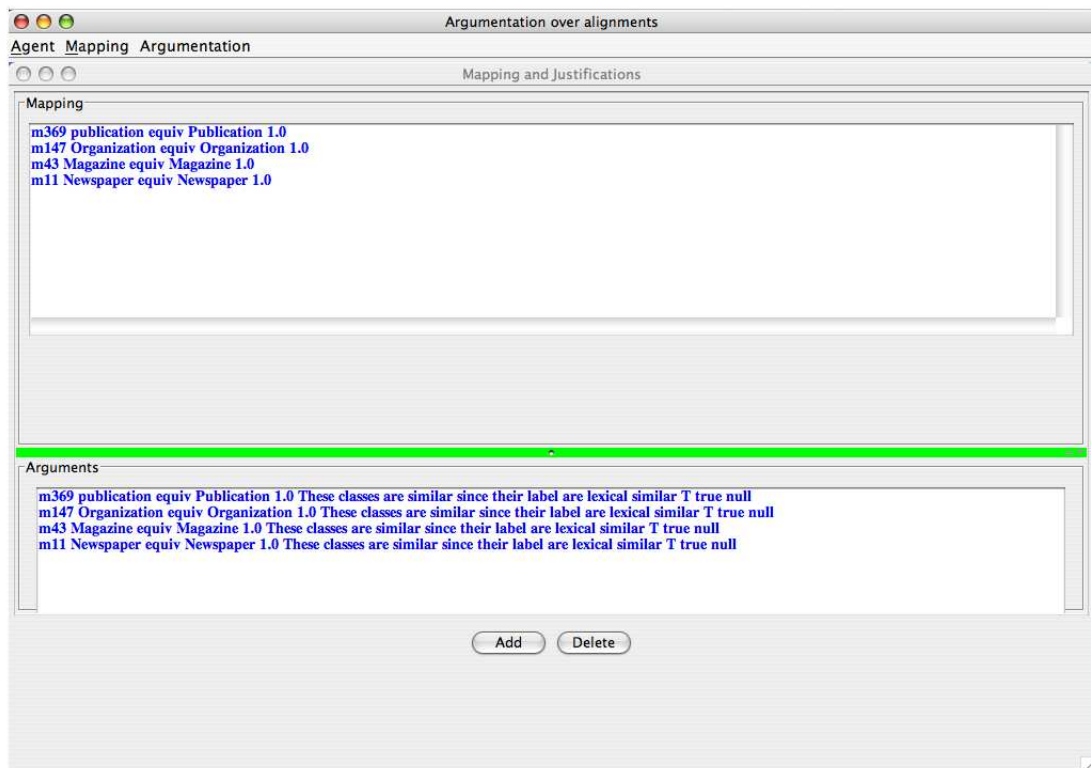
Figure 3.7: Screenshot – ontologies are aligned

Eventually, the tool produces the set of mappings that are agreed as acceptable for all actors involved (see Figure 3.8). This is done using the algorithm described in [LTE+06].
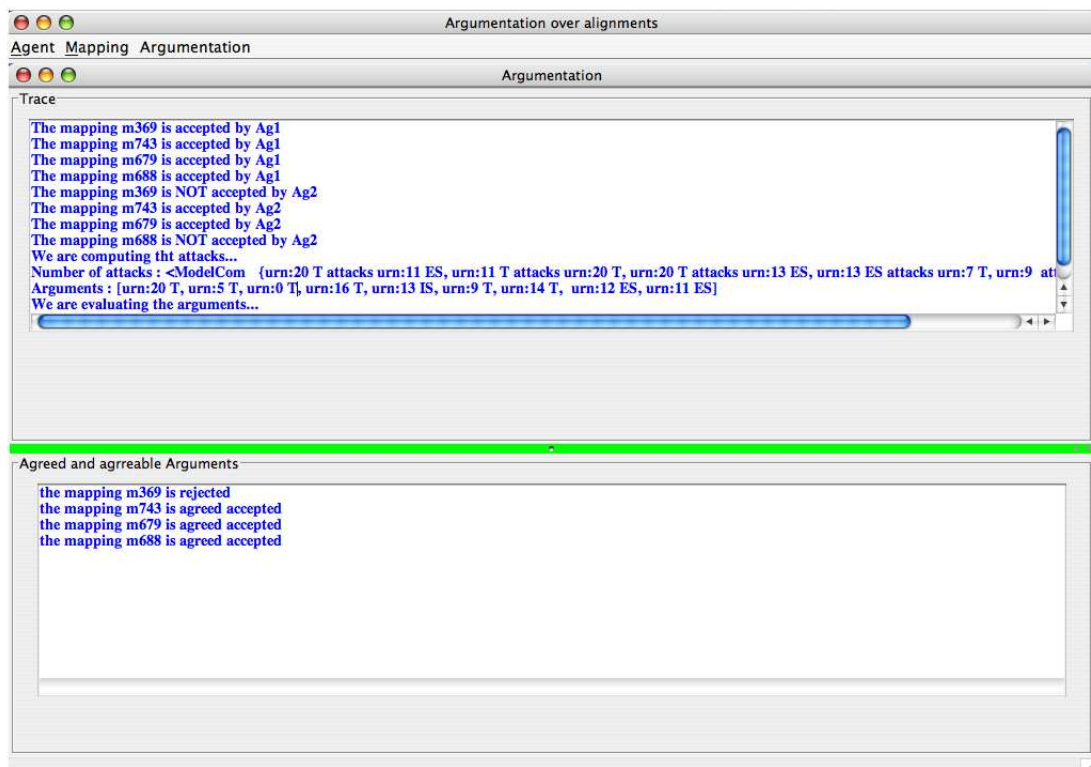
Figure 3.8: Screenshot – acceptable mappings are agreed

Chapter 4

Use Cases

It is extremely alluring to apply the ontology technology in the domains of biomedicine and healthcare. The need for structured and semantically-enabled data repositories is obvious and the implementation of the Semantic Web technologies in this area would no doubt have direct and valuable impact on the industry. Therefore we have decided to suggest two use cases from this domain. The application of the ontology lifecycle scenario in oncology is suggested in Section 4.1. Section 4.2 shows how the scenario can be applied in the knowledge flow in translational medicine. Of course it is possible to apply the ontology lifecycle scenario in any other sub-domain of medicine not presented here, e.g. in the case of development and intermediation of different anatomy ontologies, as mentioned in [ZB05]. We have picked the following examples for two main reasons. They are rather novel to the best of our knowledge and they clearly promise significant enhancement for the industry.

Both of the presented use cases relate to the business cases 2.12 (*Hospital Information System*) and 2.16 (*Integration of Biological Data Repositories*) from the Knowledge Web industrial deliverable D1.1.2 [NM04]. The connection with the business cases is further developed in Section 4.3.

4.1   Oncology

Oncology forms a special field of medicine in the sense that it overlaps with many other branches of medicine, such as internal medicine, gynecology, neurology, dermatology and/or urinology, because a cancer can appear in any sub-system of human body these disciplines are concerned about. Therefore various rather disparate, but integrated fields of knowledge should be on hand for oncologists in order to efficiently support the day-to-day medical decision processes. A very important part of this can be naturally realised by application of the presented ontology lifecycle scenario to the knowledge acquisition, representation and maintenance in oncology.

4.1.1 Motivation

Currently, the knowledge in a medical (oncological) institution is typically scattered among various database repositories with unclear semantics and large amounts of medical records in semi-structured natural language. In oncology, this problem is even bigger, as the relevant information could be in an external specialised resource the oncologists are not even aware of.

It is currently impossible to utilise this knowledge optimally in everyday practice, as there is no way to deliver all the relevant information from such sources at a moment when it is needed. In order to get the complete information available, one would need to query all the databases, search through all the patient records and then make a structured representation of the results of the search in order to get the overall idea. This is obviously impossible in clinical or even research practice.

On the other hand, formal ontologies are provided with efficient and universal mechanisms for querying and reasoning. Moreover, they present structured representation of knowledge by themselves and can serve as a base of knowledge mediation. Our ontology lifecycle scenario clearly defines the phases of ontology evolution and offers suitable tools and methods for its realisation. Furthermore, the dynamic nature of the scenario can handle the changing and growing of knowledge in the medicine settings (new patient records, new treatment methods applied in practice, new drugs, etc.), which is also very important for the practical application.

4.1.2 Realisation

Figure 4.1 demonstrates the concrete application of the scenario in the oncology domain.

The *Oncology* part of the figure shows an instance of the intra-actor cycle (described in Section 3.2) within an oncology institution. O_{Onc} represents the master ontology initially developed by community experts using collaborative MarcOnt Portal. The experts utilise their personal knowledge and/or existing external sources. We suggest some examples of these sources, as shown in the diagram:

- UMLS[1] (*Unified Medical Language System*) is a semantic network covering general conceptual hierarchies of medicinal knowledge, that can be utilised to design the core of the O_{Onc} ontology;

- GO[2] (*Gene Ontology*) is a large but rather informally modelled repository of knowledge concerning gene structures and processes, that can enhance the initial ontology by specific classes of genes and cellular metabolism processes related to effects of drugs and/or diseases;

---

[1] http://umlsinfo.nlm.nih.gov/
[2] http://www.ebi.ac.uk/ego

Figure 4.1: Lifecycle scenario applied in the oncology domain
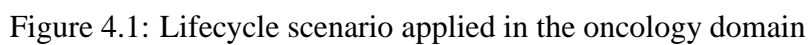
- ChEBI[3] (*Chemical Entities of Biological Interest*) is a dictionary of molecular entities focused on chemical compounds, that can enhance the initial ontology with information on active parts of drugs and/or toxins;

- Patient ontology enhances the initial ontology with expert knowledge on patients, their relations to other medical concepts and their typical properties in the medical point of view;

- Clinical ontology enhances the initial ontology with expert knowledge on specific clinical procedures, concepts and relations (diagnostic methods, clinical tools, therapies, etc.).

Note that the massive collaborative effort of designing the initial ontology is needed only once. After development, evaluation and possible revisions of the top conceptual structure, only minor changes are generally required. Thus there is no need to establish a permanent dedicated staff for this task, though an initial investment into the master working ontology development is still necessary.

The second part of the lifecycle's *creation* component – the ontology learning – plays the main role after establishment of the master collaborative ontology. Domain-specific resources are continually processed by the Text2Onto knowledge extraction tool. The resources are mainly EMRs (Electronic Medical Records), i.e. patient records in a semi-structured natural language which are electronically stored. The fact that they are pre-structured makes these resources relatively easy to mine in a precise manner. They also comprise relevant scientific papers or clinical trial protocols[4], which document the execution of the clinical trials on the human patients involved. Proprietary data stores of these natural language resources can also be utilised after appropriate simple preprocessing.

When the results of ontology learning are evaluated, the ontologies qualified above a certain threshold (for example those having F-measure above 0.6 as depicted in the Figure 4.1) are integrated into the master ontology in order to extend it with more specific domain concepts, relations and instances. The integration utilises the *negotiation* component (see Section 2.6) and the method proposed in Section 5.2.

The oncology ontology is mainly related to cancer and its treatment. However, as we mentioned earlier in this section, it also deals with another medicine disciplines. It is very likely that it shares some terms with these related fields. If there are respective ontologies for these domains, we can try to map the oncology ontology to them and see how we can improve or refine it using the knowledge from another fields of medicine. For this task, we need a specific ontology mapping tool. A solution to the problem is provided by the *negotiation* component of the lifecycle. After defining the respective preferences and initial

---

[3] http://www.ebi.ac.uk/chebi

[4] Briefly put, clinical trials are studies of the effects of newly developed drugs on selected sample of real patients. See http://en.wikipedia.org/wiki/Clinical_trial for general overview. [Com97] presents detailed EU specification of the conduction and documentation of clinical trials. The clinical trial protocols *de facto* document the execution of the trial with human patients involved.

alignments for the ontologies involved in negotiation, it can directly mediate the oncology ontology with concepts and relations from internal medicine, dermatology, radiology (the $O_{Int}, O_{Derm}, O_{Rad}$ ontologies in Figure 4.1[5], respectively) and other fields. Thus we can refine and/or extend the terminology and assertions with possibly more specific and complex knowledge on special topics in the respective disciplines.

The primary role of the ontology in this use case is to integrally represent evolving oncology knowledge and efficiently support the decision process in everyday clinical practice as well as to provide a base for specific research. Automatically generated mapping to another medicine ontologies allows the users to directly enhance their own ontology by the potentially very valuable knowledge from related disciplines.

4.2   Translational Medicine

The W3C consortium has recently recognised the challenges that translational medicine poses to the Semantic Web initiative [KHA05]. Briefly, the term "translational medicine" denotes an area of medicine that studies mutual benefits of clinical medicine (*bedside*) and basic research (*bench*) in the sense of bi–directional transition of results between these two fields.

4.2.1   Motivation

Both research and clinical practice in the medicine domain are very data-intensive fields. It is virtually impossible to re-use and fully utilise the "traditionally" stored data even within one institution, as mentioned in the previous section.

Moreover, it is difficult to mediate data between different institutions if there is no meaning that could give us a hint on how to map the knowledge further. Knowledge representation, management and especially knowledge transition or mediation are the crucial aspects of the translational medicine infrastructure. An efficient solution to knowledge flow management would enhance research in translational medicine by automatic support for substantial amount of the related activities (e.g. the knowledge extraction, update or mapping).

We believe that implementation of the presented ontology lifecycle scenario can provide translational medicine with both methodology and tools for seamless knowledge transfer between the research and clinical practice. It can also positively influence even the "isolated" knowledge management on both sides by delivering common framework for it. Moreover, the automatic tools delivered within the scenario (namely the *ontology*

---

[5]Note that the lines labelled MAP in the mediation part of the figure present mapping relations – if there is an arrow, the line goes from the more specific concept or relation to the less specific one; if there is no arrow, the line symbolises equivalence.

learning and *negotiation* components) could dramatically increase the efficiency of the whole process with significant benefits for the industry.

4.2.2 Realisation

Figure 4.2 shows an example of the application of the ontology lifecycle scenario in the translational medicine domain. It is divided into various parts, symbolising the *bench*, *bedside* and *transition* parts that are described in the three following sections, respectively. Note that the principle of the lifecycle scenario application is similar to the oncology-specific realisation described in Section 4.1.2. However, the aims and thus also particular parts differ.

Bench

O_{BN} in the diagram represents the master ontology initially developed by research community experts using collaborative MarcOnt Portal. The experts utilise their personal knowledge and existing external sources (namely UMLS, GO and ChEBI) that were already described in Section 4.1.2.

The developed ontology is quite similar to the one built in the same stage of the oncology use case. However, it is much more general and does not take the patient and clinical knowledge primarily into account, since this kind of information is rather irrelevant at the bench side.

The initial effort of master ontology design has also to be done only once. Consequently, the ontology is continually extended by the learned knowledge in the very same way as described in Section 4.1.2, although the processed resources slightly differ. Electronic medical records are not processed at the bench side, but scientific papers still provide relevant information.

Other resources worth mentioning are the clinical trial preceding data. As clinical trials present a well-founded methodological characteristic for the translational medicine, they are prominent within this use case. The preceding data contain results (generally in natural language) of experiments (laboratory tests, etc.) prior to execution of the trial phase with human patients. As such, they contain valuable data with respect to the basic research in tasks similar to the domain of the respective trial.

The main role of the ontology in this phase is to serve as a base for mediation between the bench and bedside. However, it can also provide the researchers with a semantically-enabled repository of the previous research results and their general conceptual background. State of the art ontology reasoning and querying techniques allow them to efficiently use even the knowledge implicitly present in the dynamically extended bench ontology. This can eventually support the research itself.

Bedside

The initial *collaborative creation* phase is rather minimalistic here – the appropriate fraction of the ontology developed in the bench side of the use case ($O_{BN/S}$) is sufficient as a working ontology. Optionally, it only needs to be initially extended by a very basic taxonomy of clinically-specific concepts and relations – in the sense of the clinical and patient ontology introduced and discussed in Section 4.1.2.

The *ontology learning* component plays a major role at the bedside, processing the vast amounts of data in the form of electronic medical records and clinical trial protocols that describe the part of clinical trials conducted with human patients involved. The master working ontology O_{BD} is then extended with this "practical" knowledge in the same manner as described in Section 4.1.2.

The side-effect of the lifecycle scenario application in the clinical practice is similar to the role ontology plays in the oncology use case (universal and structured repository of the domain knowledge, easy to query base of information with meaning). However, the primary role of the ontology in this part of translational medicine is again to serve as a base for mediation between the bench and bedside.

Transition

Transition fully utilises the argumentation-based method [LTE$^+$06] of the meaning *negotiation*. Ontologies O_{BN}, O_{BD} representing the bench and bedside parts, respectively, are associated (by human agents) with partial ordering that defines preferred mappings of concepts.

It can reflect for example preference of lexical mappings at the bench side, as researchers may be interested in studying concrete drug effects at the bedside. On the other hand, practitioners at the bedside may seek to augment their ontology with new methods proposed in basic research and thus would prefer structural mappings. After defining the respective preferences and initial alignment for the bench and bedside ontologies, the *negotiation* component establishes the agreed mapping between them.

This part of the use case utilises the ontologies developed at the bench and bedside and creates the mappings between some concepts, relations and individuals. The users from both sides can see how the concepts they are interested in are mapped to the other side. By browsing the vicinity of the mapped concepts, they can discover important consequences or facts from the other side. Following this new knowledge, they can extend their own ontology accordingly, incorporating it into their everyday clinical practice or research at the same time.
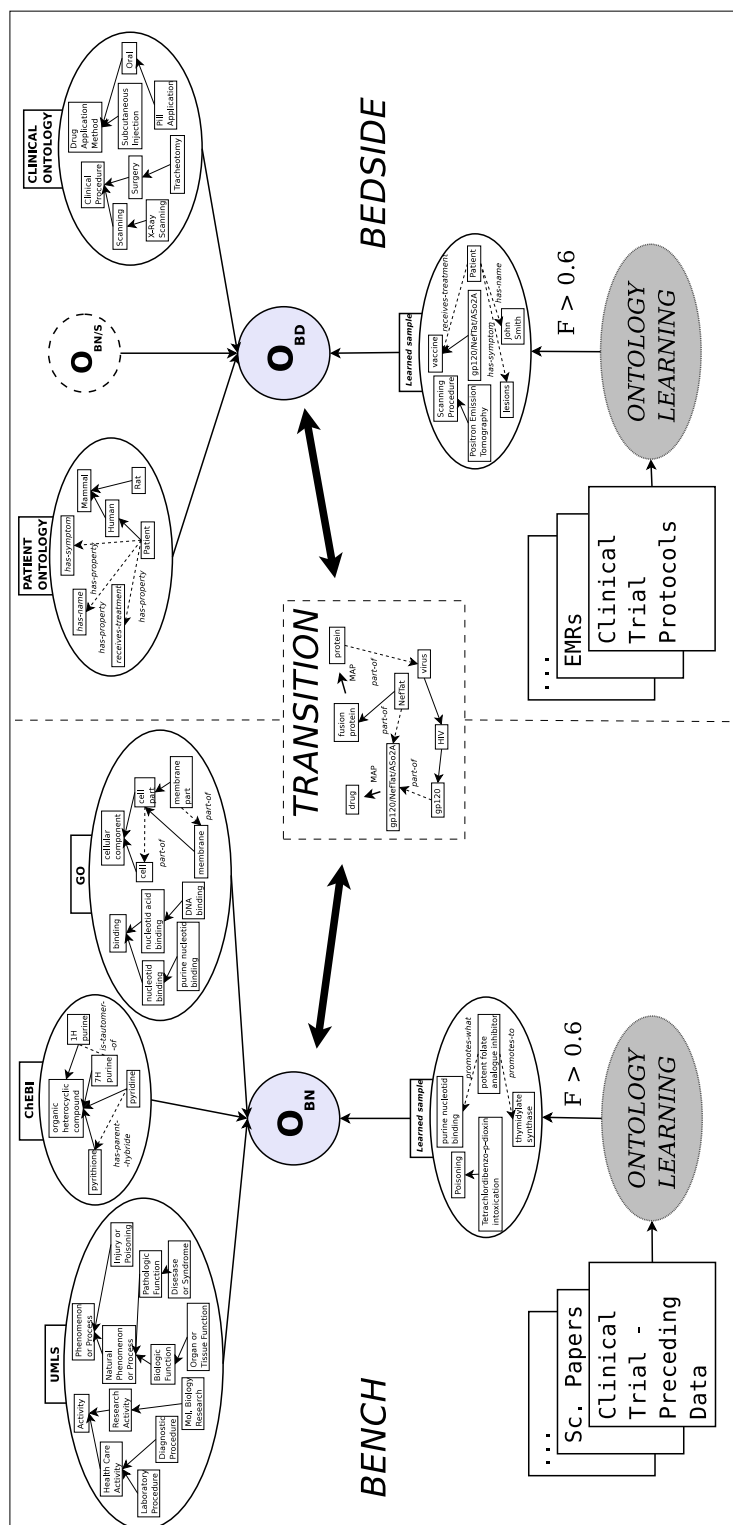
Figure 4.2: Lifecycle scenario applied in the translational medicine

4.3 Relation to Industrial Business Cases

The oncology use case (Section 4.1) is directly related to the business case 2.12 – *Hospital Information System* in [NM04]. The case challenges the heterogeneity of medical resources and difficulties with an intelligent consolidation and presentation of data. The presented lifecycle scenario provides a universal and coherent framework of representation, re-use and utilisation of medical data in ontologies. Moreover, there is no need to start completely from scratch within this new proposed framework, as the large amounts of knowledge currently present within the disparate resources can be at least partially extracted and adopted by means of ontology learning and integration proposed in Section 5.2. The business case 2.16 in [NM04] – *Integration of Biological Data Repositories* – is addressed mainly in the phase of initial ontology development, but there is no strong connection with the case's challenges and motivations.

On the other hand, the translational medicine use case (Section 4.2) explicitly deals with integration of various biological data resources both in the collaborative and ontology learning parts at the bench side. However, the connection is not so strong, as the data integration is motivated rather by the translational medicine needs than an ambition to create universal mappings between any such sources on the Internet. However, similarly to the oncology use case, the business case 2.12 aims are akin to what is going on at the bedside of the translational medicine use case.

Chapter 5

Notes on Seamless Integration

The scenario presented here provides a general methodology to connect different parts of the ontology lifecycle in the dynamic, continually changing environment. However, to make it really applicable, we must ensure that there is a common interchange framework allowing a seamless knowledge flow within the cycle. We first analyse the related topics in Section 5.1. Section 5.2 then discusses a proposal of a method aiming at the efficient semi-automatic integration of learned and collaboratively developed ontologies, which we see as a crucial problem within the integration of the particular phases of the lifecycle.

5.1 Analysis

As we can see in Table 5.1 below, the format common to all parts of the lifecycle is OWL [BvHH$^+$04]. Therefore, there is no need to tackle conversions between different formats – we can always rely on the RDF-based OWL standard.

When realising the content flow in the lifecycle, we have to take the following transitions into account:

1. $creation \rightarrow evaluation$ in the *ontology learning* phase;

2. $creation \rightarrow evaluation \rightarrow versioning$ in the *collaborative ontology development* phase;

3. integration of the *learned* and *collaborative* ontologies;

4. mediation of ontologies released by different actors in the domain.

Transition 1 presents no conceptual difficulties: once we have selected the appropriate NLP-based and data-driven evaluation methods, it is straightforward to apply them in this case by means of a script or a simple application. For the supervised evaluation, the OntoManager tool is directly applicable.

Component	Tools and/or methods	Languages
Creation (collaborative)	MarcOnt Portal, CLONE	OWL (DL variant)
Creation (ontology learning)	Text2Onto	RDF Schema, OWL, F-Logic (translated from internal representation)
Evaluation (collaborative)	OntoManager	RDF extension (internal format used in KAON framework [OVMS04]), but OWL supported as well
Evaluation (automated)	NLP metrics, statistic and data–driven measures	language independent
Versioning	SemVersion	RDF–centric, but handles also semantic change management based on RDF Schema and OWL
Negotiation (collaborative)	MarcOnt Portal	OWL (DL variant)
Negotiation (automatic)	Argumentation–based alignment framework	OWL

Table 5.1: Lifecycle (sub)components, respective tools, methods and supported languages

Transition 2 requires an extension of the collaborative MarcOnt ontology development portal by methods of unsupervised evaluation (the tennis-measure). Support for community-based evaluation is already incorporated here, as well as is the versioning. Again, the OntoManager tool is directly applicable here for additional formalised supervised evaluation. Transition 4 is solved by the automatic *negotiation* component.

Transition 3 is however tricky even from the conceptual point of view. We further discuss this topic and propose a solution and partial implementation in the following section.

5.2 Integration of the Learned and Collaborative Ontologies

There are very data-intensive disciplines (for example medicine) where it is usually not feasible to cover the whole domain only by collaborative means. In such cases, ontology learning can significantly help by producing possibly less precise and complex, but much broader ontologies that can enhance the primary ontologies developed by experts.

We have developed a proposal and partial implementation of a tool offering a solution to this task (see [NDKH07] for a recent description of the tool). It has been developed as a part of an extension to the MarcOnt Initiative – the *MarcOntX* agent. In principle, it is designed to generate natural language suggestions on relevant extensions from the learned

ontology to the community experts, who develop the collaborative ontology. Suggestions are automatically sorted according to their conformance to the priorities specified by the community.

5.2.1 Design of the Suggestion Generator

In the overview of the MarcOntX agent (see Figure 5.1), the O_E is an external ontology that contains possible extensions applicable to the master community ontology (represented with O_W). Note that the components of the figure related to RDF and JeromeDL conversion are specific to MarcOnt Initiative and are not relevant for the general perspective introduced here, even though they are part of the MarcOntX agent design. The MarcOntX agent core (the *suggestion generation* component) compares the O_E ontology with the dump of O_W ($O_{W/D}$) and generates the suggestions that are proposed to the community.
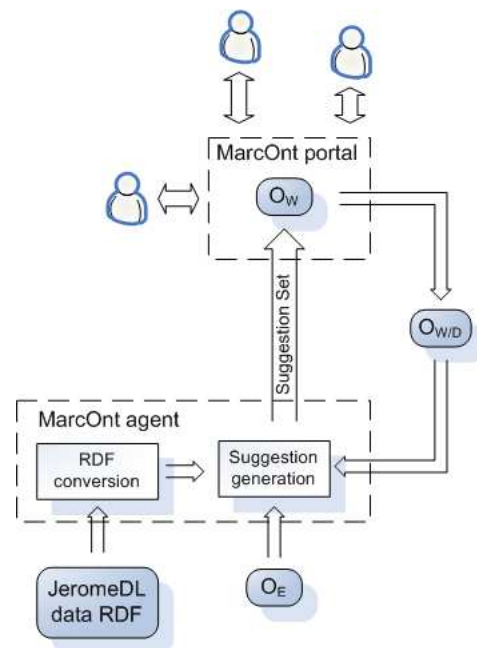
Figure 5.1: Architecture of the MarcOntX agent

Possible extension of a master ontology O_W by elements contained in an external ontology O_E naturally corresponds to the differences between them. These are discovered by means of the SemVersion library [VG06], which is a part of the MarcOnt Portal[1]. In particular, the possible extensions are equal to the additions O_E brings into O_W.

---

[1]Note that ontology alignment could also be used in the task of comparing two ontologies here. However, the diff computation is much simpler and easier to implement and interpret when generating the suggestions. Nonetheless, we actually use the alignment in application of the suggestion generation in our

However, the number of additions can be quite large, so an ordering that takes a relevance measure of possible suggestions into account is needed. Thus we can for example eliminate suggestions with low relevance level when presenting the final set to the users (without overwhelming them with a large number of possibly irrelevant suggestions). The suggestions are not meant to be exhaustive, but should rather identify a reasonable area of focus for the community, typically when extending the precise working ontology by broader results of ontology learning. And eventually, as the users are generally supposed to be laymen with respect to ontology engineering, the suggestions should be as simple and intuitive as possible. Algorithm 1 describes this basic idea in the form of a meta-code (the SemVersion library acts as a kind of diff operator here).

Algorithm 1 Meta-algorithm of the suggestion generation

Require: O_E, O_W — ontologies in RDF–based format
Require: $PREF \neq \emptyset$ — set of user preferences

1: $T_E \leftarrow makeRDFTriples(O_E)$
2: $T_W \leftarrow makeRDFTriples(O_W)$
3: $DIFF \leftarrow semVersion.getSemanticDiff(T_E, T_W)$
4: $TRIPLES \leftarrow DIFF.getAdded()$
5: $SORTED \leftarrow sortTriples(TRIPLES, PREF)$
6: $SUGG \leftarrow []$
7: **for** $T \in SORTED$ **do**
8:    $SUGG \leftarrow SUGG \oplus getNL(T)$
9: **end for**
10: **return** $SUGG$

The $makeRDFTriples()$ function creates RDF triples from non–RDF ontologies (thus being identity for RDF input). It is based on a simple method described in [TPCB06]. The method is not lossless in general (i.e. the precise semantics of complex OWL structures like union or other sophisticated constructors is not preserved). However, as was said before, we aim at the autonomous production of fairly simple suggestions, therefore the losses are not harmful with respect to the task in question. The suggestions are produced in the form of very simple natural language statements. These are obtained directly from the sorted triples, using a minor modification of the generation process in CLIE (the $getNL()$ function) described in [TPCB06]. We are currently experimenting with its incorporation into the MarcOntX agent. Concrete examples of such natural language suggestions are shown in the next section.

The system has to be able to process thousands of possible extensions from learned ontologies. The most important and tricky element is the sorting of the triples in the extending set. As a possible solution to this task, we have proposed and implemented a method (the $sortTriples()$ function) based on string subsumption and Levenshtein distance [Lev66]. These two measures are used within relevance computation by comparing the predicate, subject and object lexical labels of a triple to two sets (S_p, S_n) of words, provided by users. The S_p and S_n sets contain preferred and unwanted words respectively, concerning the lexical level of optimal extensions. The general structure of the sorting function is given in Algorithm 2. Note that the complexity of $HASH$ structure

lifecycle's implementation – see Section 5.2.2 for details.

sorting mostly contributes to the overall complexity of the relevance-based sorting of suggestions. As can be see from Algorithms 2 and 3, the complexity is in $O(mnl^2 + m \log m)$ (m – number of triples, n – number of words in the preference sets, l – maximal length of a word in triple labels, basically a constant), which gives $O(m(n + \log m))$. Since the size of the sets of user preferences can in practical terms be treated as constant, we obtain the $O(m \log m)$ complexity class with respect to the number of triples, which is feasible.

Algorithm 2 The triple sorting method

Require: $TRIPLES$ — list of triples
Require: $PREF = \{S_p, S_n\}$ — user preferences

1: $HASH = \{\}$
2: **for** $T \in TRIPLES$ **do**
3:     $HASH[getScore(T, S_p, S_n)] \leftarrow T$
4: **end for**
5: **return** $sort(HASH)$

The $getScore()$ function is crucial in the sorting algorithm. It is given by the formula:

$$getScore(T, S_p, S_n) = rel(T, S_p) - rel(T, S_n),$$

where $rel(T, S)$ is a function measuring the relevance of the triple T with respect to the words in the set S. The higher the value, the more relevant the triple is. We develop the relevance function in detail in Algorithm 3. Concrete examples of the sorting performance and its preliminary evaluation are provided in the following section.

The function naturally measures the "closeness" of the P, S, O labels to the set of terms in S_w. The value of 1 is achieved when the label is a direct substring of or equal to any word in S_w or vice versa. When the Levenshtein distance between the label and a word in S_w is lower than or equal to the defined threshold t, the relevance decreases from 1 by a value proportional to the fraction of the distance and t. If this is not the case (i.e. the label's distance is greater than t for each word in S_W), a similar principle is applied for possible word-parts of the label and the relevance is further proportionally decreased (the minimal possible value being 0).

5.2.2 Use Case and Preliminary Evaluation

Figure 5.2 shows the application of the MarcOntX agent when producing suggestions related to the extension of a bio-medical ontology by knowledge learned from domain resources (scientific publications, medical records, etc.). A mapping between the learned ontology and master ontology O_W is established by the *negotiation* component of the lifecycle (see Section 2.6). The ontologies are merged according to these mappings. Suggestions are produced from the comparison of the merged and the former O_W ontology and passed to the community members in order to assist them when extending O_W by the learned knowledge.

Algorithm 3 The relevance function

Require: $T = (P, S, O)$ — triple, consisting of P, S, O — *predicate*, *subject* and *object* labels respectively; possibly multiword
Require: S_w — set of words
Require: $w_r, w_c, \rho \in (0, 1)$ — real constants; by setting w_c or w_r non-equal to 1, we can favour the structure (predicate) or content (subject and object) parts of triples respectively; ρ influences the absolute value of relevance measure
Require: t — integer constant; maximal allowed distance
Require: $levDist(s_1, s_2)$ — Lev. distance implementation

1: $R_P, R_S, R_O \leftarrow 0$
2: **for** $elem \in \{P, S, O\}$ **do**
3:   **if** $elem$ is a substring of or equals to any word in S_w or vice versa **then**
4:     $R_{elem} \leftarrow 1$
5:   **else**
6:     $d \leftarrow \infty$
7:     **for** $v \in S_w$ **do**
8:       **if** $levDist(elem, v) < d$ **then**
9:         $d \leftarrow levDist(elem, v)$
10:       **end if**
11:     **end for**
12:     **if** $d \leq t$ **then**
13:       $R_{elem} \leftarrow (1 - \frac{d}{t+1})$
14:     **else if** $elem$ is a multiword term **then**
15:       $L \leftarrow$ set of single terms in the $elem$ label expression
16:       $EXP \leftarrow 0$
17:       **for** $u \in L$ **do**
18:         **if** u is a substring of or equals to any word in S_w or vice versa **then**
19:           $EXP \leftarrow EXP + 1$
20:         **else**
21:           $d \leftarrow \infty$
22:           **for** $v \in S_w$ **do**
23:             **if** $levDist(u, v) < d$ **then**
24:               $d \leftarrow levDist(u, v)$
25:             **end if**
26:           **end for**
27:           **if** $d \leq t$ **then**
28:             $EXP \leftarrow EXP + (1 - \frac{d}{t+1})$
29:           **end if**
30:         **end if**
31:       **end for**
32:       **if** $EXP = 0$ **then**
33:         $R_{elem} \leftarrow 0$
34:       **else**
35:         $R_{elem} \leftarrow \rho^{\frac{1}{EXP}}$
36:       **end if**
37:     **end if**
38:   **end if**
39: **end for**
40: **return** $\frac{w_r R_P + w_c 2 max(R_S, R_O)}{3}$

Even though this is an abstract example with no evaluation, we have already implemented the part we claim to be most crucial, as specified in Section 5.2.1: the relevance-based sorting algorithm. As a proof of concept, we performed its preliminary evaluation on a set of 19 randomly selected triples from the ChEBI[2], GO[3], UMLS[4] and Wikipedia[5] bio-medical resources. The $S_p, |S_p| = 10$ and $S_n, |S_n| = 6$ sets were artificially designed

---

[2]http://www.ebi.ac.uk/chebi
[3]http://www.ebi.ac.uk/ego
[4]http://umlsks.nlm.nih.gov/kss
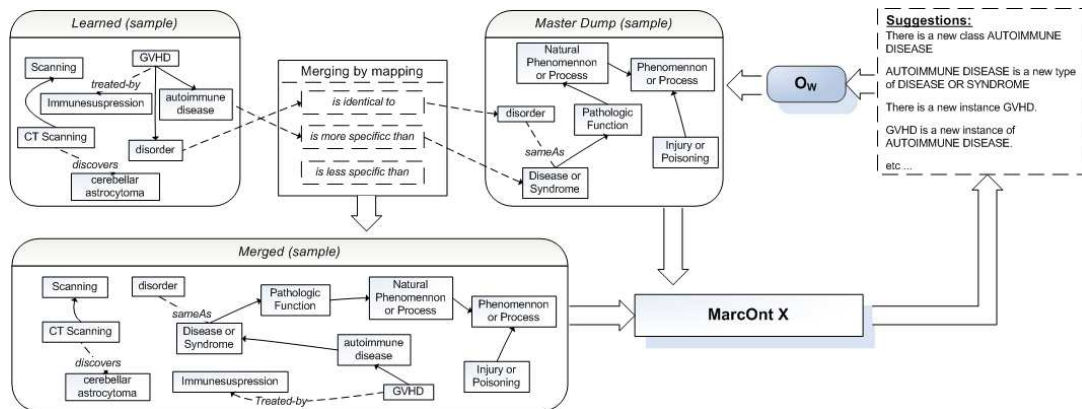[5]http://en.wikipedia.org/wiki/Chemotherapy

Figure 5.2: Medicine use case – ontology integration

to express interest in diseases and their symptoms and unconcern in chemical terminology and particular patients. We ran three sortings t_1, t_2, t_3 with no preferences and with structure and content preferences respectively. Samples of sorted triples and their relevance are given in Figure 5.3.

```
Triple:  has-form(GVH disease, acute)
    Relevancy:  0.8888888888888895
                    ...
Triple:  inhibit(steroids, tissue swelling)
    Relevancy:  0.19333333333333336
                    ...
Triple:  has-name(Patient, John Smith)
    Relevancy:  -0.4999999999999989
                    ...
```

Figure 5.3: Sample of sorted triples

We evaluated the correctness of placement of particular triples in the relevance scale according to the defined preferences. Table 5.2 shows the results. Note that the distribution of (ir)relevant triples would be most probably different in a real data-intensive environment, with much more triples in the $Rel = 0$ column, as the expert preferences cannot generally cover the extension space in whole.

	$Rel > 0$	$Rel = 0$	$Rel < 0$	Correct
t_1	11	1	7	84.21
t_2	9	3	7	78.95
t_3	12	1	6	78.95
avg	10.7	1.7	6.7	80.7

Table 5.2: Relevance–based sorting evaluation results

The average ratio of 80.7% correctly placed triples (with no single triple considered irrelevant when stated relevant by the algorithm and vice versa) has confirmed our design

objective so far. Nonetheless, tests with a real user community must be performed in order to carry out proper evaluation.

Chapter 6

Conclusion

We have presented a proposal of an ontology lifecycle scenario and its technical realisation in the environment of dynamic knowledge flow, which is characteristic for virtually any practical application of the Semantic Web technologies. In this final chapter, we sum up the key points of the proposed scenario, emphasise its relevance for the industry and show the directions of our future work.

6.1   Summary of the Presented Lifecycle Scenario

The scenario presents a simple, yet naturally applicable methodology for various phases of the ontology lifecycle. It is also provided by an application-oriented study on the integration of the separate phases. Problems concerning their integration were identified and solutions to them were devised and partially implemented in the trickiest case (see Section 5.1).

We have associated the respective phases with the suitable applications, discussed appropriateness of the selected solution and shown how the techniques are used all together. The evaluation methods were selected and placed into relevant phases of the lifecycle in order to assess the quality of the different ontology instances and/or parts in the most proper way (covering the open problem 1. out of three in the list presented in Section 2.1.2). The ontologies are evaluated not only after deployment, but also within the creation itself. Thus we can increase the quality of the results of whole ontology development process. We also identified the parts of the lifecycle that need further development, as summed up in Section 6.3.

A novel semi-automatic method of integration of learned ontologies into collaboratively developed ones has been designed and partially implemented (covering the open problem 2. out of three in Section 2.1.2). This makes the whole scenario directly applicable even in such data-intensive and highly dynamic domains as medicine. The technique of integration of learned and collaborative ontologies employs merging based on

41

an alignment agreed by the *negotiation* component. The component is also used when mediating ontologies being delivered by different actors in a field. These two applications cover the third and last open problem presented in Section 2.1.2.

6.2   Relevance for the Industry

The use cases in Section 4 clearly show how the presented lifecycle scenario can be applied in the bio-medicine domain, namely in oncology and in translational medicine.

In the medicine industry, an immense need has been recognised recently for newly structured data repositories. The demand is that such stores should provide the user with as much relevant data as possible without having to perform tedious and time-consuming search through scattered resources of medical knowledge. Ontologies offer a natural solution to this situation, as they are by definition repositories of structured knowledge. When adequately created and managed, they can help the medical experts better than the traditional non-semantic stores of the domain knowledge by themselves.

Another problem characteristic for classical medical knowledge management is that the applicable sources are heterogeneous. They do not adhere to any common standard and they are typically stored in many different formats from relational databases to natural language records. If these resources are transformed into ontologies, state of the art alignment techniques can then be used, mapping and inter-relating the knowledge previously stored in an incomparable form.

The application of the dynamic ontology lifecycle scenario copes with the above problems and suggest ways to solve them. Putting our use cases into the context of business cases identified in [NM04], we show the direct impact the implementation of the scenario would have on these industry areas, amongst others.

Moreover, we inherently implement several universal requirements presented in D1.2.4 – Architecture of the Semantic Web Framework. The scope of our lifecycle covers mainly the following dimensions and components of the framework:

- *ontology development and management* – we provide or work on the ontology editor, browser, evaluator and learner (4 out of 5) components of this dimension within the creation phase of the lifecycle;

- *ontology evolution* – ontology evolution visualizer and ontology versioner (2 out of 4) components are provided within the currently implemented versioning and collaborative creation phases of the lifecycle;

- *ontology alignment* – ontology matcher, alignment manipulation and ontology merger (3 out of 8) components are implemented or under development within the negotiation phase and ontology integration module of the lifecycle.

6.3   Future Work

In this deliverable, we have presented a conceptual vision of an ontology lifecycle scenario, together with concrete suggestions on its practical implementation. However, certain things remain to be done before it can be fully applied.

First, the argumentation based negotiation tool introduced in Section 2.6 should be extended by direct incorporation of the ontology alignment service in order to provide even the initial ontology mappings automatically. Second, the merging based on negotiation must be implemented within the integration of the learned and collaborative ontologies (as presented in Section 5.2). Third, the tools for NLP-based and data-driven evaluation suitable for its particular role in the lifecycle have to be implemented (or identified and adapted – for example the simple tools for LA and BDM computation introduced in [MPSC05] can be utilised). Last but not least, the work on the remaining parts of the suggestion generation algorithm designed in Section 5.2.1 must be finished.

After implementation of all the missing parts, identified in the previous paragraph, the whole concept of the dynamic ontology lifecycle scenario must be evaluated. And moreover, the evaluation should be preferably performed in cooperation with appropriate industrial partners in line with the suggested use cases. Thus we can continually transfer the proposed framework into the industry practice.

Bibliography

[ACMO04]   Karl Aberer, Philippe Cudré-Mauroux, and Aris M. Ouksel. Emergent semantics principles and issues. In *Proceedings of Database Systems for Advances Applications, 9th International Conference, DASFAA 2004*, 2004.

[BADW04]   C. Brewster, H. Alani, S. Dasmahapatra, and Y. Wilks. Data driven ontology evaluation. In *Proceedings of LREC 2004*, 2004.

[BC03]   T. Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *Journal of Logic and Computation*, 13(3):429–448, 2003.

[BCM05]   P. Buitelaar, P. Cimiano, and B. Magnini, editors. *Ontology Learning from Text: Methods, Evaluation and Applications*. IOS Press, 2005.

[Ber90]   B. Berliner. CVS II: Parallelizing software development. In *Proceedings of the Winter 1990 USENIX Conference*, pages 341–352. USENIX, 1990.

[BGM05]   J. Brank, M. Grobelnik, , and D. Mladenic. A survey of ontology evaluation techniques. In *Proceedings of the Conference on Data Mining and Data Warehouses (SiKDD 2005)*, 2005.

[BH03]   J. Bao and V.G. Honavar. Collaborative Ontology Building with Wiki@nt – A Multi-agent Based Ontology Building Environment . In *EON 2004 Workshop*, 2003.

[BK06]   Abraham Bernstein and Esther Kaufmann. GINO—a guided input natural language ontology editor. In *Proceedings of 5th International Semantic Web Conference (ISWC 2006)*. Springer–Verlag, 2006. (in press).

[BT02]   S. C. Bailin and W. Truszkowski. Ontology Negotiation: How Agents Can Really Get to Know Each Other. In *Proceedings of the WRAC 2002*, 2002.

[BvEP05]   R.J. Beun, R.M. van Eijk, and H. Prüst. Ontological feedback in multiagent systems. In *Proceedings of Third International Conference on Autonomous Agents and Multiagent Systems*, 2005.

[BvHH⁺04] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. *OWL Web Ontology Language Reference*, 2004. Available at (February 2006): `http://www.w3.org/TR/owl-ref/`.

[CK03] L. Ceccaroni and E. Kendall. A graphical environment for ontology development. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, 2003.

[CLCGP06] O. Corcho, A. Lopez-Cima, and A. Gomez-Perez. The ODESeW 2.0 semantic web application framework. In *Proceedings of WWW 2006*, pages 1049–1050, New York, 2006. ACM Press.

[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*, 2002.

[Com97] European Commission. Good clinical practice. Technical report, European Commission, 1997. Legislative basis — Directive 75/318/EEC, available at (November 2006): `http://ec.europa.eu/enterprise/pharmaceuticals/eudralex/vol-3/pdfs-en/3cc1aen.pdf`.

[CV05] Philipp Cimiano and Johanna Völker. Text2Onto - a framework for ontology learning and data-driven change discovery. In Andres Montoyo, Rafael Munoz, and Elisabeth Metais, editors, *Proceedings of the NLDB 2005 Conference*, volume 3513 of *Lecture Notes in Computer Science*, pages 227–238, Alicante, Spain, JUN 2005. Springer.

[DLD⁺04] R. Dhamankar, Y. Lee, A. Doan, A. Halevy, and P. Domingos. iMAP: Discovering complex semantic matches between database schemas. In *Proceedings of the International Conference on Management of Data (SIGMOD)*, pages 383–394, 2004.

[DMDH04] A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Ontology matching: A machine learning approach. In *Handbook on Ontologies*, chapter 19, pages 385–404. Springer–Verlag, 2004.

[EBB⁺04] J. Euzenat, Thanh Le Bach, Jesus Barrasa, Paolo Bouquet, , Jan De Bo, Rose Dieng, Marc Ehrig, , Manfred Hauswirth, Mustafa Jarrar, , Ruben Lara, , Diana Maynard, Amedeo Napoli, Giorgos Stamou, Heiner Stuckenschmidt, Pavel Shvaiko, Sergio Tessaris, Sven Van Acker, and Ilya Zaihrayeu. D2.2.3: State of the art on ontology alignment. Technical report, Knowledge Web, 2004.

[ELTV04]   J. Euzenat, D. Loup, M. Touzani, and P. Valtchev. Ontology alignment with OLA. In *Proceedings of the 3rd EON Workshop, 3rd International Semantic Web Conference, Hiroshima (JP).*, 2004.

[ES04]   M. Ehrig and S. Staab. QOM - quick ontology mapping. In *ISWC 2004: Third International Semantic Web Conference. Proceedings*, pages 683–697, 2004.

[ES05]   M. Ehrig and Y. Sure. FOAM - framework for ontology alignment and mapping; results of the ontology alignment initiative. In *Proceedings of the Workshop on Integrating Ontologies*, 2005.

[EV04]   J. Euzenat and P. Valtchev. Similarity-based ontology alignment in OWL-Lite. In *Proceedings of the European Conference on Artificial Intelligence (ECAI 2006)*, 2004.

[Fel98]   C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[FFR96]   A. Farquhar, R. Fickas, and J. Rice. The Ontolingua Server: a tool for collaborative ontology construction. In *Proceedings of the 10th Banff Knowledge Acquisition Workshop*, Banff, Alberta, Canada, 1996.

[FLGPJ97] M. Fernandez-Lopez, A. Gomez-Perez, and N. Juristo. Methontology: from ontological art towards ontological engineering. In *Proceedings of the AAAI97 Spring Symposium Series on Ontological Engineering*, pages 33–40, Stanford, USA, March 1997.

[FLGPR00] M. Fernandez-Lopez, A. Gomez-Perez, and M. D. Rojas. Ontologies' crossed life cycles. In *Proceedings of International Conference in Knowledge Engineering and Management*, pages 65–79. Springer–Verlag, 2000.

[Gia05]   F. Giasson. Wikipedia as a collaborative editing-ontology tool, 2005. Available at: `http://fgiasson.com/blog/index.php?title=wikipedia_as_a_collaborative_editing_ont&more=1&c=1&tb=1&pb=1`.

[GMF⁺03]   John H. Gennari, Mark A. Musen, Ray W. Fergerson, William E. Grosso, Monica Crubezy, Henrik Eriksson, Natalya F. Noy, and Samson W. Tu. The evolution of Protégé: an environment for knowledge-based systems development. *International Journal of Human–Computer Studies*, 58(1):89–123, 2003.

[GNV03]   A. Gangemi, R. Navigli, and P. Velardi. Corpus driven ontology learning: a method and its application to automated terminology translation. *IEEE Intelligent Systems*, pages 22–31, 2003.

[GPFLC04] A. Gomez-Perez, M. Fernandez-Lopez, and O. Corcho. *Ontological Enginee-ering*. Advanced Information and Knowledge Processing. Springer-Verlag, 2004.

[GW00] N. Guarino and C. Welty. Ontological analysis of taxonomic relationships. In *Proceedings of ER-2000: The 19th International Conference on Concep-tual Modeling*, 2000.

[HBS06] M. Hepp, D. Bachlechner, and K. Siorpaes. Harvesting Wiki Consensus - Using Wikipedia Entries as Ontology Elements. In *Proceedings of the 1st Workshop SemWiki2006 - From Wiki to Semantics at ESWC 2006*, 2006.

[HS98] U. Hahn and K. Schnattinger. Towards text knowledge engineering. In *Proc. of 15th National Conference on Artificial Intelligence (AAAI-98)*, pages 524–531, Menlo Park, CA, 1998. MIT Press.

[HSG+05] J. Hartmann, P. Spyns, A. Giboin, D. Maynard, R. Cuel, M. C. Suarez-Figueroa, and Y. Sure. Methods for ontology evaluation (D1.2.3). Deliver-able 123, Knowledge Web, 2005.

[KBD05] S. Kruk, J. Breslin, and S. Decker. MarcOnt initiative. Líon Deliverable 3.01, DERI, Galway, 2005.

[KDM02] E.F. Kendall, M.E. Dutra, and D.L. McGuinness. Towards a Commercial Ontology Development Environment. In *International Semantic Web Con-ference (ISWC), Late Breaking Topics*, 2002.

[KDZ05] Sebastian Ryszard Kruk, Stefan Decker, and Lech Zieborak. JeromeDL - Adding Semantic Web Technologies to Digital Libraries. In *Proceedings of DEXA'2005 Conference*, 2005.

[KF01] Michel Klein and Dieter Fensel. Ontology versioning for the semantic web. In *Proceedings of 1st Int Semantic Web Working Symposium*, pages 75–91, Stanford, CA, USA, 2001. Stanford University.

[KFKO02] Michel Klein, Dieter Fensel, Atanas Kiryakov, and Damyan Ognyanov. Ontology versioning and change detection on the web. In *Proceedings of EKAW 2002*, pages 197–212, Berlin, 2002. Springer–Verlag.

[KHA05] V. Kashyap, T. Hongsermeier, and S. Aronson. Can semantic web techno-logies enable translational medicine? (or can translational medicine help en-rich the semantic web?). Technical Report No. CIRD-20041027-01, W3C & Partners Healthcare System, 2005.

[Lev66] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Cybernetics Control Theory*, 10:707–710, 1966.

[LTE+06]   L. Laera, V. Tamma, J. Euzenat, T. Bench-Capon, and T. R. Payne. Reaching agreement over ontology alignments. In *Proceedings of 5th International Semantic Web Conference (ISWC 2006)*. Springer-Verlag, 2006.

[MBG+02]   C. Masolo, S. Borgo, A. Gangemi, N. Guarino, A. Oltramari, and L. Schneider. Sweetening ontologies with DOLCE. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management*, 2002.

[MPL06]   Diana Maynard, Wim Peters, and Yaoyong Li. Metrics for evaluation of ontology-based information extraction. In *Proceedings of EON 2006 Workshop — Evaluation of Ontologies for the Web*, 2006.

[MPSC05]   Diana Maynard, Wim Peters, Marta Sabou, and Philipp Cimiano. Prototypes of language dependent tools for ontology evaluation (D2.3.6). Deliverable 236, Knowledge Web, 2005.

[MPSC06]   D. Maynard, W. Peters, M. Sabou, and P. Cimiano. Prototypes of language dependent tools for evaluation. Technical Report D2.3.6, KnowledgeWeb Deliverable, 2006.

[MS04]   Alexander Maedche and Steffen Staab. Ontology learning. In S. Staab and R. Studer, editors, *Handbook on Ontologies*, chapter 9, pages 173–190. Springer–Verlag, 2004.

[MSR06]   M. Morge, Y. Secq, and J.-C. Routier. Formal framework for inter-agents dialogue to reach an agreement about a representation. In *Proceedings of the Workshop on Computational Models of Natural Argument (CMNA 2006)*, 2006.

[NDKH07]   Vít Nováček, Maciej Dabrowski, Sebastian Ryszard Kruk, and Siegfried Handschuh. Extending community ontology using automatically generated suggestions. In *Proceedings of FLAIRS 2007*. AAAI Press, 2007. In press.

[NJQ05]   G. Cheng N. Jian, W. Hu and Y. Qu. Falcon-AO: Aligning Ontologies with Falcon. In *Proceedings of the K-CAP Workshop on Integrating Ontologies*, 2005.

[NM01]   N.F. Noy and M.A. Musen. Anchor-PROMPT: Using non-local context for semantic Matching. In *Proceedings of the Workshop on Ontologies and Information Sharing at the International Joint Conference on Artificial Intelligence (IJCAI)*, 2001.

[NM04]   Lyndon Nixon and Malgorzata Mochol. Prototypical business use cases (D1.1.2). Deliverable 112, Knowledge Web, 2004.

[O. 04]   O. Corcho and A. Gmez-Prez and R. Gonzlez-Cabero and MC Surez-Figueroa. ODEval: a Tool for evaluating RDF(S), DAML+OIL, and OWL Concept Taxonomies. In *First IFIP Conference on Artificial Intelligence Applications and Innovations (AIAI2004)*, 2004.

[OVMS04]   Daniel Oberle, Raphael Volz, Boris Motik, and Steffen Staab. An extensible ontology software environment. In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, International Handbooks on Information Systems, chapter III, pages 311–333. Springer, 2004.

[PCTB06]   T. Polajnar, H. Cunningham, V. Tablan, and K. Bontcheva. Report: Controlled language ie component version 1 (d2.2.1). Deliverable 221, SEKT, 2006.

[PGPM99]   H. S. Pinto, A. Gomez-Perez, and J. P. Martins. Some issues on ontology integration. In *11th Proceedings of the Workshop on Ontologies and Problem Solving Methods during IJCAI-99*, 1999.

[Rod95]   J. Roddick. A survey of schema versioning issues for database systems. *Information and Software Technology*, 37:383–393, 1995.

[Sab05]   Marta Sabou. *Building Web Service Ontologies*. PhD thesis, Vrije Universiteit Amsterdam, 2005.

[Sch97]   D. J. Schultz. IEEE standard for developing software life cycle processes. Technical Report 1074-1997, IEEE, 1997.

[SEA⁺02]   Y. Sure, M. Erdmann, J. Angele, S. Staab, R. Studer, and D. Wenke. OntoEdit: Collaborative Ontology Development for the Semantic Web. In *1st International Semantic Web Conference (ISWC2002)*, Sardinia, 2002. Springer.

[SGPM05]   P. Shvaiko, F. Giunchiglia, P. Pinheiro da Silva, and D. McGuinness. Web explanations for semantic heterogeneity discovery. In *Proceedings of ESWC 2005*, pages 303–317, 2005.

[SMR05]   N. Silva, P. Maio, and J. Rocha. An Approach to Ontology Mapping Negotiation. In *Proceedings of the Workshop on Integrating Ontologies*, 2005.

[SR03]   N. Silva and J. Rocha. MAFRA Semantic Web Ontology MApping FRAmework . In *Proceedings of the Seventh Multi-Conference on Systemics, Cybernetics and Informatics*, 2003.

[SS04]   S. Staab and R. Studer, editors. *Handbook on Ontologies*. International Handbooks on Information Systems. Springer-Verlag, 2004.

[ST06]   Umberto Straccia and Raphaël Troncy. Towards Distributed Information Retrieval in the Semantic Web: Query Reformulation Using the oMAP Framework. In *Proceedings of the 3rd European Semantic Web Conference (ESWC 06)*, pages 378–392, Budva, Montenegro, June 11-14, 2006.

[Ste02]   M. Stevenson. Combining disambiguation techniques to enrich an ontology. In *Proceedings of ECAI-02 workshop – Machine Learning and Natural Language Processing for Ontology Engineering*, Lyon, France, 2002.

[Sto04]   L. Stojanovic. *Methods and Tools for Ontology Evolution*. PhD thesis, University of Karlsruhe, 2004.

[TPCB06]   V. Tablan, T. Polajnar, H. Cunningham, and K. Bontcheva. User–friendly ontology authoring using a controlled language. In *Proceedings of LREC 2006 - 5th International Conference on Language Resources and Evaluation*. ELRA/ELDA Paris, 2006.

[vDBD⁺05]   J. van Diggelen, R. Beun, F. Dignum, R. van Eijk, and J.-J. Meyer. A decentralized approach for establishing a shared communication vocabulary. In *Proceedings of the AMKN*, 2005.

[VEK⁺05]   M. Völkel, C. F. Enguix, S. R. Kruk, A. V. Zhdanova, R. Stevens, and Y. Sure. SemVersion – versioning RDF and ontologies (D2.3.3). Deliverable 233, Knowledge Web, 2005.

[VG06]   Max Völkel and Tudor Groza. SemVersion: RDF-based ontology versioning system. In *Proceedings of the IADIS International Conference WWW/Internet 2006 (ICWI 2006)*, 2006.

[VPKV04]   A. G. Valarakos, G. Paliouras, V. Karkaletsis, and G. Vouros. Enhancing ontological knowledge through ontology population and enrichment. In *Proceedings of EKAW 2004*, pages 144–156, Berling, 2004. Springer–Verlag.

[ZB05]   S. Zhang and O. Bodenreider. Alignment of multiple ontologies of anatomy: Deriving indirect mappings from direct mappings to a reference. In *Proccedings of AMIA 2005 Annual Symposium*, pages 864–869. American Medical Informatics Association, 2005.

[ZKHF05]   A. V. Zhdanova, R. Krummenacher, J. Henke, and D. Fensel. Community–driven ontology management: Deri case study. In *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence*, pages 73–79. IEEE Computer Society Press, 2005.

Related Deliverables

The work presented here is directly related to the following deliverables:

Project	Number	**Title** and relationship
KW	D1.1.2	**Prototypical business use cases** studies the needs of the industry using elaborated use cases of semantics-enabled business solutions.
KW	D1.2.3	**Methods for ontology evaluation** provides a survey of various (either human-oriented or automatic) methods of ontology evaluation.
KW	D1.2.4	**Architecture of the semantic web framework** presents a framework defining the structure and funcionalities of typical Semantic Web applications' components.
KW	D2.3.3v2	**Full RDF versioning system** describes the implementation and application possibilities of an RDF-based versioning system.
KW	D2.3.5v2	**Consensus Making Environment** presents a mechanism of collaborative ontology development with incorporated versioning.
KW	D2.3.6	**Prototypes of language dependent tools for ontology evaluation** contains initial notes on implementation of ontology evaluation tools.
KW	D2.3.7	**Report on negotiation/argumentation techniques among agents complying to different ontologies** introduces a technique used for computation of agreed ontology alignment among agents with different preferences.
SEKT	D1-7-1	**Ontology generation from scratch-software V1.0** offers a general survey of automatic ontology extraction methods.
SEKT	D3-3-1	**Data-driven Change Discovery V1** presents the Text2Onto ontology learning framework, the methods utilised and a mechanism for data-driven change tracking.