NeOn: Lifecycle Support for Networked Ontologies

Integrated Project (IST-2005-027595)

Priority: IST-2004-2.4.7 — “Semantic-based knowledge and content systems”

D3.3.1: Matching ontologies for context

Deliverable Co-ordinator: **Jérôme Euzenat**

Deliverable Co-ordinating Institution: **INRIA**

Other Authors: **Mathieu d’Aquin (OU), Marta Sabou (OU), Antoine Zimmermann (INRIA)**

This deliverable establishes infrastucture compoments that help contextualising ontologies by finding relations that it has with other (context) ontologies. We present the Alignment Server as the infrastructure for providing this contextualisation operation. We also explain how the context expressed as such can be used in order to find relation between different ontologies (matching).

Document Identifier:	NEON/2006/D3.3.1/0.8	Date due:	February 16th, 2006
Class Deliverable:	NEON EU-IST-2005-027595	Submission date:	February 26, 2006
Project start date	March 1, 2006	Version:	0.8
Project duration:	4 years	State:	Final
		Distribution:	Public

NeOn Consortium

This document is part of a research project funded by the IST Programme of the Commission of the European Communities, grant number IST-2005-027595. The following partners are involved in the project:

Open University (OU) – Coordinator Knowledge Media Institute – KMi Berrill Building, Walton Hall Milton Keynes, MK7 6AA United Kingdom Contact person: Martin Dzbor, Enrico Motta E-mail address: {m.dzbor, e.motta}@open.ac.uk	**Universität Karlsruhe – TH (UKARL)** Institut für Angewandte Informatik und Formale Beschreibungsverfahren – AIFB D-76128 Karlsruhe Germany Contact person: Peter Haase E-mail address: pha@aifb.uni-karlsruhe.de
Universidad Politécnica di Madrid (UPM) Campus de Montegancedo 28660 Boadilla del Monte Spain Contact person: Asunción Gómez Pérez E-mail address: asun@fi.ump.es	**Software AG (SAG)** Uhlandstrasse 12 64297 Darmstadt Germany Contact person: Walter Waterfeld E-mail address: walter.waterfeld@softwareag.com
Intelligent Software Components S.A. (ISOCO) Calle de Pedro de Valdivia 10 28006 Madrid Spain Contact person: Richard Benjamins E-mail adress: rbenjamins@isoco.com	**Institut 'Jožef Stefan' (JSI)** Jamova 39 SL–1000 Ljubljana Slovenia Contact person: Marko Grobelnik E-mail address: marko.grobelnik@ijs.si
Institut National de Recherche en Informatique et en Automatique (INRIA) ZIRST – 665 avenue de l'Europe Montbonnot Saint Martin 38334 Saint-Ismier France Contact person: Jérôme Euzenat	**University of Sheffield (USFD)** Dept. of Computer Science Regent Court 211 Portobello street S14DP Sheffield United Kingdom Contact person: Hamish Cunningham
Universität Kolenz-Landau (UKO-LD) Universitätsstrasse 1 56070 Koblenz Germany Contact person: Steffen Staab E-mail address: staab@uni-koblenz.de	**Consiglio Nazionale delle Ricerche (CNR)** Institute of cognitive sciences and technologies Via S. Marino della Battaglia 44 – 00185 Roma-Lazio Italy Contact person: Aldo Gangemi E-mail address: aldo.gangemi@istc.cnr.it
Ontoprise GmbH. (ONTO) Amalienbadstr. 36 (Raumfabrik 29) 76227 Karlsruhe Germany Contact person: Jürgen Angele E-mail address: angele@ontoprise.de	**Asociación Española de Comercio Electrónico (AECE)** C/lcalde Barnils, Avenida Diagonal 437 08036 Barcelona Spain Contact person: Jose Luis Zimmerman E-mail address: jlzimmerman@fecemd.org
Food and Agriculture Organization of the United Nations (FAO) Viale delle Terme di Caracalla 00100 Rome, Italy Contact person: Marta Iglesias E-mail address: marta.iglesias@fao.org	**Atos Origin S.A. (ATOS)** Calle de Albarracín, 25 28037 Madrid Spain Contact person: Tomás Pariente Lobo E-mail address: tomas.parientelobo@atosorigin.com

Changes

Version	Date	Author	Changes
0.1	05.09.2006	Jérôme Euzenat	initial version
0.2	05.12.2006	Jérôme Euzenat	drafted introduction and structure
0.3	09.01.2007	Jérôme Euzenat	filled various part with adequate discussions
0.4	30.01.2007	Jérôme Euzenat	Integrated general assembly comments
0.5	05.02.2007	Marta Sabou	Improved Chaper 5
0.6	13.02.2007	Jérôme Euzenat	Improved Chaper 3 & wrote executive summary
0.7	15.02.2007	Jérôme Euzenat & Marta Sabou	Various improvements
0.8	26.02.2007	Jérôme Euzenat	Implemented quality control comments and renumbered as D3.3.1

Executive Summary

Ontologies and other resources on the web are always designed in a particular context that is not often explicit in the resource itself. This renders resources ambiguous and can lead their exploitation to provide incomplete, or worse incorrect, results. This problem is even more acute when resources from heterogeneous provenance are linked.

In the NeOn project, context can be expressed within a network of ontologies through relationships between ontologies. Such relationships would provide the missing axioms that specify the meaning of knowledge further and help considering them in their context.

These relations must be established from the initial resources which will be linked to contextual resourcs. This can be achieved through the use of ontology matching techniques. Moreover, this knowledge of context can help matching ontologies.

The present deliverable aims at (1) explaining these notions, and (2) providing an account of the tools developed within the project in order to deal with these functions. There are two specific tools presented here: the Alignment Server dedicated to storing and sharing alignment information between ontologies which can be used as a source of contextualisation and a specific context-based matching algorithm that takes advantage of ontologies available on the web.

We thus introduce the operation of contextualisation (and decontextualisation) as expressing (or dropping) the context as relationships between ontologies. It can also be seen as part of introducing ontologies within a particular networked ontology. We present the available ontology alignment format as an instantiation of the Mapping metamodel developed as part of the NeOn networked ontology model.

The Alignment Server is a component based on the existing Alignment API that provides matching services to clients. It is reachable through human users, agents and web services. The Alignment Server provides support for storing and sharing alignments in standard formats. It can thus be used by context-based matchers in order to compute an alignment between some ontology and a general purpose resource. It is also designed for sheltering new matching tools as the ones developed in NeOn, but not exclusively. In the future, it will be integrated within the NeOn toolkit.

Finally, we present a new context-based matching tool that takes advantage of multiple recontextualisations from the resources to the many ontologies available on the web. The basic idea is to automatically discover the common context of these ontologies under the form of a third ontology containing the relevant background knowledge. This technique has been implemented using Swoogle to find ontologies relating the entity to be matched. This ontology will then provide the relations between these entities. This promising technology has been experimented with FAO resources (AGROVOC) and we plan to use it for providing relationships between the many FAO resources.

Contents

Chapter 1

Contextualising ontologies through matching

Domain ontologies are designed to be applied in a particular context and use terms in a sense that is relevant to this domain, e.g., *Ontology* in computer science, and which is not related to similar concepts in other domains. They do not fully specify concepts because part of their specification is implicit from the context. We will use here the word “constraint” for this specification in a broad sense: any axiom constrains the meaning of the terms it uses.

NeOn has in its core the ambitious scenario that ontologies are developed in an open environment in a distributed fashion. This means that ontologies will be used in settings that are not those that have led to their design. In order to use them properly, it is necessary to be able to identify this context so that appropriate actions can be taken: either not using them or adapting them to the new context.

The context of some knowledge or ontology is given by additional knowledge in which or in the perspective of which this knowledge has been elaborated. This knowledge can vary in nature and expression. For instance, in work package 2, the context of elaboration of ontologies is expressed a argumentative structures about ontology design rationale. In work package 3 [Haase *et al.*, 2006a], the context of ontologies is prominently placed in the framework of networked ontologies: the context of an ontology is given by the network of other ontologies with which it is related.

From this definition, a pair of dual operation can be associated with context (see Figure 1.1): contextualisation and decontextualisation. Contextualisation or recontextualisation is the action of finding the relations of an ontology with other ontologies which express its context. Decontextualisation, as the opposite, extracts one particular ontology from its context. These operation can be combined, for instance, if someone wants to transfer one ontology from a domain to another one, by first decontextualising it and recontextualising it to another domain.

As can be considered from this brief description, contextualising an ontology is a matter of matching it to other

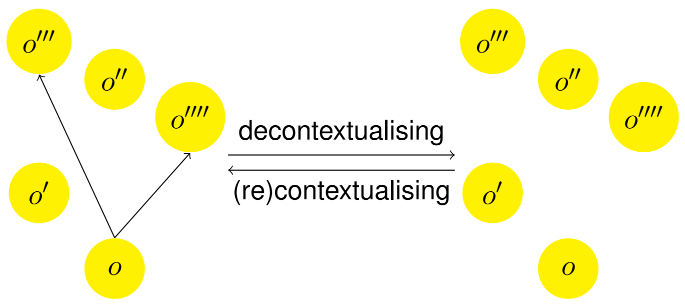


Figure 1.1: Contextualising/Decontextualising ontologies in the framework of networked ontologies

ontologies which will provide its context. For that purpose ontology matching technologies can be used. So we would like to provide support for contextualising ontologies through ontology matching. Moreover, recontextualising ontologies can be succesfully used for helping the process of matching an ontology into another one. Thus, we also aim at taking into account the contextualisation operation for matching.

The present deliverable is a prototype report that describes the work that has been carried out for developping technologies achieving these operation of contextualising and using it for matching. After briefly introducing the expression of contextual relation within networked ontologies (Chapter 2) and matching technology (Chapter 3), it describes the technology that has been developed as an Alignment Server in order to provide applications with the ontology matching support enabling contextualising ontologies (Chapter 4). It also describes the work and experiments carried on for matching ontologies by recontextualising them (Chapter 5).

This work takes advantage of the previous work in work packages 1 and 3. It shows a practical instantiation of the NeOn networked ontology model using the ontology and alignment (a.k.a. mapping) components. It is also partially experimented on NeOn use cases resources: the AGROVOC thesaurus.

Chapter 2

Expressing context through alignments within networked ontologies

In this chapter, we recast the intuition presented before in the context of networked ontologies. In particular we explain in what sense context are networks of ontologies (§2.1), what are networked ontologies (§2.2) and how they can be represented within NeOn's model of networked ontologies (§2.3).

2.1 Context as network

NeOn has in its core the ambitious scenario that ontologies are developed in an open environment in a distributed fashion. This means that ontologies will be used out of their initial definition context.

Context is traditionaly supplemental knowledge that allows interpreting, or interpreting better, some other pieces of knowledge. For an ontology, this is knowledge that helps using it. In our first deliverable [Haase *et al.*, 2006a], the network of ontologies with which an ontology is related is its primary form of context. Indeed, this network provides additional knowledge, usually related to the knowledge within the ontology. According to our generic definition of context, this should allow modifying the interpretation of the ontology.

2.2 Models of networked ontologies

In the present deliverable, we are concerned with the representation of context as a networked ontology. We must thus start from the definition of networked ontologies given by Work package 1 (and most notably Deliverable D1.1.1 [Haase *et al.*, 2006b]).

There are roughly two constructions from the NeOn model for networked ontologies that we will use for expressing context in this deliverable: Ontologies and mappings (or alignments). The NeOn model already offers the metamodel for these two kinds of entities. We will thus take advantage of these for representing context.

The metamodel also provides a specific instantiation of the Ontology metamodel into OWL. It however does not provide such a concrete instantiation for the Mapping metamodel. We will use here a very general definition of alignment that will be used in this deliverable and associated software.

2.3 Alignments within networked ontologies

The so-called Mapping metamodel is reproduced in Figure 2.1. It covers the basic elements found in an alignment as defined in [Bouquet *et al.*, 2004b; Euzenat and Shvaiko, 2007]: an alignment between two ontologies is a set of correspondences, e.g., equivalence, subsumption, relating the elements of these ontologies, e.g., concepts, properties, formulas.

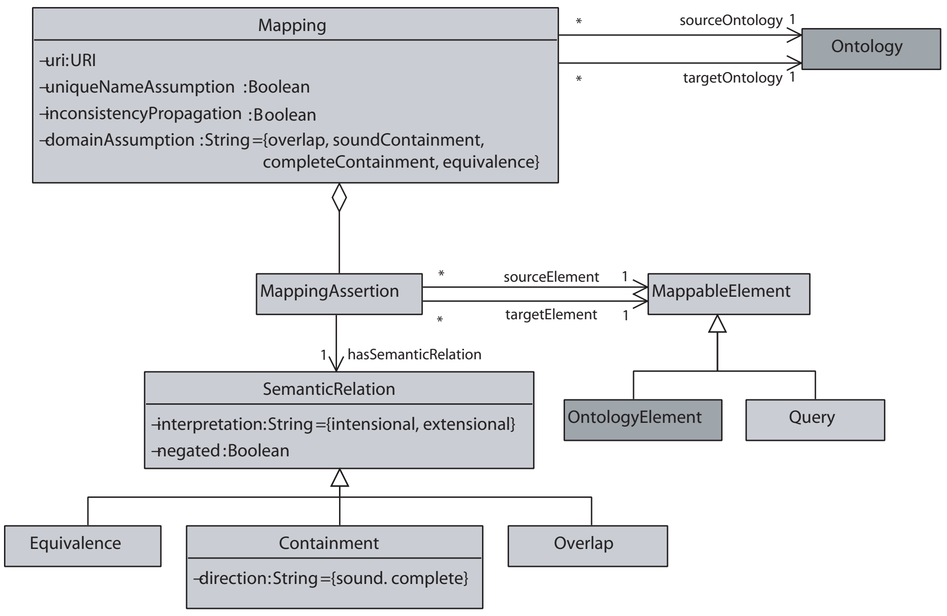

Mapping
-uri:URI
-uniqueNameAssumption :Boolean
-inconsistencyPropagation :Boolean
-domainAssumption :String={overlap, soundContainment, completeContainment, equivalence}
sourceOntology
targetOntology
Ontology
MappingAssertion
sourceElement
targetElement
MappableElement
hasSemanticRelation
SemanticRelation
-interpretation:String={intensional, extensional}
-negated:Boolean
OntologyElement
Query
Equivalence
Containment
-direction:String={sound. complete}
Overlap

Figure 2.1: Mapping metamodel for NeOn

We give here a precise definition of alignment as it is generally accepted:

Definition 1 (Alignment) *Given two ontologies o and o' over ontology languages L and L', a set of alignment relations Θ and a confidence structure Ξ, an* alignment *is made up of a set of correspondences*

$$\langle id, e, e', r, n \rangle,$$

such that

- *id is a unique identifier of the given correspondence;*
- *$e \in Q_L(o)$ and $e' \in Q'_{L'}(o')$ are ontology entities;*
- *$r \in \Theta$ is the relation between these entities; and*
- *$n \in \Xi$ is the confidence in the correspondence.*

This definition clearly matches what is modelled in the mapping meta model and is the basis for the Alignment format and its implementation as the Alignment API. We will thus take the structure of the Alignment API as an instantiation of the mapping metamodel (in fact this format is already very generic and can apply to many kinds of ontologies).

The Alignment API [Euzenat, 2004] has been designed to help developing applications based on alignments. It has been developed in the aim of manipulating a standard alignment format for sharing among matching systems, but it provides the features required for sharing them more widely. The API is a JAVA description of tools for accessing alignments in the format presented above.

The Alignment API can be used in conjunction with an ontology language API (the OWL-API is currently available, but other instantiation could be based on totally different languages). This implementation offers the following services:

- Computing and representing alignments;
- Piping alignments algorithms (for improving an existing alignment);
- Manipulating (trimming and hardening) and combining (merging, composing) alignments;
- Generating “mediators” (transformations, axioms, rules in format such as XSLT, SWRL, OWL, C-OWL, WSML);

– Comparing alignments (like computing precision and recall or a symmetric distance with regard to a particular reference alignment).

The API also provides the ability to compose matching algorithms and manipulating alignments through programming. Part of the interface of the API is presented in Table 4.1. The API can be used for producing transformations, rules or bridge axioms independently from the algorithm that produced the alignment. Since its definition, several matching systems have been developed within this API (OLA, oMap) and more of them are able to generate its format (FOAM, Prompt, Falcon, etc.).

This will allow us to consider OWL ontologies and alignments as networked ontologies and to offer software for dealing with contexts in the framework of networked ontologies.

Before turning to the software that are goal of the present deliverable, we will recall some basics about matching algorithms.

2.4 Summary

Contexts as described in Deliverable D1.1.1 can be expressed within the actual networked ontology model. We currently consider implementing this notion through the existing Alignment API and an ontology API.

We propose, in Chapter 4, an architecture for taking advantage of this.

Chapter 3

Techniques for matching networked ontologies

Matching two (or more) ontologies consists of finding an alignment between these ontologies. Hence, matching is the basic operation for (re)contextualising ontologies and more generally integrating an ontology within a network of ontologies.

This a very important task because it helps restoring interoperability, but a difficult one because it is very hard to find these correspondences in the general case: independently built ontologies can vary a lot on the terminology they use and the way they model the same entities.

There have been many techniques for ontology matching [Euzenat and Shvaiko, 2007]. We give a very brief overview of these techniques and their justifications within the NeOn project and contrast them with the technique that is presented in Chapter 5 which is specifically related to context.

3.1 Matching needs for context

In the framework of NeOn, many resources are not ontologies per se and have been designed for specific purposes. In particular the resources developed by FAO (like AGROVOC or other databases) are relatively self-contained resources which have been designed for being used in the FAO context. Linking these resources together in a network of ontology is appealing but not straightforward. The design context of these resources which is implicit should be rendered explicit before linking theses resources together.

It is part of the NeOn ambition to provide tools that facilitate this recontextualisation of resources.

The task of creating a network of ontologies from FAO resources thus requires:

- (re)contextualising these resources;
- using the context for relating these resources.

Both tasks can be achieved through matching:

- by matching the resources to external, generaly well established, resources, and
- by matching the resources together through the result of the previous operation.

The process is depicted in Figure 3.1: the two initial resources are matched to existing ontologies in order to link them to ontological axioms which are then considered as context. Then the resulting networked ontologies can be matched by using the so provided context. The results of this second phase will provide an alignment within the networked ontologies. The first phase is usually carried out with traditional matching algorithms. This is especially true when the resources are database entries or loosely formalised thesauri: in this case, the matching operation will be based on the names of entries and their textual annotations. Once the links with ontologies have been provided, more sophisticated strategies can be considered. This is especially true when the contextualising phase has linked the resources to the same ontology.

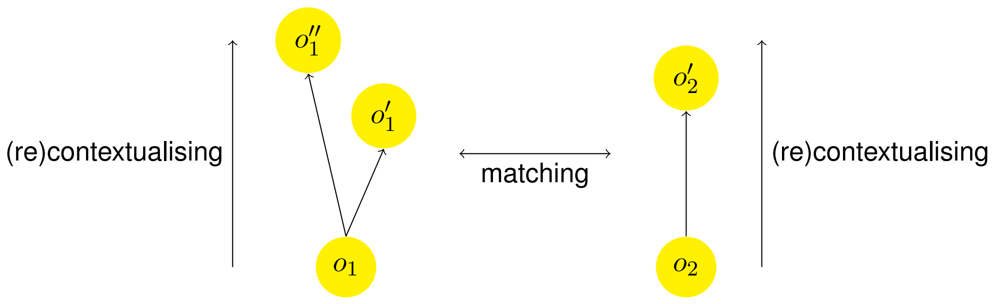


Figure 3.1: Matching ontologies through contextualisation

We consider below these two sort of matching techniques.

Matching each of the ontologies to a common third ontology put these two ontologies in the same context. This should be easier than the direct matching because: (i) upper level ontologies having a broad coverage are more prone to provide matching super concepts of these ontologies, (ii) there are more chances that these alignments already exists due to the availability of these ontologies. Such alignments, are typically those that could be expected to be found in an Alignment Server.

Once the alignments between the two ontologies and the upper-level one are obtained, it is easier to partition the ontology matching task in smaller matching problems because the matching will identify the concepts having common super-concepts and can take advantage of the exclusion assertions typically found in the upper-level ontologies.

3.2 General matching systems

There have already been many reviews of ontology matching algorithms [Rahm and Bernstein, 2001; Wache *et al.*, 2001; Kalfoglou and Schorlemmer, 2003; Euzenat *et al.*, 2004; Euzenat and Shvaiko, 2007] so we will be brief and refer the reader to these other presentations. Ontology matching consists of generating an alignment from two (or more) ontologies. There are many different features of ontologies that are usually used for providing matching:

terminological techniques are based on the text found within ontologies for identifying ontology entities (labels), documenting them (comments) or other surrounding textual sources (related element labels). These techniques come from natural language processing and information retrieval. They can use the string structure themselves (string distances), the ontology as corpus (statistical measures based on the frequency of occurence of a term) or external resources (such as dictionaries).

structural techniques are based on the relations between ontology entities. These can be relations between entities and their attributes, including constraints on their values, or relations with other entities. These relation take advantage of type comparison techniques or more elaborate graph techniques (tree distances, path matching, graph matching).

extensional techniques compare the extension of entities. These extensions can be made of other entities (instances) as well as related resources (indexed documents). They differ depending on if the two ontologies share resources (e.g., they index the same set of documents) or not (in which case a similarity between the extensions may be established). These techniques can come from data analysis and statistics.

semantic techniques are based on the semantic definition of ontologies. They use theorem provers for finding consequences of a particular alignment. This can be used for expanding the alignment or, on the contrary, for detecting conflicting correspondences.

Of course, most of the systems combine several techniques in order to improve their results. The techniques can be combined by aggregating distance results [van Hage *et al.*, 2005], by using selection functions for choosing which one to use in the present case [Jian *et al.*, 2005; Tang *et al.*, 2006], or by deeply involving them all in global distance computation [Euzenat and Valtchev, 2004; Melnik *et al.*, 2002].

Moreover, there is a difference when training sets are available or not (this is most often useful when a matching algorithm is needed for recognising instances). In this case, one can apply classical techniques such as Bayes learning, vector support machines or decision trees.

Some toolboxes developed by NeOn partners already offer a number of these techniques like FOAM [Ehrig, 2007] or the Alignment API [Euzenat, 2004]. Other have experience on matching ontologies in specific contexts [Gangemi *et al.*, 2003; Lopez *et al.*, 2006].

In the context of NeOn, the goal is really to find the relationship between the resources. Moreover these resources are not always ontologies: they can be thesauri or databases. In this case, the structure and the semantics of resources are not the most usable features of these resources. So the main techniques to be used are terminological techniques and extension based techniques.

3.3 Context-based matching

Context-based matching is a specific kind of operation that take advantage of the ontology context in order to provide a more accurate match. We pretend that it can only occur in networked ontologies. While the idea of using context for matching is relatively recent, we can distinguish between several ways to achieve this depending on the kind of resources that express (or are used to express) context and their degree of specialisation. This is illustrated in Figure 3.2.

Figure 3.2 displays two resources that are contextualised against two other resources. We will see below that most of the time this hypothesis is simplified by contextualising the resources against one common resource. Example of non formal specialised resources include the ASFA thesaurus[1], non formal general purpose resources include WordNet[2], formal specific resources include the Foundational Model of Anatomy[3], and formal general purpose resources include DOLCE[4].

By using specialised resources (e.g., the Formal Model of Anatomy or the Unified MLS in medicine), one can be sure that the concepts in the contextualised resources can be matched accurately to their corresponding concepts in the ontology. However, by using more general resources there are more probability that an alignment already exists and can be exploited right away.

By using formal resources (e.g., DOLCE or the Formal Model of Anatomy) it is possible to reason within or across these formal models in order to deduce the relation between two terms. By using informal resources (e.g., UMLS or WordNet), it is possible to extend the set of senses that are covered by a term and to increase the number of terms that can express these concepts. There is thus more opportunity to find the match between two terms.

The work on context-based matching is at its beginning. We describe here the first attempts at achieving it. We can bet that there will be many further developments. Such an experiment has been carried out in the context of NeOn and is reported in Chapter 5.

3.3.1 Context-based matching with informal resources

An interesting use of contextualisation is in semantic matchers as presented in [Giunchiglia and Shvaiko, 2003]. Such matchers apply only to formalised ontologies with explicit axioms. However, they are bootstraped by a contextualisation phase that uses informal resources (typically WordNet). WordNet is used for finding initial relationships between terms in both ontologies based on their corresponding WordNet synsets. For instance, WordNet will find that prawn and shrimp are synonyms in one of the WordNet senses of prawn.

[1]Aquatic Sciences and Fisheries Abstracts, http://www.fao.org/fi/asfa/asfa.asp
[2]http://wordnet.princeton.edu
[3]http://sig.biostr.washington.edu/projects/fm/AboutFM.html
[4]http://www.loa-cnr.it/DOLCE.html

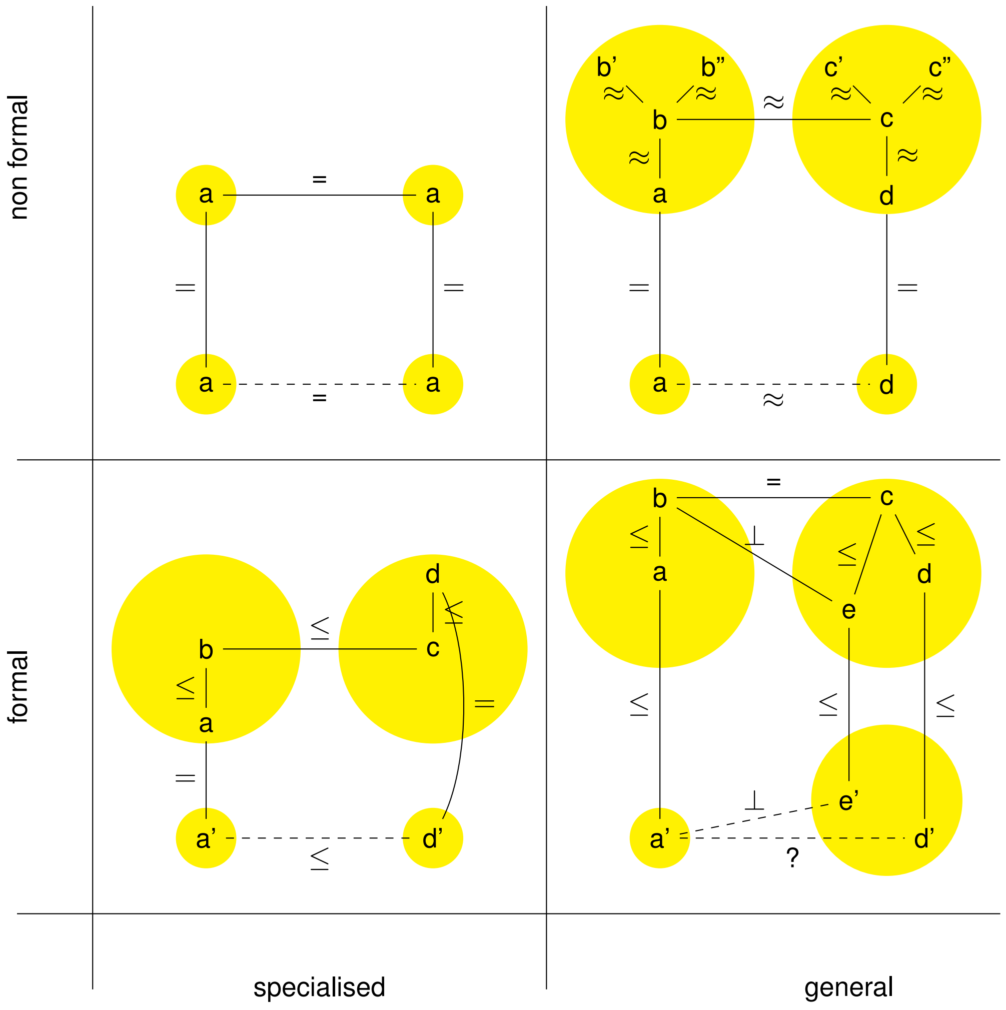


Figure 3.2: Classification of matching with contexts (dashed lines represent the infered correspondences).

The matcher then “amplifies” these initial alignments by using the formal knowledge in the ontologies. A semantic matcher will consider the set of axioms of both ontologies combined with those relevant axioms provided by WordNet (e.g., *Prawn* $=$ *Shrimp*). Then it will use some theorem prover for amplifying this correspondence. For instance, if *Shrimp* $\leq$ *Sea food* it will deduce that *Prawn* $\leq$ *Sea food*.

This amplification can also be used in order to check the consistency of the merged ontology.

3.3.2 Context-based matching with formal resources

Recontextualising ontologies can typically be achieved by finding a mapping between these ontologies and a common upper-level ontology (e.g., Cyc [Lenat and Guha, 1990], Suggested Upper Merged Ontology (SUMO) [Niles and Pease, 2001] or Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) [Gangemi *et al.*, 2003]) that is used as external source of common knowledge.

An experiment has been carried out within the WonderWeb project by expressing Fishery resources (such as AGROVOC, AFSA and FIGIS) within the DOLCE upper level ontology [Gangemi, 2004]. The goal was not at that time to provide networked ontologies, but to merge these resources into a common Core Ontology of Fisheries. It has involved transforming manually the resources into lightweight ontologies expressed with respect to DOLCE and then using reasoning facilities for detecting relations and inconsistency between entities of this ontology.

Other work involved matching with a more specialised ontology that is known to cover the ontologies to match. Suppose we want to match the anatomy part of the CRISP[5] database to the anatomy part of the MeSH[6] thesaurus. In this case, [Aleksovski *et al.*, 2006] uses the FMA ontology[7] as background knowledge which gives the context to the matching task. This approach works in two steps as follows:

Contextualising (also called Anchoring) is matching ontologies CRISP and MeSH to the background ontology FMA. This can be done by using any available method, such as string-based techniques or structure-based techniques.

Deriving relations is the process of (indirect) matching of CRISP and MeSH by using the correspondences discovered during the anchoring step. Since concepts of CRISP and MeSH become a part of the background ontology FMA via anchors, checking if these concepts are related, can be therefore performed by using a reasoning service (in the same fashion as discussed in §3.3.1) in the background ontology.

Intuitively, combining the anchor relations with the relations between the concepts of the reference ontology is used to derive the relations between concepts of CRISP and MeSH.

For example, the concept of brain from CRISP, denoted by *Brain*$_{CRISP}$, could be easily anchored to the concept brain of FMA, denoted by *Brain*$_{FMA}$. Similarly, the concept of head from MeSH, denoted by *Head*$_{MeSH}$, could be anchored to a background knowledge concept *Head*$_{FMA}$. In the reference ontology FMA there is a *part of* relation between *Brain*$_{FMA}$ and *Head*$_{FMA}$. Therefore, we can derive that *Brain*$_{CRISP}$ is less general than *Head*$_{MeSH}$.

Since the domain specific ontology provides the context for the matching task, the concept of *Head* was correctly interpreted, meaning the upper part of the human body, instead of, for example, meaning a chief person. Note this is not so straightforward as can be shown by replacing FMA by WordNet: in WordNet the concept of *Head* has 33 senses (as a noun). Finally, once the context of the matching task has been established, as our example shows, a number of heuristics, such as string-based techniques, can perform well in the recontextualising step.

[5]http://crisp.cit.nih.gov/

[6]http://www.nlm.nih.gov/mesh/

[7]http://sig.biostr.washington.edu/projects/fm/AboutFM.html

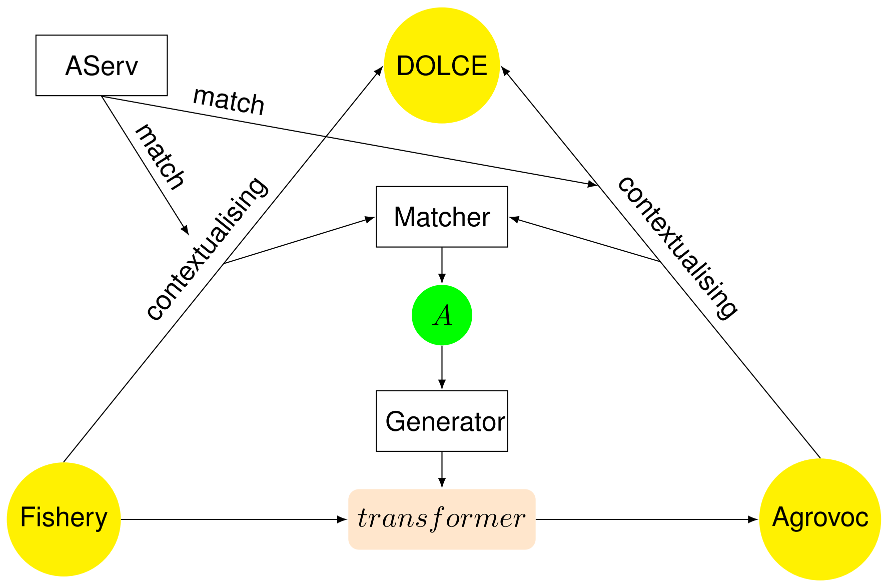


Figure 3.3: Matching ontologies after contextualising them in a common upper level ontology.

3.3.3 Hybrid approach

Of course, the four approaches that we presented in Figure 3.2 can be freely mixed in order to achieve better accuracy or to find more correspondences between the ontologies. For instance, using the OntoWordNet resource[8] which is based on DOLCE allows to benefit from both formal and informal resources.

3.4 Summary

We have seen the particularly interesting use of ontology matching for contextualising ontologies and the role of context in matching. There are many different matching techniques which can be used for contextualising ontologies. We do not commit to one particular technique but leave this open to future experiments in the context of NeOn use cases. If we want to deal with contextualisation in NeOn in a open and extensible way, we need to offer a framework in which contextualisation can occur naturally both for matching purposes and for other purposes. Thus we need to provide a a generic architecture for computing and sharing these alignments.

For that purpose, we propose to use an Alignment Server based on the already developed Alignment API. This server will provide the opportunity to take advantage of existing relations between ontologies and to integrate matching systems for contextualising ontologies. This is illustrated by Figure 3.3.

Moreover, a new research trend, uses recontextualisation as understood here in order to improve matching performances. We go further in that direction in Chapter 5 by providing results of a new such matcher implemented for NeOn.

We will present in the next chapter the proposed architecture for this alignment server. The last chapter will present a new approach for using context in matching by taking advantage of the many ontologies available on the semantic web.

[8] http://www.loa-cnr.it/ontologies/OWN/OWN.owl

Chapter 4

Architecture for alignment discovery and exploitation

In order to support networked ontology users in dealing with contexts and, in particular, recontextualising, we propose the use of an Alignment Server which is able to both store alignment resources to be used for contextualising and matching techniques for generating these alignments.

The goal of the Alignment Server is that different actors can share the available alignments and methods for finding alignments. Such a server will enable to match ontologies, store the resulting alignment, store manually provided alignments, extract merger, transformer, mediators from those alignments.

We present here the architecture of the Alignment Server and its integration within the NeOn toolkit (§4.1). We then present its first functions (§4.2) and finally display the working of the Alignment Server on some available resources (§4.3).

4.1 Architecture

The Alignment server is built around the Alignment API developed by INRIA (see Figure 4.1). It will thus provide access to all the features of this API. The server ensures the persistence of the alignments through the storage of these in a relational database (we currently use MySQL, but any JDBC compliant database management system should work). The access to the API is achieved through a protocol which extends the one designed in [Euzenat *et al.*, 2005b]. Plug-ins allow the remote invocation of the alignment server. At the time of writting, three plug-ins are available for the server:

– HTTP/HTML plug-in for interacting through a browser;
– JADE/FIPA ACL for interacting with agents;
– HTTP/SOAP plug-in for interacting as a web service.

In the context of NeOn, our goal is that the server is available:

at design time through a plug-in to the NeOn toolkit: It can be integrated either as an Eclipse plug-in that embarks the Alignment API or as an Eclipse plug-in that connects through web services to some alignment server.

at run time through the web service access of the server (or any other available plug-in). It can also be used for generating alignments in the format used in the KAONp2p software.

So, the Alignment Server is clearly an "infrastructure component" (as of D6.2.1) which can be available directly for applications or through the NeOn toolkit (with the help of a plug-in providing GUI and engineering level capabilities).

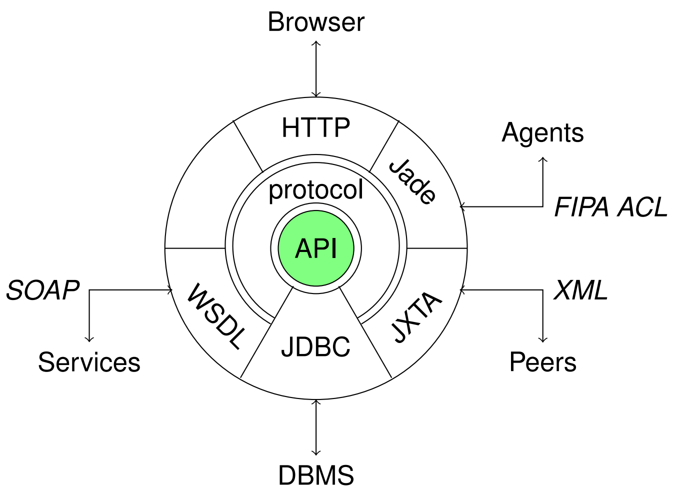


Figure 4.1: The Alignment Server is built on the Alignment API that is seated on top of a relational database repository for alignment and is wrapped around a simple protocol. Each access method is a plug-in that interacts with the server through the protocol. Currently, HTML, agent and web service plug-in are available.

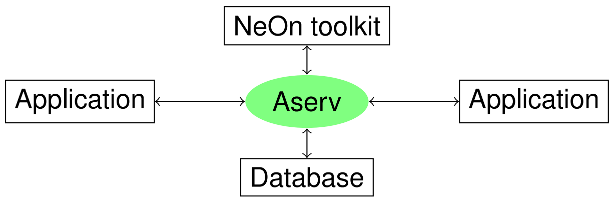


Figure 4.2: Distribution of the components using the Alignment Server: all can run on different machines.

There is no constraint that the alignments are computed online or off-line (i.e., they are stored in the alignment store) or that they are processed by hand or automatically. This kind of information can however be stored together with the alignment in order for the client to be able to discriminate among them.

The components of the Alignment Server as well as the connected clients can be distributed in different machines as presented in Figure 4.2. Several servers can share the same databases (the server works in write once mode: it never modifies an alignment but always creates new ones; not all the created alignments being stored in the database *in fine*). Applications can reach the Alignment Server by any way they want (e.g., starting by using Jade and then turning to web service interface).

Alignment services must be found on the semantic web. For that purpose they can be registered by service directories (e.g., UDDI for web services). In this first version the Alignment Server is called in a wired mode. However, we plan to develop protocols for the server to be registered by directories (this is already possible for Jade directories) and that servers can communicate in order to help each others (see Figure 4.3). Services or other agents should also be able to subscribe some particular results of interest by these services.

These directories are useful for other web services, agents, peers to find the alignment services. They are even more useful for alignment services to basically outsource some of their tasks. In particular, it may happen that:

– they cannot render an alignment in a particular format;
– they cannot process a particular matching method;
– they cannot access a particular ontology;
– a particular alignment is already stored by another service.

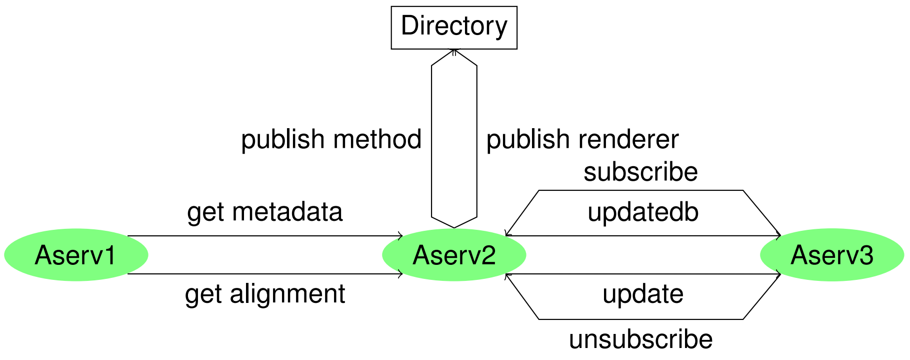


Figure 4.3: Protocol extension for registering and cooperating. It can work as a subscribe/notify protocol or as a direct invocation.

In these events, the concerned alignment service will be able to call other alignment services. This is especially useful when the client is not happy with the alignments provided by the current service, it is then possible to either deliver alignments provided by other services or to redirect the client to these services.

Moreover, this opens the door to value-added alignment services which use the results of other services as a pre-processing for their own treatments or which aggregate the results of other services in order to deliver a better alignment.

4.2 Functions

This infrastructure is able to store and retrieve alignments as well as providing them on the fly. We call it an infrastructure because it will be shared by the applications using ontologies on the semantic web. However, it may be seen as a directory or a service by web services, as an agent by agents, as a library in ambient computing applications, etc.

Services that are necessary in such an infrastructure are:

- storing alignments, whether they are provided by automatic means or by hand;
- storing annotations in order for the clients to evaluate them and to decide to use one of them or to start from it (this starts with the information about the matching algorithms, the justifications for correspondences that can be used in agent argumentation, as well as properties of the alignment);
- producing alignments on the fly through various algorithms that can be extended and parametrised;
- manipulating alignments by inverting them, applying thresholds;
- generating knowledge processors such as mediators, transformations, translators, rules as well as to process these processors if necessary;
- finding similar ontologies and contacting other such services in order to ask them for operations that the current service cannot provide by itself.

These tasks are summarised in Table 4.1.

Most of these services correspond to what is provided by any implementation of the Alignment API. The main principle of the Alignment API is that it can always be extended. In particular, it is possible to add new matching algorithms and mediator generators that will be accessible through the API. They will also be accessible through the alignment services. Services can thus be extended to new needs without breaking the infrastructure.

Moreover, the kind of annotations put on alignments is also extensible. So far, alignments contain information about:

- the kind of alignment it is (1:1 or n:m for instance);

Service	Syntax
Finding a similar ontology	$O' \Leftarrow Match(O, t)$
Align two ontologies	$A' \Leftarrow Align(O, O', A, p)$
Thresholding	$A' \Leftarrow Threshold(A, V)$
Generating code	$P \Leftarrow Render(A, language)$
Translating a message	$m' \Leftarrow Translate(m, A)$
Storing alignment	$n \Leftarrow Store(A, O, O')$
Suppressing alignment	$Delete(n)$
Finding (stored) alignments	$\{n\} \Leftarrow Find(O, O')$
Retrieving alignment	$\langle O, O', A\rangle \Leftarrow Retrieve(n)$

Table 4.1: Services provided by the alignment service and corresponding API primitives (O denotes an ontology, A an alignment, p parameters, n an index denoting an alignment, P a program realising the alignment and t and m some expressions, namely, terms to be matched and messages to be translated).

– the algorithm that provided it (or if it has been provided by hand);
– the language level used in the alignment (level 0 for the first example, level 2Horn for the second one);
– the confidence value in each correspondence.

Other valuable information that may be added to the alignment format are:

– the parameters passed to the generating algorithm;
– the properties satisfied by the correspondences (and their proof if necessary);
– the certificate from an issuing source;
– the limitations of the use of the alignment;
– the arguments in favour or against a correspondence [Euzenat *et al.*, 2005b].

The Alignment API has been opened so that it is now possible to store more annotations to alignments and correspondences.

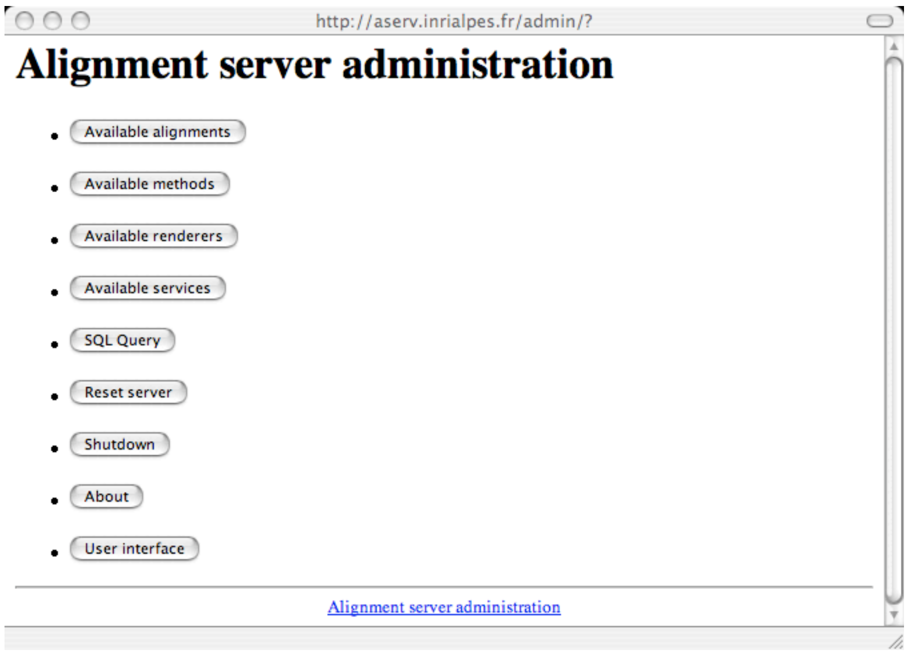
4.3 Alignment Server HTML interaction

We display here screendumps of the HTML interface of the current alignment server. This illustrates the features offered by the Alignment Server better than a long descriptive text. The goal, however, is not that NeOn uses this interface but rather that it takes advantage of the web service interface.

4.4 Summary

We have presented a structure for finding and sharing alignments between ontologies. It will be used for integrating some NeOn developments and will be integrated within the NeOn toolkit. It is working (and an experimental server is available) but is still under heavy developments.

This architecture can be used for recontextualising ontologies and providing suport for building networked ontologies. The next chapter is dedicated to the development of specific techniques for matching ontologies based on recontextialisation.



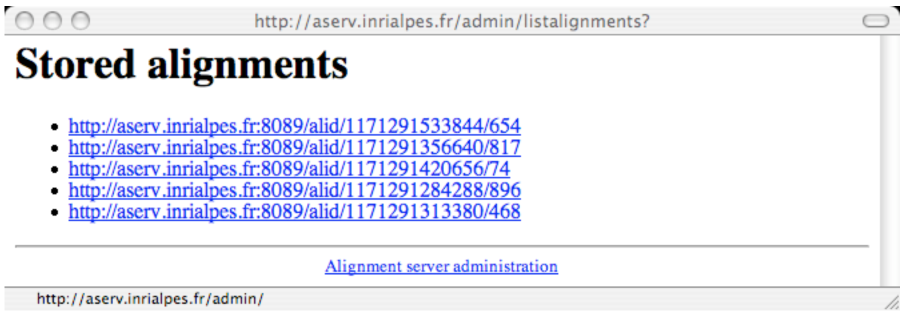
http://aserv.inrialpes.fr/admin/?

Alignment server administration

- Available alignments
- Available methods
- Available renderers
- Available services
- SQL Query
- Reset server
- Shutdown
- About
- User interface

Alignment server administration

Figure 4.4: System menu providing access to management primitives.



http://aserv.inrialpes.fr/admin/listalignments?

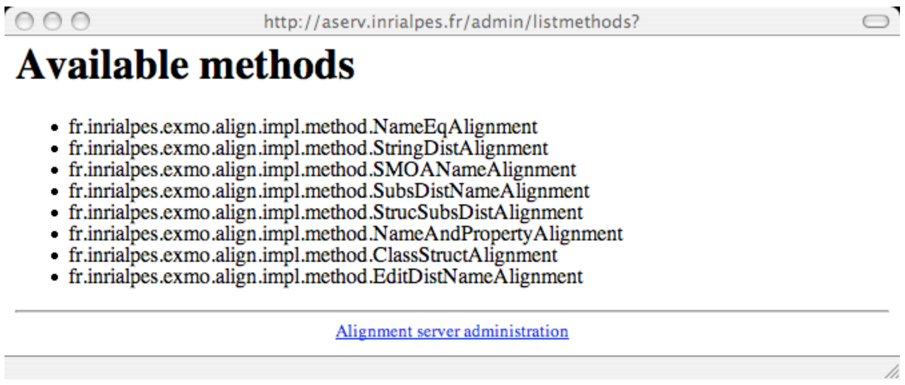
Stored alignments

- http://aserv.inrialpes.fr:8089/alid/1171291533844/654
- http://aserv.inrialpes.fr:8089/alid/1171291356640/817
- http://aserv.inrialpes.fr:8089/alid/1171291420656/74
- http://aserv.inrialpes.fr:8089/alid/1171291284288/896
- http://aserv.inrialpes.fr:8089/alid/1171291313380/468

Alignment server administration

http://aserv.inrialpes.fr/admin/

Figure 4.5: List of available alignments in the server.



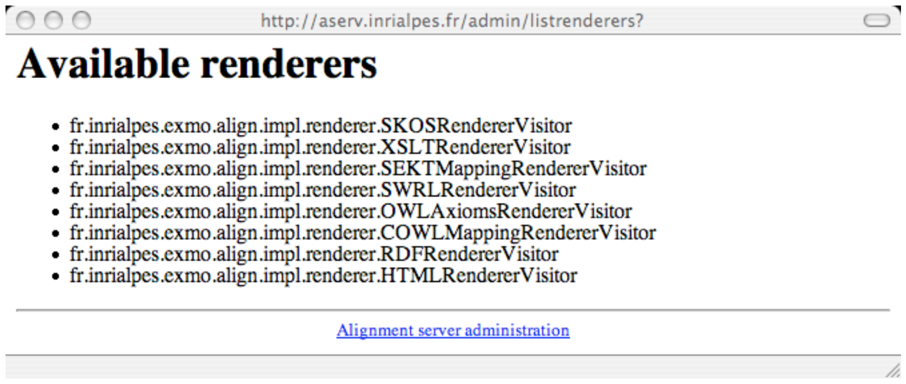
http://aserv.inrialpes.fr/admin/listmethods?

Available methods

- fr.inrialpes.exmo.align.impl.method.NameEqAlignment
- fr.inrialpes.exmo.align.impl.method.StringDistAlignment
- fr.inrialpes.exmo.align.impl.method.SMOANameAlignment
- fr.inrialpes.exmo.align.impl.method.SubsDistNameAlignment
- fr.inrialpes.exmo.align.impl.method.StrucSubsDistAlignment
- fr.inrialpes.exmo.align.impl.method.NameAndPropertyAlignment
- fr.inrialpes.exmo.align.impl.method.ClassStructAlignment
- fr.inrialpes.exmo.align.impl.method.EditDistNameAlignment

Alignment server administration

Figure 4.6: List of available methods in the server.

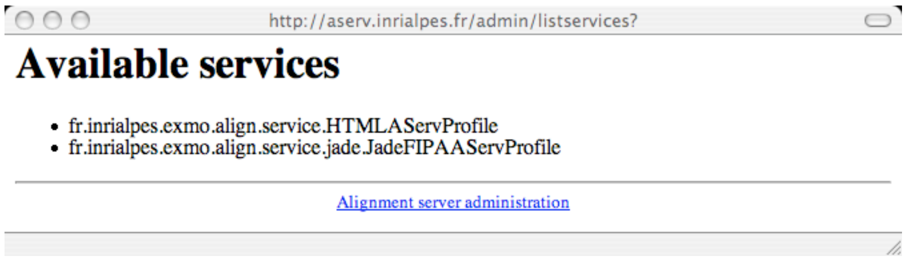

http://aserv.inrialpes.fr/admin/listrenderers?

Available renderers

- fr.inrialpes.exmo.align.impl.renderer.SKOSRendererVisitor
- fr.inrialpes.exmo.align.impl.renderer.XSLTRendererVisitor
- fr.inrialpes.exmo.align.impl.renderer.SEKTMappingRendererVisitor
- fr.inrialpes.exmo.align.impl.renderer.SWRLRendererVisitor
- fr.inrialpes.exmo.align.impl.renderer.OWLAxiomsRendererVisitor
- fr.inrialpes.exmo.align.impl.renderer.COWLMappingRendererVisitor
- fr.inrialpes.exmo.align.impl.renderer.RDFRendererVisitor
- fr.inrialpes.exmo.align.impl.renderer.HTMLRendererVisitor

Alignment server administration

Figure 4.7: List of available renderers in the server.

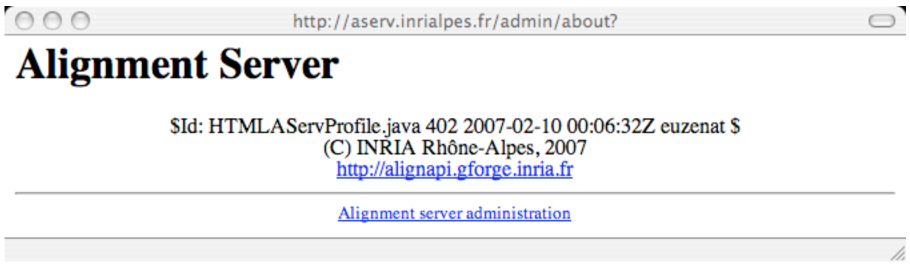

http://aserv.inrialpes.fr/admin/listservices?

Available services

- fr.inrialpes.exmo.align.service.HTMLAServProfile
- fr.inrialpes.exmo.align.service.jade.JadeFIPAAServProfile

Alignment server administration

Figure 4.8: List of available services in the server.

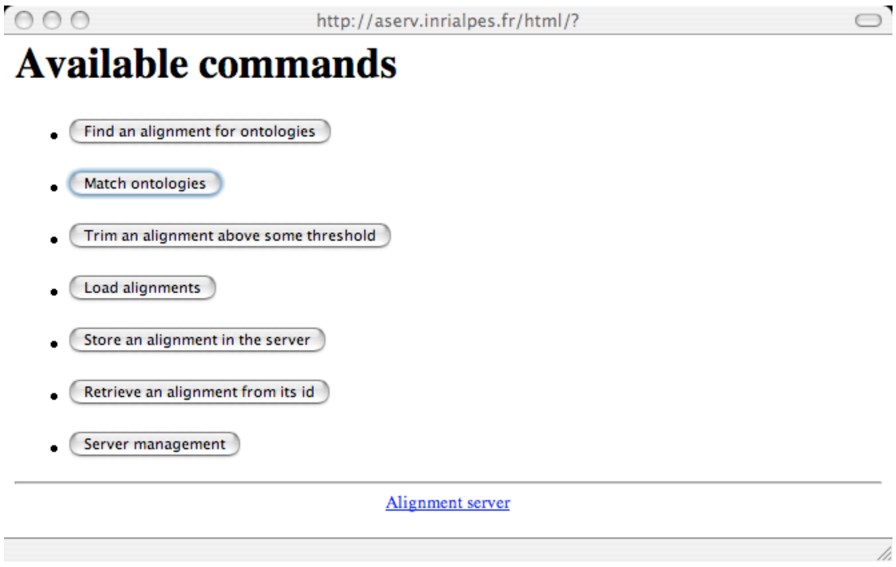

http://aserv.inrialpes.fr/admin/about?

Alignment Server

$Id: HTMLAServProfile.java 402 2007-02-10 00:06:32Z euzenat $
(C) INRIA Rhône-Alpes, 2007
http://alignapi.gforge.inria.fr

Alignment server administration

Figure 4.9: Information about the server.



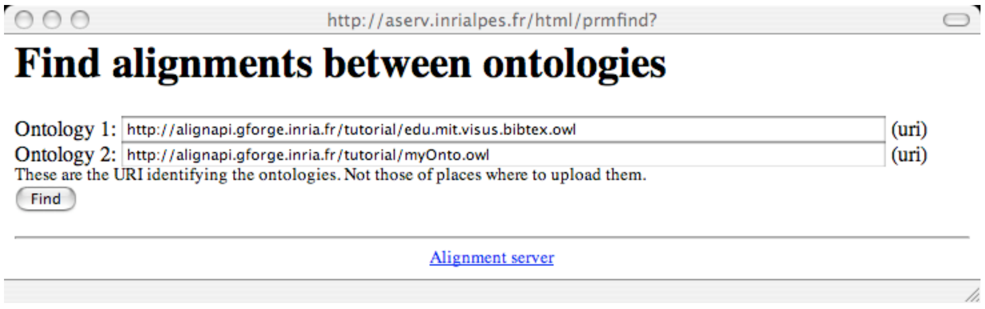
http://aserv.inrialpes.fr/html/?

Available commands

- Find an alignment for ontologies
- Match ontologies
- Trim an alignment above some threshold
- Load alignments
- Store an alignment in the server
- Retrieve an alignment from its id
- Server management

Alignment server

Figure 4.10: User menu providing access to alignment computation and manipulation operations.



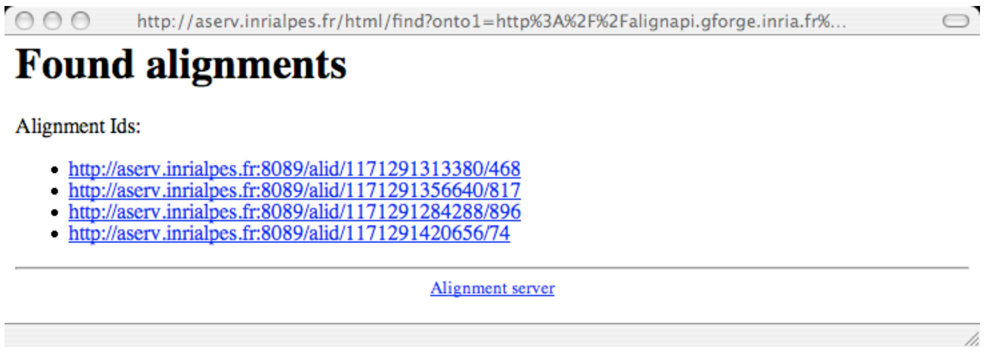
http://aserv.inrialpes.fr/html/prmfind?

Find alignments between ontologies

Ontology 1: http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl (uri)
Ontology 2: http://alignapi.gforge.inria.fr/tutorial/myOnto.owl (uri)
These are the URI identifying the ontologies. Not those of places where to upload them.
Find

Alignment server

Figure 4.11: Looking for alignments between two specific ontologies. This returns alignments available from the server.



http://aserv.inrialpes.fr/html/find?onto1=http%3A%2F%2Falignapi.gforge.inria.fr%...

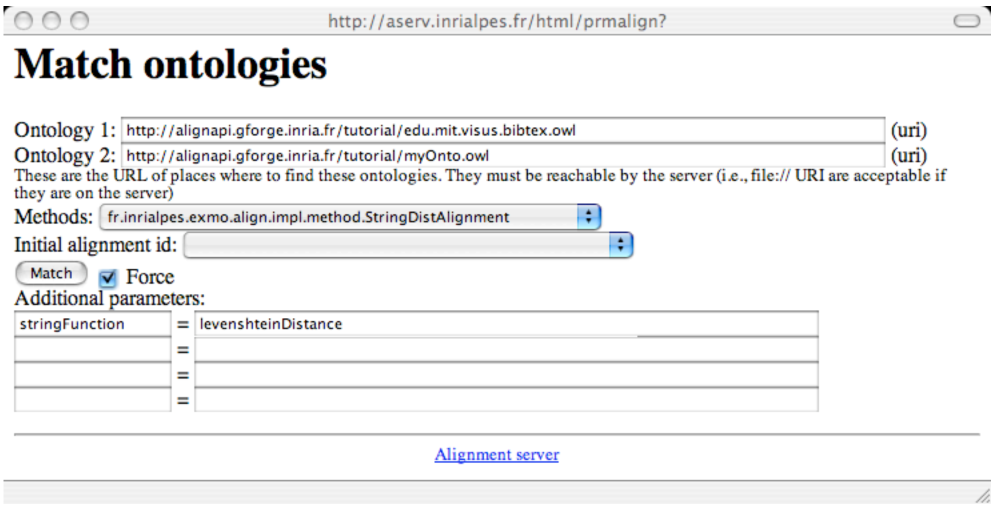
Found alignments

Alignment Ids:

- http://aserv.inrialpes.fr:8089/alid/1171291313380/468
- http://aserv.inrialpes.fr:8089/alid/1171291356640/817
- http://aserv.inrialpes.fr:8089/alid/1171291284288/896
- http://aserv.inrialpes.fr:8089/alid/1171291420656/74

Alignment server

Figure 4.12: Found alignments between the ontologies.


http://aserv.inrialpes.fr/html/prmalign?

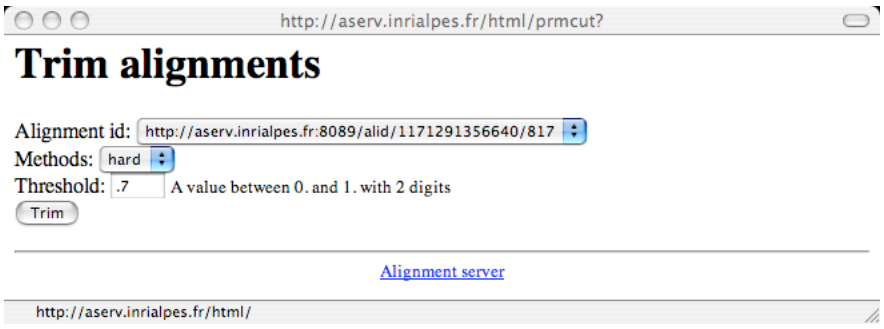
Match ontologies

Ontology 1: http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl (uri)
Ontology 2: http://alignapi.gforge.inria.fr/tutorial/myOnto.owl (uri)
These are the URL of places where to find these ontologies. They must be reachable by the server (i.e., file:// URI are acceptable if they are on the server)
Methods: fr.inrialpes.exmo.align.impl.method.StringDistAlignment
Initial alignment id:
Match ✓ Force
Additional parameters:

stringFunction	=	levenshteinDistance
	=	
	=	
	=	

Alignment server

Figure 4.13: Invoking a matching algorithm for matching two ontologies. The interface allows the selection of the method and the input of additional parameters. It can force the alignment to be recomputed if a similar one already exists.


http://aserv.inrialpes.fr/html/prmcut?

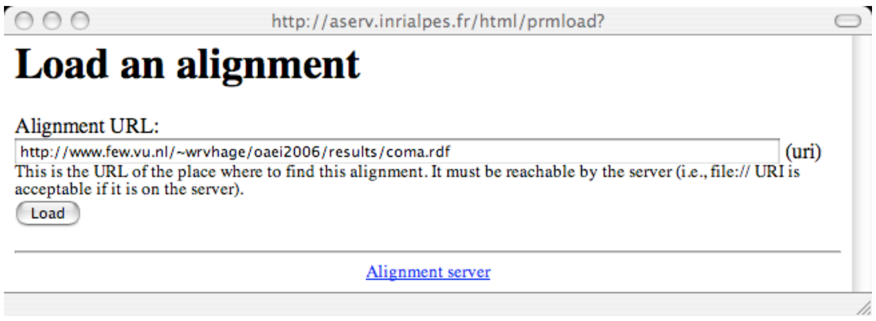
Trim alignments

Alignment id: http://aserv.inrialpes.fr:8089/alid/1171291356640/817
Methods: hard
Threshold: .7 A value between 0. and 1. with 2 digits
Trim

Alignment server

http://aserv.inrialpes.fr/html/

Figure 4.14: Applying thresholds to existing alignments. Various methods for applying these thresholds are available.

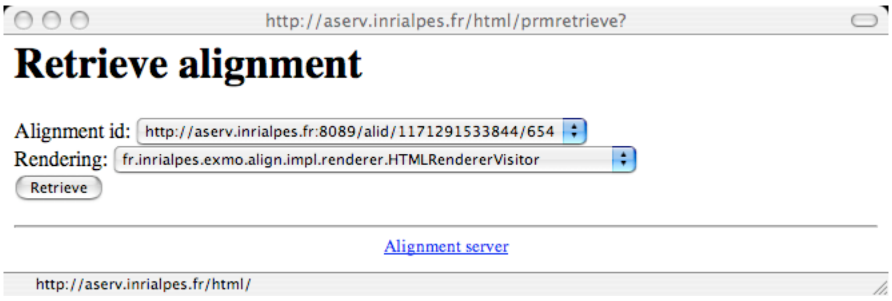

http://aserv.inrialpes.fr/html/prmload?

Load an alignment

Alignment URL:
http://www.few.vu.nl/~wrvhage/oaei2006/results/coma.rdf (uri)
This is the URL of the place where to find this alignment. It must be reachable by the server (i.e., file:// URI is acceptable if it is on the server).
Load

Alignment server

Figure 4.15: Uploading an alignment on the server (here from the OAEI web site).


http://aserv.inrialpes.fr/html/prmretrieve?

Retrieve alignment

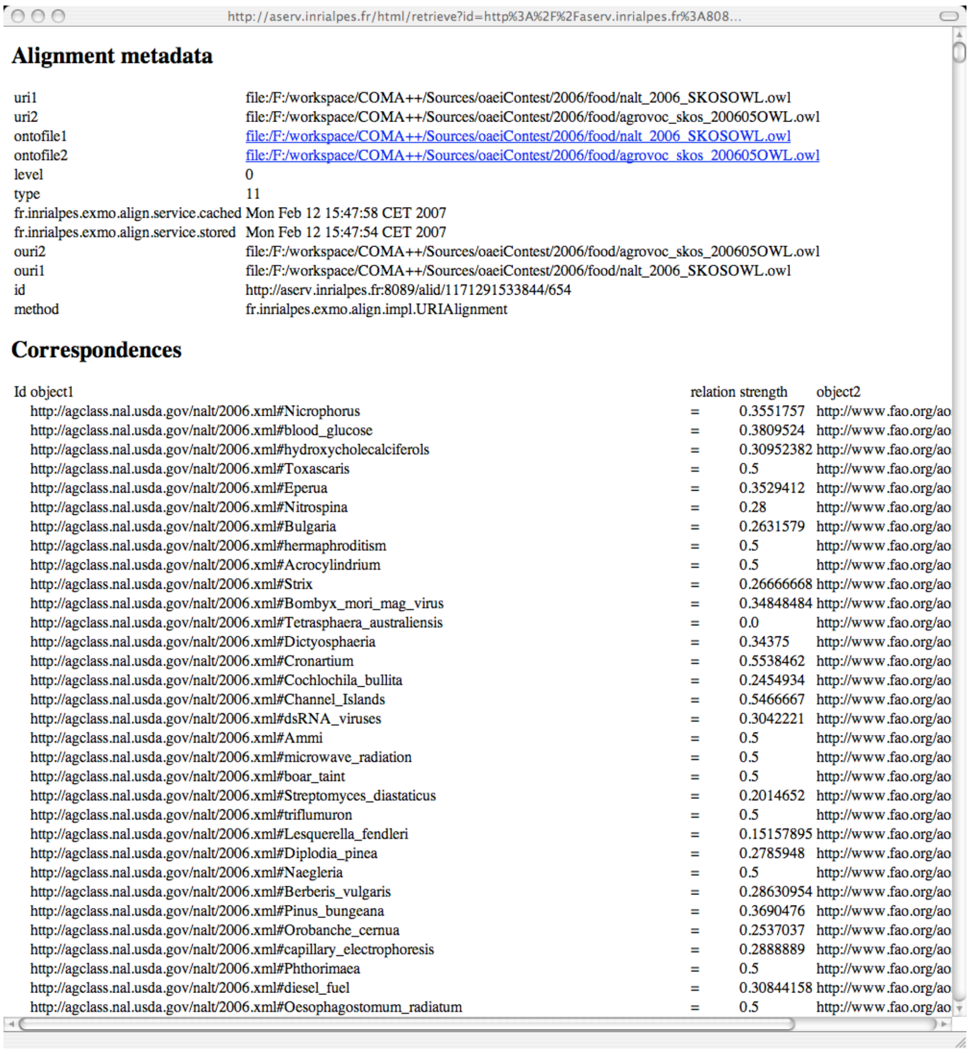
Alignment id:	http://aserv.inrialpes.fr:8089/alid/1171291533844/654
Rendering:	fr.inrialpes.exmo.align.impl.renderer.HTMLRendererVisitor

Retrieve

Alignment server

http://aserv.inrialpes.fr/html/

Figure 4.16: Rendering an alignment of the server in a particular format (here HTML, but all renderers are accessible).


http://aserv.inrialpes.fr/html/retrieve?id=http%3A%2F%2Faserv.inrialpes.fr%3A808...

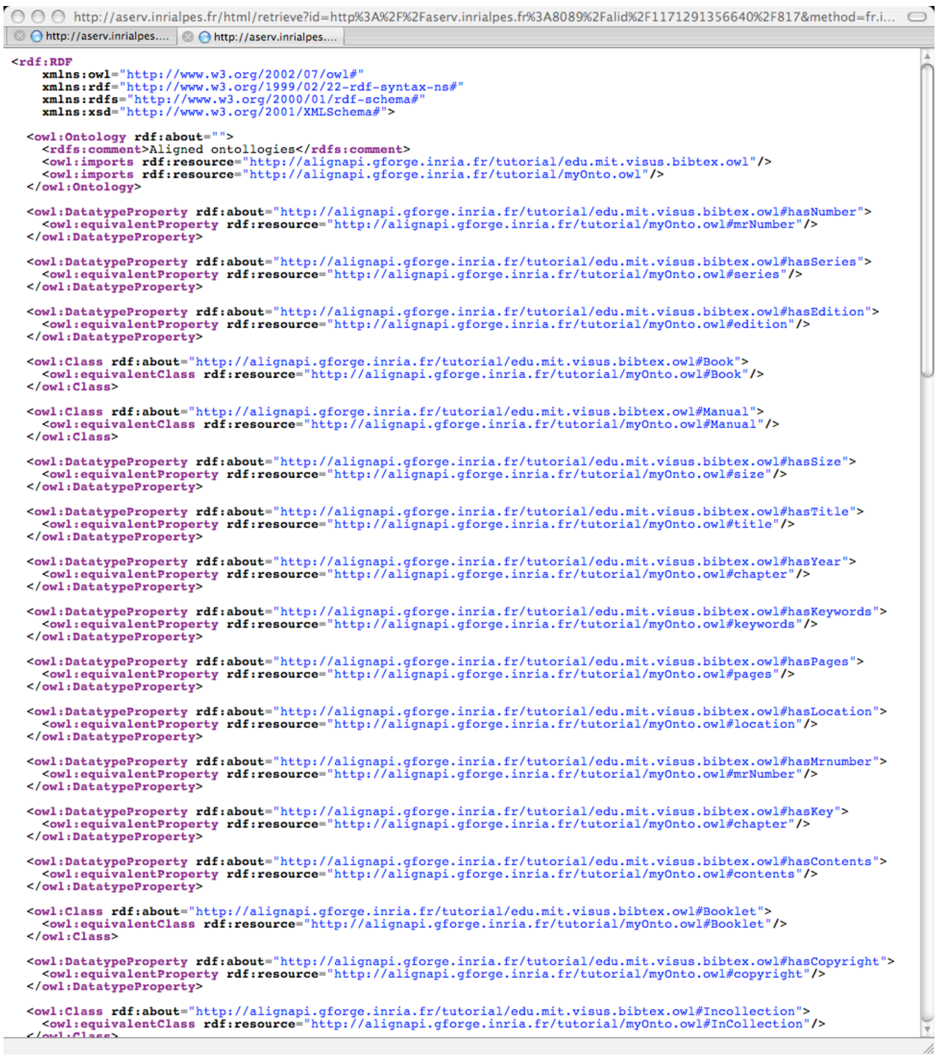
Alignment metadata

uri1	file:/F:/workspace/COMA++/Sources/oaeiContest/2006/food/nalt_2006_SKOSOWL.owl
uri2	file:/F:/workspace/COMA++/Sources/oaeiContest/2006/food/agrovoc_skos_200605OWL.owl
ontofile1	file:/F:/workspace/COMA++/Sources/oaeiContest/2006/food/nalt_2006_SKOSOWL.owl
ontofile2	file:/F:/workspace/COMA++/Sources/oaeiContest/2006/food/agrovoc_skos_200605OWL.owl
level	0
type	11
fr.inrialpes.exmo.align.service.cached	Mon Feb 12 15:47:58 CET 2007
fr.inrialpes.exmo.align.service.stored	Mon Feb 12 15:47:54 CET 2007
ouri2	file:/F:/workspace/COMA++/Sources/oaeiContest/2006/food/agrovoc_skos_200605OWL.owl
ouri1	file:/F:/workspace/COMA++/Sources/oaeiContest/2006/food/nalt_2006_SKOSOWL.owl
id	http://aserv.inrialpes.fr:8089/alid/1171291533844/654
method	fr.inrialpes.exmo.align.impl.URIAlignment

Correspondences

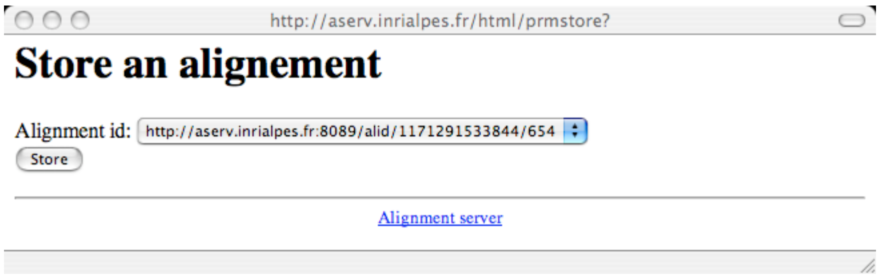
Id	object1	relation	strength	object2
	http://agclass.nal.usda.gov/nalt/2006.xml#Nicrophorus	=	0.3551757	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#blood_glucose	=	0.3809524	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#hydroxycholecalciferols	=	0.30952382	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Toxascaris	=	0.5	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Eperua	=	0.3529412	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Nitrospina	=	0.28	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Bulgaria	=	0.2631579	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#hermaphroditism	=	0.5	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Acrocylindrium	=	0.5	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Strix	=	0.26666668	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Bombyx_mori_mag_virus	=	0.34848484	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Tetrasphaera_australiensis	=	0.0	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Dictyosphaeria	=	0.34375	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Cronartium	=	0.5538462	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Cochlochila_bullita	=	0.2454934	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Channel_Islands	=	0.5466667	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#dsRNA_viruses	=	0.3042221	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Ammi	=	0.5	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#microwave_radiation	=	0.5	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#boar_taint	=	0.5	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Streptomyces_diastaticus	=	0.2014652	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#triflumuron	=	0.5	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Lesquerella_fendleri	=	0.15157895	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Diplodia_pinea	=	0.2785948	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Naegleria	=	0.5	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Berberis_vulgaris	=	0.28630954	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Pinus_bungeana	=	0.3690476	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Orobanche_cernua	=	0.2537037	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#capillary_electrophoresis	=	0.2888889	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Phthorimaea	=	0.5	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#diesel_fuel	=	0.30844158	http://www.fao.org/ao
	http://agclass.nal.usda.gov/nalt/2006.xml#Oesophagostomum_radiatum	=	0.5	http://www.fao.org/ao

Figure 4.17: Displaying an alignment in HTML as the result of the previous command. This is between AGROVOC and NAL and has been generated by the COMA system (note the side bar).


http://aserv.inrialpes.fr/html/retrieve?id=http%3A%2F%2Faserv.inrialpes.fr%3A8089%2Falid%2F1171291356640%2F817&method=fr.i...

```
<rdf:RDF
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#">

  <owl:Ontology rdf:about="">
    <rdfs:comment>Aligned ontollogies</rdfs:comment>
    <owl:imports rdf:resource="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl"/>
    <owl:imports rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl"/>
  </owl:Ontology>

  <owl:DatatypeProperty rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#hasNumber">
    <owl:equivalentProperty rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#mrNumber"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#hasSeries">
    <owl:equivalentProperty rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#series"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#hasEdition">
    <owl:equivalentProperty rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#edition"/>
  </owl:DatatypeProperty>

  <owl:Class rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#Book">
    <owl:equivalentClass rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#Book"/>
  </owl:Class>

  <owl:Class rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#Manual">
    <owl:equivalentClass rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#Manual"/>
  </owl:Class>

  <owl:DatatypeProperty rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#hasSize">
    <owl:equivalentProperty rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#size"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#hasTitle">
    <owl:equivalentProperty rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#title"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#hasYear">
    <owl:equivalentProperty rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#chapter"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#hasKeywords">
    <owl:equivalentProperty rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#keywords"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#hasPages">
    <owl:equivalentProperty rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#pages"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#hasLocation">
    <owl:equivalentProperty rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#location"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#hasMrnumber">
    <owl:equivalentProperty rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#mrNumber"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#hasKey">
    <owl:equivalentProperty rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#chapter"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#hasContents">
    <owl:equivalentProperty rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#contents"/>
  </owl:DatatypeProperty>

  <owl:Class rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#Booklet">
    <owl:equivalentClass rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#Booklet"/>
  </owl:Class>

  <owl:DatatypeProperty rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#hasCopyright">
    <owl:equivalentProperty rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#copyright"/>
  </owl:DatatypeProperty>

  <owl:Class rdf:about="http://alignapi.gforge.inria.fr/tutorial/edu.mit.visus.bibtex.owl#Incollection">
    <owl:equivalentClass rdf:resource="http://alignapi.gforge.inria.fr/tutorial/myOnto.owl#InCollection"/>
  </owl:Class>
```

Figure 4.18: Displaying another alignment as a set of OWL axioms.


http://aserv.inrialpes.fr/html/prmstore?

Store an alignement

Alignment id: http://aserv.inrialpes.fr:8089/alid/1171291533844/654

Store

Alignment server

Figure 4.19: Storing an alignment permanently in the database.

Chapter 5

Contextual matching

In Chapter 3 a distinction has been made between generic matching systems on one hand and context-based matching on the other hand. In this chapter we present a context-based matching system that re-contextualises the mapped ontologies by relying on dynamically selected background knowledge from online ontologies. We start this chapter by a closer look to the advantages of context-based matchers with respect to generic matching systems, and conclude on the major limitation of existing context-based matchers, namely their domain dependence (Section 5.1). Then we describe our technique which, while it inherits all the desirable features of context-based matching is domain independent (Section 5.2). We also report on the evaluation of this technique on a large scale, real life data set involving the AGROVOC thesaurus which is of core interest to NeOn (see WP 7). The experimental setup and the obtained results are presented in Sections 5.3 and 5.4 and discussed in Section 5.5.

5.1 A closer look to context-based matchers

Methods that rely on a context ontology overcome two major limitations of traditional techniques. First, as observed by [Aleksovski *et al.*, 2006], traditional methods fail when there is little lexical overlap between the labels of the ontology entities, or when the ontologies have weak or dissimilar structures. Indeed, these techniques are based on the hypothesis that syntactic correspondences imply semantic relations. While in many cases meaningful mappings can be derived from string and structural similarities, this hypothesis is far from being always verified. For instance, the relation between the concepts *Beef* and *Food* may not be discovered on the basis of syntactic considerations, but becomes obvious when considering the meaning of these concepts (their semantics). By ignoring such semantics, syntactic techniques fail to identify several important mappings. Relying on a context ontology allows deriving mappings in cases when the compared ontologies are dissimilar in their labels and structure. Indeed many alignments between dissimilar structures that cannot be found with traditional techniques are likely to already exist in the context ontology. Such alignments, are typically those that could be expected to be found in an Alignment Server.

A second advantage of using a context ontology is that its richly axiomatised nature guarantees the semantic nature of the derived mappings. In contrast to this, the majority of existing systems generally derive correspondences between terms, eventually associated with a confidence that aggregates different similarity values. For example, during the 2005 Ontology Alignment Contest (OAC) [Euzenat *et al.*, 2005a] only one algorithm (CtxMatch [Bouquet *et al.*, 2005]) was able to go beyond equivalences and also produce partial relations in the form of subconcept relations. In this year, all the tools produced only weighted correspondences [Euzenat *et al.*, 2006], thus failing to provide semantic mappings that would be directly exploitable in dynamic scenarios as machine interpretable relations.

A major disadvantage of the above mentioned approaches that rely on contextualisation is that the appropriate context ontology needs to be manually selected prior to matching. In many scenarios this approach is unfeasible as we might not know in advance which terms from which ontologies need to be matched. Even in the cases where a reference ontology can be manually selected prior to performing the matching, there is

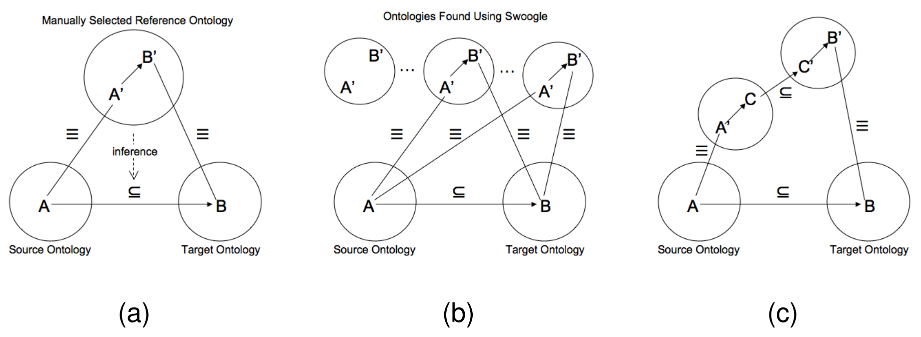


Figure 5.1: Using ontologies as background knowledge for semantic matching: (a) using a manually selected reference ontology (e.g., [Aleksovski *et al.*, 2006]); (b) using Swoogle to find the appropriate ontologies (S1) (c) recursively exploiting multiple ontologies (S2).

no guarantee that such an ontology actually exists. Indeed, unlike medicine, many domains lack large, richly formalised domain ontologies that would cover the ontologies to be matched.

In the next section we describe an approach which relies on dynamic selection of context ontologies from the Web thus avoiding dependence on a single, large, manually pre-selected ontology (Section 5.2). This technique complies with the Semantic Web vision: it derives semantic mappings by exploring multiple and heterogeneous online ontologies that are dynamically selected and exploited through Swoogle [Ding *et al.*, 2005]. As such, while enjoying all the advantages derived from the use of context ontologies, the technique remains completely domain independent.

5.2 Matching by dynamically exploring online knowledge

Our hypothesis is that the growing amount of online available semantic data which makes up the Semantic Web can be used as a source of contextual knowledge. Indeed, this large-scale, heterogeneous semantic data collection provides formally specified knowledge which allows deriving high quality semantic mappings. Moreover, the size and heterogeneity of the collection makes it possible to dynamically select and combine the appropriate knowledge and to avoid the manual selection of a single, domain specific knowledge source. In the following we describe two increasingly sophisticated strategies to discover and exploit online available ontologies for matching. Each strategy is presented as a procedure that takes two candidate concept names (A and B) as an input and returns the discovered mapping between them. We use the letters A and B to refer to these candidate concept names. The corresponding concepts in the selected ontology are A′ and B′ ("anchor terms"). We rely on description logic syntax for semantic relations occurring between concepts in an ontology, e.g., $A' \sqsubseteq B'$ means that A′ is a sub-concept of B′ in a selected ontology and $A' \perp B'$ means that A′ and B′ are disjoint. The returned mappings are expressed using C-OWL [Bouquet *et al.*, 2004a] like notations, e.g., such as $A \xrightarrow{\sqsubseteq} B$ or $A \xrightarrow{\perp} B$.

5.2.1 S1: Mappings Within One Ontology

Our simplest strategy consists of using Swoogle to find ontologies containing concepts with the same names as the candidate concepts and to derive mappings from their relationship in the selected ontologies. This ontology provides the context in which the matching operation is performed. Figure 5.1(b) illustrates this strategy with an example where three ontologies are discovered containing the concepts A′ and B′ with the same names as A and B. The first ontology contains no relation between the anchor concepts, while the other two ontologies contain a subsumption relation. The concrete steps of this strategy are:

1. Select ontologies containing concepts A′ and B′ corresponding to A and B;

2. For each resulting ontology:
   - if $A' \equiv B'$ then derive $A \xrightarrow{\equiv} B$;
   - if $A' \sqsubseteq B'$ then derive $A \xrightarrow{\sqsubseteq} B$;
   - if $A' \sqsupseteq B'$ then derive $A \xrightarrow{\sqsupseteq} B$;
   - if $A' \perp B'$ then derive $A \xrightarrow{\perp} B$;
3. If no ontology is found, no mapping is derived;

In [Sabou *et al.*, 2006], we have discussed a range of implementation choices for this strategy. In particular, we have distinguished two variants depending on the number of ontologies that are inspected to derive a mapping. The first and simplest variant stops as soon as a mapping is discovered. This is the easiest way to deal with the multiple returned ontologies but it assumes that the first discovered relation can be trusted and there is no need to inspect the other ontologies. The second variant, prefers accuracy to time performance and consists of deriving all possible mappings from all the ontologies that contain the terms to be mapped and then combining them in various ways. This is a more difficult variant because mappings resulting from different sources can be different (e.g., $A \xrightarrow{\sqsubseteq} B$ and $A \xrightarrow{\sqsupseteq} B$), or, in the worst case, inconsistent (e.g., $A \xrightarrow{\sqsubseteq} B$ and $A \xrightarrow{\perp} B$). Several ways of dealing with these contradictions can be considered: we can keep all the mappings (favoring recall), only keep mappings without contradiction (favoring precision), keep the mappings that are derived from most of the ontologies, or try to combine the results (e.g., by deriving $A \xrightarrow{\equiv} B$ from $A \xrightarrow{\sqsubseteq} B$ and $A \xrightarrow{\sqsupseteq} B$).

5.2.2 S2: Cross-Ontology Mapping Discovery

The previous strategy assumes that a semantic relation between the candidate concepts can be discovered in a single ontology. However, some relations could be distributed over several ontologies. Therefore, if no ontology is found that relates both candidate concepts, then the mappings should be derived from two (or more) ontologies. Note that in this case the context for the mapping would not anymore be provided by a single ontology, but rather a collection (a network) of ontologies. In this strategy, matching is a recursive task where two concepts can be mapped because the concepts they relate in some ontologies are themselves mapped (Figure 5.1(c)). Concretely:

1. If no ontologies are found that contain both A' and B' then select all ontologies containing a concept A' corresponding to A;
2. For each of the resulting ontologies:
   (a) for each P such that $A' \sqsubseteq P$, search for mappings between P and B';
   (b) for each C such that $A' \sqsupseteq C$, search for mappings between C and B';
   (c) derive mappings using the following rules:
   - (r1) if $A' \sqsubseteq P$ and $P \xrightarrow{\sqsubseteq} B'$ then $A \xrightarrow{\sqsubseteq} B$
   - (r2) if $A' \sqsubseteq P$ and $P \xrightarrow{\equiv} B'$ then $A \xrightarrow{\sqsubseteq} B$
   - (r3) if $A' \sqsubseteq P$ and $P \xrightarrow{\perp} B'$ then $A \xrightarrow{\perp} B$
   - (r4) if $A' \sqsupseteq C$ and $C \xrightarrow{\sqsupseteq} B'$ then $A \xrightarrow{\sqsupseteq} B$
   - (r5) if $A' \sqsupseteq C$ and $C \xrightarrow{\equiv} B'$ then $A \xrightarrow{\sqsupseteq} B$

In this strategy, steps (a) and (b) can be run in parallel and stopped when one of them is able to establish a mapping. These two steps correspond to the recursive part of the algorithm. The task of *searching for mappings between C (P) and B'* can be realised using either strategy S1 or S2.

5.3 Experimental setup

In this section we describe the concrete implementations of S1 and S2 that were used (Section 5.3.1) as well as the particularities of the data sets (Section 5.3.2).

5.3.1 Implementation details

The anchoring mechanism in the case of both strategies (i.e., finding `A', B'`) is based on strict string matching. For simple terms (made up of one word) we find anchors that match this word as well as its lemma (i.e., base form): a term *Persons* will be anchored to concepts labeled either *Persons* or *Person*. For compound terms (containing multiple words) we discover concept labels containing the same words, in the same order, but possibly written according to different naming conventions: *WindMill = Wind_Mill = wind mill*. We implemented both strategies by using data extracted from Swoogle'05[1]. For S1 we implemented both the variant that returns the first derived mapping and the one that returns all the possible mappings between two terms. While S2 is much more time consuming than S1, its complexity can be reduced by taking some simplifying assumptions. To reduce the time of our experiments, we implemented S2 in such a way that only the first direct parent of the discovered `A'` concepts is considered (instead of exploring *all* parents). Then, we did not consider more specific concepts of `A'` both to simplify the algorithm, but also because this information cannot be queried from Swoogle (i.e., for any given concept we can only get its parents). Therefore rules $r4$ and $r5$ are not applied.

5.3.2 Data sets

The mapping was performed between two very large, real life thesauri. The United Nations Food and Agriculture Organization (FAO)'s AGROVOC thesaurus, version May 2006, consists of 28 174 descriptor terms (i.e., preferred terms) and 10 028 non-descriptor terms (i.e., alternative terms). The United States National Agricultural Library (NAL) Agricultural thesaurus, version 2006, NALT consists of 41 577 descriptor terms and 24 525 non-descriptor terms.

It is important to note that these thesauri describe a broad range of domains ranging from animal species, to chemical substances and information technology. Also, they use several technical terms (e.g., from chemistry) and a considerable amount of Latin terms (e.g., to describe animal species). Since our algorithm depends on the thesauri terms being represented in online ontologies, we filtered out those that are covered according to the anchoring mechanism described above. We found that 2 046 (1 594 simple, 452 compound) AGROVOC descriptor terms were mentioned in at least one ontology and 2 595 (1908 simple, 687 compound) NALT terms were also covered. Only 7,2% AGROVOC and 6,2% NALT terms are covered due to the fact that online ontologies are mostly concerned with generic knowledge instead of specialised, domain terms. Another explanation is that compound terms are much harder to anchor than simple ones. In fact, we found 16% (10%) of the simple and only 2,4% (2,8%) of the compound terms of AGROVOC (NALT respectively). We use these two sets of terms as input to our experiment and try to determine mappings between them.

5.4 Experimental results

We describe the results of applying both variants of S1 (Sections 5.4.1 and 5.4.2), S2 (Section 5.4.3) and compare these results to those of other tools (Section 5.4.4).

[1]Swoogle'06 is too unstable to allow performing extensive experiments.

	Nr.	Examples
Subclass ($\xrightarrow{\sqsubseteq}$)	1477	$Lamb \xrightarrow{\sqsubseteq} Sheep$, $Soap \xrightarrow{\sqsubseteq} Detergent$, $Asbestos \xrightarrow{\sqsubseteq} Pollutant$ $Oasis \xrightarrow{\sqsubseteq} Ecosystem$, $RAM \xrightarrow{\sqsubseteq} ComputerEquipment$
SuperClass ($\xrightarrow{\sqsupseteq}$)	1857	$Shop \xrightarrow{\sqsupseteq} Supermarket$, $Spice \xrightarrow{\sqsupseteq} BlackPepper$, $Valley \xrightarrow{\sqsupseteq} Canyon$ $Infrastructure \xrightarrow{\sqsupseteq} Highway$, $Storm \xrightarrow{\sqsupseteq} Tornado$, $Rock \xrightarrow{\sqsupseteq} Crystal$
Disjoint ($\xrightarrow{\perp}$)	229	$Fluid \xrightarrow{\perp} Solid$, $Fluid \xrightarrow{\perp} Gas$, $Pond \xrightarrow{\perp} River$, $Plant \xrightarrow{\perp} Animal$ $Newspaper \xrightarrow{\perp} Journal$, $Fruit \xrightarrow{\perp} Vegetable$, $Female \xrightarrow{\perp} Male$
Total	3563	

Table 5.1: Discovered mappings between AGROVOC and NALT.

	First Round	Second Round
Correct	1173	1144
False	218	421
Not-Evaluated	259	85
Precision	71%	69%
Average Precision	**70%**	

Table 5.2: Mapping evaluation.

5.4.1 Results for strategy S1, first variant

The first variant of S1 discovered 3 563 distinct mappings: 229 disjoint, 1 477 subclass and 1 857 superclass relations (see Table 5.1).

Given the high number of discovered mappings as well as the lack of a Gold Standard against which to compare them, we performed a manual assessment on a statistically significant subset of the results (1650 mappings, i.e., 46%) in order to get an indication of the precision. In a first round, we asked seven subjects (all working in the field of the Semantic Web) to mark their 200+ mappings as *correct*, *false* or *"I don't know"* in cases when they were unable to judge the correctness of the mapping. In a second round, the sample was split in half between two of the authors. While in the first round we asked our subjects to rank only those mappings that they were sure about without any investigation, in the second round we have investigated some mappings that require specialised knowledge (e.g., by querying Google). In the first round we obtained a precision of 71% (we counted the unevaluated items as false) and in the second a very close value of 69% (Table 5.2). Therefore, we can conclude that an indicative value for the precision of our technique is around 70%.

This two stage evaluation process provided interesting information on the "ambiguity" of the derived mappings. We found that, evaluators agreed on 1 045 assessments (two ratings of *correct* or *false*) and disagreed on 293 (one rating of *correct* and one of *false*). The rest of 312 assessments were rated at least once with *I don't know*, and therefore they do not really represent disagreements (either one or both of the evaluators did not know enough about the terms to judge the mapping). Therefore, strictly speaking, the evaluators disagreed on 18% of the mappings. This shows that many mappings are straightforward to evaluate, but at the same time, a certain dependence on the perspective of the evaluator exists.

An analysis of the false mappings led us to distinguish two sets of common causes for deriving such erroneous mappings. These were:

Errors introduced by exploring low quality ontologies. Some of the low quality online ontologies incorrectly use inheritance to express a range of relations. First, roles are often modelled as subclass relations, for example, that $Aubergine, Leek \sqsubseteq Ingredient$ (in fact, $Leek$ is a $Vegetable$ but in some contexts it plays the role of an ingredient) or $Wool \sqsubseteq Product$ (however, $Wool$ is a $Fabric$, but it can also play the role of a $Product$). Second, inheritance is often used to model a part-whole relation (e.g., $Branch \sqsubseteq Tree$). Third, more broadly, inheritance is used as a way to model the fact that there exists some type of relation between two concepts, e.g., $Survey \sqsubseteq Marketing$, $Irrigation \sqsubseteq Agriculture$,

$Biographies \sqsubseteq People$. When explored, all these types of mistakes lead to erroneous mappings. Finally, there are a set of correct inheritance relations which introduce errors due to the inaccurate labeling of their concepts. For example, in $Oil \sqsubseteq Industry$, Oil refers to $OilIndustry$ rather than the concept of Oil itself. Or, in $Enzyme \sqsubseteq Database$, $Enzyme$ refers to an $EnzymeDatabase$ rather than describing the class of all enzymes. This last type of errors could be detected if the context of the anchor terms would be considered during matching. However, our technique does not explore the context of the used ontologies yet. This leads to another set of errors.

Errors introduced by ignoring ontological context. The current implementation of our technique works at the level of concept labels (i.e., strings) rather than concepts as defined in the context of each particular ontology. Because of their heterogeneity, online ontologies model different perspectives, different views of the world. As a result, relations between concepts are highly dependent on the context of the ontology. For example, $University \sqsubseteq Building$ is a valid relation if the first concept refers to the edifice in which courses are held, while $University \sqsubseteq Organisation$ is valid when we consider the first term in its sense of the group of people (faculty and students) involved in the educational process. A prerequisite for correct mappings is that the *anchoring* process is semantic rather than syntactic (i.e., sense(`A`) = sense(`A'`) and sense(`B`) = sense(`B'`)). Unfortunately, such semantic anchoring is rather time consuming and non trivial (as the contexts, i.e., their places in the ontologies, of all four concepts need to be considered). By ignoring the *contextual senses* of the considered terms, concepts having the same label but different meaning may be mapped, leading to common errors in the matching procedure. For example, $Squash$ can be mapped both as a $Vegetable$ or as a $Sport$. The correct mapping is the one that is in concordance with the sense of $Squash$ in the source ontology.

Finally, it is interesting to observe that these mappings were derived by exploiting more than 60 individual ontologies ranging from richly axiomatised, upper level ontologies such as SUMO (140 mappings) or TAP (303 mappings) to smaller ontologies developed for a well-defined application (e.g., a demo ontology in the domain of pizza which was used 6 times). These figures show that our technique exploits a pool of heterogeneous resources.

5.4.2 Results for strategy S1, second variant

The second variant of S1 derives all mappings by inspecting all ontologies in which the terms appear. To our surprise no contradicting mapping relations were derived. We observed, however, that some of the explored ontologies lead to no relations. It is interesting to note that a mapping is more likely to be correct when the proportion of ontologies leading to it is greater than the one of ontologies containing no relation (e.g., $Vertebrate \sqsubseteq Animal$ 3 yes/1 no, $Cat \sqsubseteq Animal$ 10 yes/1 no). Otherwise, the mapping is likely to be either incorrect or to be valid only in a particular context (e.g., $Restaurant \sqsubseteq Dimension$ 1 yes/9 no, $Fish \sqsubseteq Ingredient$ 1 yes/29 no). This information could be exploited to automatically detect suspicious mappings and therefore increase precision.

5.4.3 Results for strategy S2

Strategy S2 leads to 2 033 new mappings (1 018 disjoint and 1 015 subclass relations) each derived by combining information from two ontologies ($A \sqsubseteq P$ in one ontology and then $P' \xrightarrow{rel} B$ in another). While this strategy truly embodies the Semantic Web paradigm of combining knowledge from heterogeneous sources, the fact that it explores more ontologies also increases the chance of introducing errors. We identified three common types of errors (as exemplified in Table 5.3):

Error 1. In these cases the sense of the parent is different than that of its anchor. This type of error is a direct consequence of our technique ignoring the context of the mapped concepts.

Error 2. Some errors are introduced by ontology modelling mistakes appearing in the used ontologies (as explained in Section 5.4.1).

Type	A	$\sqsubseteq$	P/ P'	Rel	B
Correct	$Helicopter$ $FarmBuildings$ RAM $Lobster$ $RyeFlour$	$\sqsubseteq$	$Aircraft$ $Building$ $ComputerMemory$ $Crustacean$ $Flour$	$\sqsubseteq$ $\sqsubseteq$ $\sqsubseteq$ $\sqsubseteq$ $\sqsubseteq$	$Vehicle$ $Infrastructure$ $ComputerHardware$ $Arthropod$ $Powder$
Error 1	$University$ $Family$ $Tree$	$\sqsubseteq$	$Complex$ $Group$ $Plant$	$\perp$ $\sqsubseteq$ $\perp$	$Protein$ $Shape$ $ResearchAndDevelopment$
Error 2	$Radar$ $WaterManagement$ $Survey$	$\sqsubseteq$	$Equipment$ $Management$ $Marketing$	$\sqsubseteq$ $\sqsubseteq$ $\perp$	$ChemicalEngineering$ $Industry$ $Consultant$
Error 3	$Wine$ Cat $Respiration$	$\sqsubseteq$	$Liquid$ $Animal$ $Process$	$\perp$ $\perp$ $\perp$	Gas $Environment$ $Mineral$

Table 5.3: Examples of typical compound mappings.

Error 3. When the semantic gap between the term to be mapped and its parent is large (e.g., $Wall \sqsubseteq Object$), then the derived mappings, while not necessarily incorrect, are usually at a too high level of abstraction to be interesting (e.g., $Wall \perp Energy$).

While this strategy leads to a lot of interesting mappings, it also introduces a substantial amount of uninteresting (Error 3) or wrong mappings. Due to time constraints, we did not evaluate this set of mappings. Nevertheless, these experiments highlighted some common errors that need to be addressed by the next versions of our algorithm. For example, to avoid Error 3, mappings between semantically distant parents (i.e., high in the hierarchy) should not be used.

5.4.4 Comparison with other techniques

Comparing ontology matching techniques is a difficult task because the lack of an appropriate Gold Standard (since the OAEI'06 evaluation was geared towards comparing systems that only produced equality mappings, the part of the Gold Standard developed for measuring precision consisted of only 200 mappings with 70% exact and 30% subsumption relations). Therefore, a thorough comparison of the techniques, taking into account recall as well as precision, is not possible. However, looking at the results, it seems that our approach is complementary to classical, string based techniques since it returns mappings relating dissimilar terms, which are unlikely to be discovered by them. Moreover, none of the five competitors discovered semantic relations other than equivalence. Conversely, our approach does not concentrate on equivalence relations. Many of the equivalence relations discovered by traditional techniques are based on string similarity. Because the terminological overlap of AGROVOC and NALT is rather high, the techniques reached precision values ranging from 61% to 83% (on the 200 mapping Gold Standard). Note, however, that in another task of this contest, where the label similarity redundancy was more reduced (i.e., the directory task), the precision of the best system was only 40.5%. These figures reinforce the hypothesis that the performance of general matching systems is dependent on the label overlap between the source and target ontology. This factor does not influence our approach (nor context based matchers in general).

We also wished to compare the performance of our approach to that of other context-based techniques. The precision values we found in the literature were reported on different data sets, therefore we consider them only as indicative. Unfortunately, S-Match only reports on recall values [Giunchiglia *et al.*, 2006]. The technique of Aleksovski et al. was evaluated on a Gold Standard of mappings for 200 concepts and produced a precision of 76% (compared to 30% and 33% achieved by two traditional techniques on the same dataset) [Aleksovski *et al.*, 2006]. The matching techniques proposed by van Hage et al. yield a range of precision values for a manually constructed Gold Standard: 17% - 30% when relying only on Google, 38% - 50% when taking into account the context given by the Google snippets, 53% - 75% when exploring a domain specific textual resource and finally 94% when validating the results of the domain specific extraction with the Google based techniques [van Hage *et al.*, 2005]. We conclude that the 70% precision of our technique correlates with the precision results obtained by the other two techniques (76% - 75%). It is important to understand, however, that the other techniques reached these high precision values only when exploring a *single, high-quality, domain specific resource* (i.e., DICE [Aleksovski *et al.*, 2006], CooksRecipes.com Cooking Dictionary [van Hage *et al.*, 2005]). However, because such resources need to be selected in advance, these techniques do not satisfy the requirements associated with the latest generation of Semantic Web applications. Compared to them, our technique achieves comparable results when dealing with the much more demanding scenario specific to these applications: i.e., *dynamically combining multiple, heterogeneous and generic ontologies*.

5.5 Discussion

The goal of our experiments was to better understand the strengths and the weaknesses of our method. Indeed, the proposed approach exhibits all the advantages of context-based techniques over traditional syntactic techniques: it derives semantic mapping relations and it is capable of dealing with dissimilar ontologies. Even more, compared to other context-based techniques, our approach is able to discover disjoint relations (only achieved by SMatch so far). Also, a differentiating feature of our technique is its domain independence: we do not require a manual pre selection of any context-giving knowledge source (these are discovered dynamically at run-time) and our extraction rules are generally valid for any ontology instead of being biased by the type of background knowledge resource (e.g., WordNet or a particular ontology). Finally, despite the fact that we rely on the modern paradigm of combining multiple, heterogeneous knowledge sources, our precision values (70%) correlate with those obtained by techniques that exploit single, heterogeneous, carefully preselected resources (76% [Aleksovski *et al.*, 2006], 75% [van Hage *et al.*, 2005]).

While the dynamic exploration of online knowledge is the novelty of our technique, it also leads to some of its limitations. The results of the approach depend on the quality of online ontologies. In the ideal case, these should cover all the concepts that need to be matched and should not exhibit any modelling defects. Our experiments show that both conditions can be easily violated: only a small fraction of the two large thesauri were covered and our precision suffers from common modelling mistakes exhibited by online ontologies (e.g., the misuse of inheritance). We believe, however, that online coverage will improve as the Semantic Web grows. Also, our experiments indicate that automatic methods could be devised to filter out potentially erroneous relations (Section 5.4.2).

The use of multiple ontologies introduces the issue of *multiple contextual mappings* which does not appear for techniques relying on a single resource. When a single resource is used as a context for matching, then all the derived mappings share the same context. In our case however, the mappings derived for an alignment can be drawn from different context-ontologies. Ontologies often conceptualise different views of the world. As a result, some of the derived mappings, while valid in the context of their ontology, can be invalid from the perspective of the source and target ontologies. Ignoring these contexts leads to erroneous mappings. This issue became evident in this large scale matching exercise: we did not observe it in our small scale experiments [Sabou *et al.*, 2006]. The problem intensifies for more complex strategies (those that rely on a chain of mappings).

Chapter 6

Conclusions

This deliverable has introduced software developed in NeOn for dealing with contexts in the framework of networked ontologies. It has first shown the use of alignments for expressing the context of an ontology with regard to other related ontologies.

It has described a software system for maintaining and sharing these alignments within the NeOn environment. The Alignment Server presented in Chapter 4 provides access to various matching methods in order to match ontologies and many alignment manipulation facilities. It is able to store alignments in a permanent repository, to deliver them on demand in various useful formats. Interaction through the Alignment Server can be achieved through agent communication, web service invocation or web browser interaction.

In Chapter 5, we presented an approach relying on contexts by automatically discovering the context in which the relations between these entities are described, this context taking the form of a third ontology containing the relevant background knowledge. This technique has been implemented using Swoogle to find ontologies relating the candidate terms. An evaluation using two real-life, large scale ontologies (AGROVOC and NALT) has shown particularly good results in terms of precision on a large number of generated mappings (around 1 650).

6.1 Current state of the prototypes

The developed prototypes are currently independent and functional. They offer all the functionalities that have been described in this deliverable and are continuously evolving for meeting the needs of the NeOn environements.

The Alignment Server is available within the Alignment API at http://alignapi.gforge.inria.fr starting with version 2.5 (december 2006). It is packaged with the Alignment API releases. Instructions for its use are available in Appendix A.

The context matcher prototype was developed in order to test the overall idea of using dynamically selected background knowledge from online ontologies. Because it was used for testing by the development team, the prototype has an API but not a user interface. In fact, this is all is needed for its integration in the Alignment Server. Further, this prototype relies on Swoogle'2005. This version of Swoogle is not updated anymore and probably will not be supported within a couple of months. Therefore our efforts are on rewritting it so that it exploits Watson. Our current work focuses on integrating the existing code with the currently emerging Watson API. We expect to have a new version within the next two months which can be made available publicly either as 1) a stand alone software or 2) a service provided by Watson. It will, as well as other matching systems, be integrated within the Alignment Server and will be available through it.

These prototypes will be demonstrated at the review meeting in a loosely coupled manner.

6.2 Further developments

These prototypes are independent efforts for developing the technology adapted to networked ontologies as considered in NeOn. As such they have shown the feasibility of the approaches that have been pursued.

However, they have now to be more tightly integrated in the wider context of NeOn. This is the role of the work that we start carrying out now. One of the obvious next steps for the development of this technique concerns its integration in the alignment server presented Chapter 4. Mappings from the reported experiment can be stored and made available on the server, and the technique would benefit from relying on the proposed alignment API. Also, the technique suffers from the limitation inherent to Swoogle. We are currently developing a Semantic Web Gateway called Watson, as an alternative to Swoogle, with the aim of overcoming these limitations. The evolution of the presented technique should then rely on Watson and on the Alignment Server for discovering relevant context to be used in matching.

This precisely involves:

- integrating the contextual matcher within the Alignment Server;
- integrating the prototype for extracting alignments from data (see Deliverable D3.2.1) within the Alignment Server;
- organising the cooperation between Watson and the Alignment Server for finding related ontologies;
- integrating the Alignment Server as a plug-in of the NeOn toolkit;
- using these tools in tasks directly related to the NeOn applications.

Appendix A

Running the Alignment Server

We provide here the minimal instructions for obtaining and interacting with the Alignment Server. The most recent information can be found on the Alignment API web site (http://alignapi.gforge.inria.fr).

A.1 Obtaining the Alignment Server

The Alignment Server can be downloaded with the Alignment API from http://gforge.inria.fr/frs/?group_id=117. The server is included with the API starting with version 2.5.

A.2 Requirements

Java 1.5 The Alignment Server runs under Java 1.5 or higher.

DBMS Using the alignment server requires an SQL database server. We see here, how to use mysql (http://dev.mysql.com/doc/refman). mysql drivers for jdbc are included in the package. If you want to use another SQL database management system, you only have to put the jdbc drivers for this database in the lib directory.

TCP ports The alignment server is a communicating system that communicates through TCP sockets which are bound to ports on your machines. In order to access the server from an outside machine it is necessary to have access to these ports. We provide here the list of default ports and options to change them as well as the necessity for the firewalls to open these ports:

plug-in	default	option	open?
HTTP	8089	-H	Y
Jade	8888	-A	
	1099		? (RMI)
	7778		Y (MTP HTTP)
WSDL	8089	-W	Y
JXTA	6666	-P	Y
mysql	3306	–dbmsport	No if on the same machine

Of course, the ports need only to be open if there is an access from the outside to the server with the corresponding plug-in.

A.3 Initializing the database

In order to use the Alignment Server, it is necessary to create its database. This can simply be done by the following shell instructions.

```
$ sudo /usr/local/mysql/bin/mysqld_safe --user=mysql &
$ /usr/local/mysql/bin/mysqladmin -u root password <mysqlpassword>
$ /usr/local/mysql/bin/mysqladmin -u root -h localhost password <mysqlpassword>
$ /usr/local/mysql/bin/mysql -u root -p<mysqlpassword>
sql> CREATE DATABASE AServDB;
sql> GRANT ALL PRIVILEGES ON AServDB.* TO adminAServ@localhost IDENTIFIED BY 'aaa345';
sql> quit
```

The database schema will be created upon the first launch of the server.

Of course, you are advised to use different user, password and database name. This can be achieved either:

– by changing values of DBMSBASE, DBMSUSER and DBMSPASS in AlignmentServer.java and re-compiling;
– by passing parameters dbmsbase, dbmsuser and dbmspass to the AlignmentServer main.

If one wants to permanently, run an Alignment Server, here is a sample backup of the server content:

```
$ /usr/local/mysql/bin/mysqldump -u adminAServ -paaa345 AServDB > toto.sql
```

And restoring:

```
$ /usr/local/mysql/bin/mysql -u adminAServ -paaa345 AServDB
sql> source toto.sql;
```

NeOn will run such a permanent server so setting up the database will not be necessary.

A.4 Launching the Alignment Server

The Alignment Server requires that the corresponding database management system server be running. In our case, this can be achieved through:

```
$ sudo /usr/local/mysql/bin/mysqld_safe --user=mysql &
```

Running the Alignment server is achieved through (use the corresponding options):

```
$ java -jar lib/alignsvc.jar -H
```

The alignment server is then available through HTTP through http://localhost:8089/html/.

Other options are:

```
$ java -jar lib/alignsvc.jar --help
usage: AlignmentService [options]
options are:
        --html[=port] -H [port]                 Launch HTTP service
        --jade[=port] -A [port]                 Launch Agent service
        --wsdl[=port] -W [port]                 Launch Web service
        --jxta[=port] -P [port]                 Launch P2P service
        --serv=class -i class                   Launch service corresponding to fully qualified clas
        --dbmshost=host -m host                 Use DBMS host
        --dbmsport=port -s port                 Use DBMS port
        --dbmsuser=name -u name                 Use DBMS user name
        --dbmspass=pwd -p pwd                   Use DBMS password
        --dbmsbase=name -b name                 Use Database name
        --debug[=n] -d [n]              Report debug info at level n
        -Dparam=value                   Set parameter
        --help -h                       Print this message
```

Bibliography

[Aleksovski *et al.*, 2006] Zharko Aleksovski, Michel Klein, Warner ten Kate, and Frank van Harmelen. Matching unstructured vocabularies using a background ontology. In S. Staab and V. Svatek, editors, *Proc. EKAW, Praha (CZ)*, LNAI. Springer-Verlag, 2006.

[Bouquet *et al.*, 2004a] P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini, and H. Stuckenschmidt. Contextualizing ontologies. *Journal of Web Semantics*, 1(4):24, 2004.

[Bouquet *et al.*, 2004b] Paolo Bouquet, Marc Ehrig, Jérôme Euzenat, Enrico Franconi, Pascal Hitzler, Markus Krötzsch, Luciano Serafini, Giorgos Stamou, York Sure, and Sergio Tessaris. Specification of a common framework for characterizing alignment. Deliverable D2.2.1, Knowledge web NoE, 2004.

[Bouquet *et al.*, 2005] P. Bouquet, L. Serafini, and S. Zanobini. Peer-to-Peer Semantic Coordination. *Journal of Web Semantics*, 2(1), 2005.

[Ding *et al.*, 2005] L. Ding, R. Pan, T. Finin, A. Joshi, Y. Peng, and P. Kolari. Finding and Ranking Knowledge on the Semantic Web. In *Proc. of ISWC*, 2005.

[Ehrig, 2007] Marc Ehrig. *Ontology alignment: bridging the semantic gap*. Semantic Web and Beyond: Computing for Human Experience. Springer, New-York (NY US), 2007.

[Euzenat and Shvaiko, 2007] Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*. Springer, Heidelberg (DE), 2007. to appear.

[Euzenat and Valtchev, 2004] Jérôme Euzenat and Petko Valtchev. Similarity-based ontology alignment in OWL-lite. In *Proc. 15th European Conference on Artificial Intelligence (ECAI)*, pages 333–337, Valencia (ES), 2004.

[Euzenat *et al.*, 2004] Jérôme Euzenat, Thanh Le Bach, Jesús Barrasa, Paolo Bouquet, Jan De Bo, Rose Dieng-Kuntz, Marc Ehrig, Manfred Hauswirth, Mustafa Jarrar, Rubén Lara, Diana Maynard, Amedeo Napoli, Giorgos Stamou, Heiner Stuckenschmidt, Pavel Shvaiko, Sergio Tessaris, Sven Van Acker, and Ilya Zaihrayeu. State of the art on ontology alignment. Deliverable D2.2.3, Knowledge web NoE, 2004.

[Euzenat *et al.*, 2005a] J. Euzenat, H. Stuckenschmidt, and M. Yatskevich. Introduction to the Ontology Alignment Evaluation 2005. In *Proc. of the Integrating Ontologie WS*, 2005.

[Euzenat *et al.*, 2005b] Jérôme Euzenat, Loredana Laera, Valentina Tamma, and Alexandre Viollet. Negotiation/argumentation techniques among agents complying to different ontologies. Deliverable 2.3.7, Knowledge web NoE, 2005.

[Euzenat *et al.*, 2006] J. Euzenat, M. Mochol, P. Shvaiko, H. Stuckenschmidt, O. Svab, V. Svatek, W. R. van Hage, and M. Yatskevich. Results of the Ontology Alignment Evaluation Initiative 2006. In *Proc. of the ISWC Ontology Matching WS*, 2006.

[Euzenat, 2004] Jérôme Euzenat. An API for ontology alignment. In *Proc. 3rd International Semantic Web Conference (ISWC)*, volume 3298 of *Lecture notes in computer science*, pages 698–712, Hiroshima (JP), 2004.

[Gangemi *et al.*, 2003] Aldo Gangemi, Nicola Guarino, Claudio Masolo, and Alessandro Oltramari. Sweetening WordNet with DOLCE. *AI Magazine*, 24(3):13–24, 2003.

[Gangemi, 2004] Aldo Gangemi. Restructuring semi-structured terminologies for ontology building: a realistic case study in fishery information systems. Deliverable D16, WonderWeb, 2004.

[Giunchiglia and Shvaiko, 2003] Fausto Giunchiglia and Pavel Shvaiko. Semantic matching. *The Knowledge Engineering Review (KER)*, 18(3):265–280, 2003.

[Giunchiglia *et al.*, 2006] F. Giunchiglia, P. Shvaiko, and M. Yatskevich. Discovering Missing Background Knowledge in Ontology Matching. In *Proc. of ECAI*, 2006.

[Haase *et al.*, 2006a] Peter Haase, Pascal Hitzler, Sebastian Rudolph, Guilin Qi, Marko Grobelnik, Igor Mozetič, Damjan Bojadžiev, Jerome Euzenat, Mathieu d'Aquin, Aldo Gangemi, and Carola Catenacci. Context languages – state of the art. Deliverable D3.1.1, NeOn, 2006.

[Haase *et al.*, 2006b] Peter Haase, Sebastian Rudolph, Yimin Wang, Saatje Brockmans, Raúl Palma, Jerome Euzenat, and Mathieu d'Aquin. Networked ontology model. Deliverable D1.1.1, NeOn, 2006.

[Jian *et al.*, 2005] Ningsheng Jian, Wei Hu, Gong Cheng, and Yuzhong Qu. Falcon-AO: Aligning ontologies with falcon. In *Proc. K-CAP Workshop on Integrating Ontologies*, pages 87–93, Banff (CA), 2005.

[Kalfoglou and Schorlemmer, 2003] Yannis Kalfoglou and Marco Schorlemmer. Ontology mapping: the state of the art. *The Knowledge Engineering Review (KER)*, 18(1):1–31, 2003.

[Lenat and Guha, 1990] Doug Lenat and Ramanathan V. Guha. *Building large knowledge-based systems*. Addison Wesley, Reading (MA US), 1990.

[Lopez *et al.*, 2006] Vanessa Lopez, Enrico Motta, and Victoria Uren. PowerAqua: Fishing the semantic web. In York Sure and John Domingue, editors, *Proc. 3rd European Semantic Web Conference (ESWC)*, volume 4011 of *Lecture notes in computer science*, pages 393–410, Budva (ME), 2006.

[Melnik *et al.*, 2002] Sergey Melnik, Hector Garcia-Molina, and Erhard Rahm. Similarity flooding: a versatile graph matching algorithm. In *Proc. 18th International Conference on Data Engineering (ICDE)*, pages 117–128, San Jose (CA US), 2002.

[Niles and Pease, 2001] Ian Niles and Adam Pease. Towards a standard upper ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS)*, pages 2–9, 2001.

[Rahm and Bernstein, 2001] Erhard Rahm and Philip Bernstein. A survey of approaches to automatic schema matching. *The VLDB Journal*, 10(4):334–350, 2001.

[Sabou *et al.*, 2006] M. Sabou, M. d'Aquin, and E. Motta. Using the Semantic Web as Background Knowledge for Ontology Mapping. In *Proc. of ISWC Ontology Matching WS*, 2006.

[Tang *et al.*, 2006] Jie Tang, Juanzi Li, Bangyong Liang, Xiaotong Huang, Yi Li, and Kehong Wang. Using bayesian decision for ontology mapping. *Journal of Web Semantics (JWS)*, 4(1):243–262, 2006.

[van Hage *et al.*, 2005] W. van Hage, S. Katrenko, and G. Schreiber. A Method to Combine Linguistic Ontology-Mapping Techniques. In *Proc. of ISWC*, 2005.

[Wache *et al.*, 2001] Holger Wache, Thomas Voegele, Ubbo Visser, Heiner Stuckenschmidt, Gerhard Schuster, Holger Neumann, and Sebastian Hübner. Ontology-based integration of information – a survey of existing approaches. In *Proc. IJCAI Workshop on Ontologies and Information Sharing*, pages 108–117, Seattle (WA US), 2001.