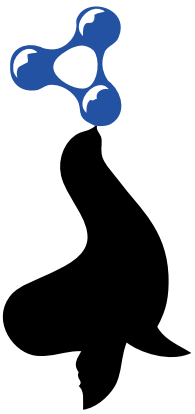
Evaluation design and collection of test data for matching tools

Cassia Trojahn dos Santos, Jérôme Euzenat, Christian Meilicke, Heiner Stuckenschmidt



SEALS

Semantic Evaluation at Large Scale

FP7 – 238975

D12.1 Evaluation Design and Collection of Test Data for Matching Tools

Coordinator: Cássia Trojahn dos Santos
With contributions from: Jérôme Euzenat, Christian Meilicke, Heiner Stuckenschmidt
Quality Controller: Raúl García Castro
Quality Assurance Coordinator: Raúl García Castro

Document Identifier:	SEALS/2009/D12.1/V1.0
Class Deliverable:	SEALS EU-IST-2009-238975
Version:	version 1.0
Date:	November 12, 2009
State:	final
Distribution:	public

Executive Summary

The goal of Work Package 12 is to provide the infrastructure for evaluating ontology matching systems and algorithms, to be aggregated in the SEALS platform. The objective of this deliverable is to document the first step of applying the SEALS evaluation methodology (§1) by identifying and discussing goals and assumptions (§1), criteria and metrics (§3) as well as datasets (§4) and tools (§5) for the SEALS evaluation campaigns, focusing on the first campaign to be held in autumn 2010.

The first evaluation campaign has as goal to evaluate the competence of matching systems with respect to isolated aspects and to compare matching systems on single criteria. For this purpose, simple evaluations will be implemented that can apply a single matching system on a single criterion and store the result for further aggregation with other results. Assumptions for this first campaign are that the matching systems can run independently and that it is possible and useful to compare systems based on different criteria separately.

We present a comprehensive review on evaluation criteria (§3) and decide which ones should be considered in the first campaign. A limited set of criteria will be used that can be tested using simple workflows (§2) as described in this deliverable. Criteria and measures to be considered are:

- Efficiency: runtime, memory consumption;
- Interoperability: compliance to the standard language RDFS and OWL-DL;
- Conformance: standard precision and recall, restricted semantic precision and recall, coherence.

We have selected a subset of the datasets and systems that have been involved in previous OAEI campaigns. The datasets were selected based on the existence of reliable reference alignments and experiences with using the datasets in evaluation campaigns. These criteria are met by the following datasets (§4): Benchmark, Anatomy, and Conference.

Tools have been selected based on maturity and availability. Based on these criteria, we have identified the following tools as natural candidates to participate in the first evaluation campaign (§5): ASMOV, Falcon-OA, SAMBO, Lily, and AROMA.

Finally, we discuss (§6) how to display and manipulate the evaluation results of the first campaign, taking as basis OAEI evaluation reports.

Document Information

IST Project Number	FP7 – 238975	**Acronym**	SEALS
Full Title	Semantic Evaluation at Large Scale		
Project URL	`http://www.seals-project.eu/`		
Document URL			
EU Project Officer	Carmela Asero		

Deliverable	**Number**	12.1	**Title**	Evaluation Design and Collection of Test Data for Matching Tools
Work Package	**Number**	12	**Title**	Matching Tools

Date of Delivery	**Contractual**	M6	**Actual**	12-11-09
Status	version 1.0		final ☒	
Nature	prototype ☐ report ☒ dissemination ☐			
Dissemination level	public ☒ consortium ☐			

Authors (Partner)	Jérôme Euzenat (INRIA), Cássia Trojahn dos Santos (INRIA), Heiner Stuckenschmidt (University Mannheim), Christian Meilicke (University Mannheim)			
Resp. Author	**Name**	Jérôme Euzenat	**E-mail**	jerome.euzenat@inrialpes.fr
	Partner	INRIA	**Phone**	+33 (476) 615 476

Abstract (for dissemination)	This deliverable presents a systematic procedure for evaluating ontology matching systems and algorithms, in the context of SEALS project. It describes the criteria and metrics on which the evaluations will be carried out and the characteristics of the test data to be used, as well as the evaluation target, which includes the systems generating the alignments for evaluation.
Keywords	ontology matching, ontology alignment, evaluation, benchmarks, efficiency measure

Version Log			
Issue Date	**Rev No.**	**Author**	**Change**
24/09/2009	1	Jérôme Euzenat	Added Chapter 6
07/10/2009	2	Cassia	Added Section 1.2 and description of basic BPEL process
08/10/2009	3	Cassia	Added advanced BPEL process and preliminary summary sections
12/10/2009	4	Cassia	Added Section 3.1.4
13/10/2009	5	Christian	Added appendix II (incomplete) and III
13/10/2009	6	Cassia	Added appendix I and reviewed sections (1 – 4.1)
19/10/2009	7	Christian	Revision of Section 3, 4, 5
19/10/2009	8	Heiner	Revision of Document, added description of restricted semantic precision and recall, added Summary
20/10/2009	9	Christian	Completed appendix II w.r.t current information status, revision of Section 2,3
20/10/2009	10	Cassia	Added new executive summary and aggregated comments from Jérôme
06/11/2009	11	Cassia	Implemented modifications as suggestions from Raúl

Project Consortium Information

Participant's name	Partner	Contact
Universidad Politécnica de Madrid		Asunción Gómez-Pérez Email: asun@fi.upm.es
University of Sheffield	The University Of Sheffield.	Fabio Ciravegna Email: fabio@dcs.shef.ac.uk
Forschungszentrum Informatik	FZI	Rudi Studer Email: studer@aifb.uni-karlsruhe.de
University of Innsbruck	STI · INNSBRUCK	Barry Norton Email: barry.norton@sti2.at
Institut National de Recherche en Informatique et en Automatique	INRIA	Jérôme Euzenat Email: Jerome.Euzenat@inrialpes.fr
University of Mannheim	UNIVERSITÄT MANNHEIM	Heiner Stuckenschmidt Email: heiner@informatik.uni-mannheim.de
University of Zurich	University of Zurich Department of Informatics DIS Dynamic and Distributed Information Systems	Abraham Bernstein Email: bernstein@ifi.uzh.ch
Open University	The Open University	John Domingue Email: j.b.domingue@open.ac.uk
Semantic Technology Institute International	STI · INTERNATIONAL	Alexander Wahler Email: alexander.wahler@sti2.org
University of Oxford	UNIVERSITY OF OXFORD	Ian Horrocks Email: ian.horrocks@comlab.oxford.ac.uk

Table of Contents

List of Figures

List of Tables

1. Introduction

Matching ontologies consists of finding corresponding entities in different ontologies. Many different techniques have been proposed for implementing this process. They can be classified along the many features that can be found in ontologies (labels, structures, instances, semantics), or with regard to the kind of disciplines they belong to (e.g., statistics, combinatorics, semantics, linguistics, machine learning, or data analysis) [37, 26, 19].

An alignment (set of correspondences) is obtained by combining these techniques towards a particular goal (obtaining an alignment with particular features, optimizing some criterion, etc). Several combination techniques are also used. The increasing number of methods available for ontology matching suggests the need to establish a consensus for evaluating these methods.

More specifically, an alignment can be characterized as a set of pair of entities (e and e'), coming from each ontologies (o and o'), related by a particular relation (r). To this, many algorithms add some confidence measure (n) expressing a degree of trust in the fact that the relation holds [11, 4, 12]. From this characterization it is possible to ask any alignment method, given:

- two ontologies to be aligned,
- a partial input alignment (possibly empty),
- a characterization of the wanted alignment (e.g. one-to-one vs. many-to-many alignments).

to output an alignment.

From this output, the quality of the alignment process could be assessed with the help of some measurement. However, very few experimental comparisons of algorithms are available. Although OAEI campaigns have already created an initial basis for evaluation that did not exist before, more progress in leveraging increased evaluation efforts has to be made in order to continue the growth of ontology matching technology. The objective of WP12 is to design principled and reproducible evaluation techniques for complex and large matching tasks.

The goal of this chapter is to present the general objective of evaluating matching systems (Section 1.1) and the overall methodology to be followed in evaluation campaigns (Section 1.2).

1.1 Purposes of Evaluation

The major and long term purpose of the evaluation of ontology alignment methods is to help designers and developers of such methods to improve them and to help users to evaluate the suitability of proposed methods to their needs. The SEALS platform will also be a means to inform industry users about available techniques. This requires both a theoretically well-founded and application oriented evaluation approach, which focuses on different aspects related to ontology matching.

The medium term goal of this work is to set up a set of reference benchmark tests for assessing the strengths and weaknesses of the available tools and to compare

them. Some of these tests are focusing the characterization of the behavior of the tools rather than having them compete on real-life problems. It is expected that they could be improved and adopted by the algorithm implementers in order to situate their algorithms. Building benchmark suites is highly valuable not just for the group of people that participates in the contests, but for all the research community.

The shorter term goal is to illustrate how it is possible to evaluate ontology alignment tools and to show that it is possible to build such an evaluation campaign. It is a common subgoal of an evaluation campaign that a regular occurrence helps improving the evaluation methodologies.

1.2 Evaluation Methodology

Evaluation is a continuous process that should be performed regularly in order to obtain a continuous improvement both in the tools and in the evaluation process itself. In SEALS, a revised version of the Knowledge Web benchmarking methodology [6], which is composed by three steps (Plan, Experiment, and Improve), is proposed (more details in the deliverable D3.1). The SEALS evaluation methodology can be described as an iterative process that is composed by five phases (Preliminary, Design, Execution, Validation, and Analysis) and ends with an improvement task.

The five phases of each iteration are the following:

Preliminary Prior to the design of the evaluations the following elements must be identified: evaluation goals, assumptions, criteria, metrics, list of suitable tools to be evaluated, features of these tools to be evaluated, and test data.

Design In this step, all details of the evaluations suggested by the members of the corresponding Work Package must be provided.

Execution In this step, the following items must be decided: the application and components required for testing the tools (e.g. a component for storing the result alignment, a component for measuring time of generating the result alignment, etc), and APIs required to access to the tools to be evaluated.

Validation Explain how the results generated by the evaluations should be validated.

Analysis Describe the techniques to analyze and interpret the results of the evaluations (e.g. applications for comparing the reference alignment with the alignment result, generation of interpretations based on the evaluation results).

While the five phases mentioned before are devoted to the tool improvement, the goal of the Improvement (or re-calibration) task is to improve the evaluation process itself after each iteration, using the lessons learned while performing the evaluation.

1.3 Summary

In this chapter, the purposes for evaluating matching systems and the kind of evaluations that will be performed were presented. The overall methodology to be followed

in SEALS evaluation campaigns was also discussed. This deliverable covers the first step of this methodology (referred to as 'Preliminary') with respect to benchmarking ontology matching tools.

In the next chapter we evaluate the variability in the alignment task, and, consequently, define the parameters that must be controlled in an evaluation. It presents the evaluation workflows, which specify what happens in a matching evaluation experiment. Chapter 3 considers the potential evaluation criteria and corresponding metrics that can be used in order to assess the matching algorithms.

2. Evaluation Workflow

The goal of this chapter is to present the dimensions and variability of alignment evaluation and some evaluation workflows representing the sequence of activities carried out in an evaluation experiment. First, the process of matching is described in detail, in order to characterize the variability of the alignment task and to know what variables must be controlled during the design of benchmarks (Section 2.1). Next, different evaluations workflows determine the way in which an evaluation experiment is conducted in terms of its input, output and relevant operations (Sections 2.2 and 2.3).

2.1 Matching Process

The matching process consists of generating an alignment (A') from a pair of ontologies (o and o'). Despite this general definition, there are various other parameters which can extend the definition of the matching process. These are the use of an input alignment (A) which is to be completed by the process, the alignment methods parameters (which can be weights for instance) and some external resources used by the alignment process (which can be general-purpose resources not made for the case under consideration, e.g., lexicons, databases) [14]. This process can be defined as follow:

Definition 1 (Matching process) *The matching process can be seen as a function f which, from a pair of ontologies o and o' to align, an input alignment A, a set of parameters p, and a set of oracles and resources r, returns a new alignment A' between these ontologies:*

$$A' = f(o, o', A, p, r)$$

This can be represented as in Figure 2.1.

Each of the elements featured in this definition can have specific characteristics which influence the difficulty of the alignment task. It is thus necessary to know and control these characteristics (called dimensions because they define a space of possible tests). The purpose of the dimensions is the definition of the parameters and characteristics of expected behavior in a benchmark experiment. In the following, such dimensions are detailed.

2.1.1 Input ontologies

Input ontologies (o, o') can be characterized by at least nine dimensions:

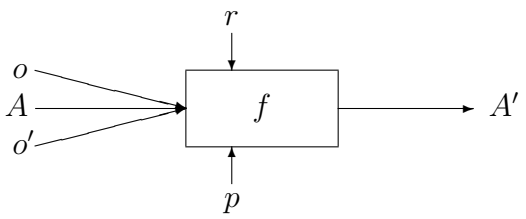


Figure 2.1: The matching process (from [14]).

Heterogeneity of the input languages: are they described in the same knowledge representation languages (e.g. OWL-Lite, OWL-DL, OWL-Full), which syntax representation is used?

Languages: In which language are the labels of the ontologies described? Labels might be described in common speech but in different languages, or they might be described in specialized technical terms (e.g. technical product catalogs, biomedical ontologies).

Number: Is the alignment an alignment between two ontologies or a multi-alignment that connects more than two ontologies?

Size: How many concepts, properties and instances do the ontologies contain?

Expressivity: Aside from the representation language, ontologies might vary with respect to their DL-expressivity (e.g. $\mathcal{SHIN}$, $\mathcal{ALCHIF}$).

Complexity: How deep is the hierarchy structured and how strong is the interconnection between ontological entities.

Consistency: Are the ontologies consistent? This might in particular affect matching tools using reasoning components.

Correctness: Are there modeling errors in the ontologies? Real-world ontologies will often contain some incorrect axioms.

Completeness: Have all relations (e.g. subsumption, disjointness, property restrictions) been modeled in detail? Are the ontologies a fine-grained and complete descriptions of the represented domain?

In the setting up of a particular test, it is necessary to decide for the use of a formal language. In SEALS, it is supposed to consider alignments of ontologies expressed in the same representation language.

Tasks involving multi-alignment are very specific. Usually matching is triggered by editors that want to expand an ontology or web services to compose. This involves the alignment of two ontologies. Bringing other ontologies in the process does not help solving the problem. Multi-alignment is rather reserved to ontology normalization or mining. For the moment it seems preferable to consider only two ontologies to align. This should hold until competitors complain that multi-alignment would be worthwhile.

2.1.2 Input alignment

The input alignment (A) can have the following characteristics:

Multiplicity: How many entities of one ontology can correspond to one entity of the others? (see "Output alignment").

Completeness: The input alignment can be empty, can contain only few correspondences or nearly all correct correspondences.

Coverage: Even a complete input alignment can nevertheless only cover a small fraction of the ontologies to be aligned. This is based on the fact that ontologies might cover different, only partially overlapping domains.

Correctness: The input alignment might contain some erroneous correspondences, in particular when it is used to simulate user input.

Relations: (see "Output alignment").

The input alignment may vary with respect to these dimensions. However, in the simple scenarios the input alignment will be empty. In the first evaluation campaign we will therefore also use empty input alignments in most evaluations.

2.1.3 Parameters

Parameters (p, r) of the alignment process can be identified as:

Oracles/resources: Are oracles authorized? If so, which ones (the answer can be any)? Is human input authorized?

Training: Can training be performed on a sample?

Proper parameters: Are some parameters necessary? And which are they? This point is quite important when a method is very sensitive to the variation of parameters. A good tuning of these must be available.

Many systems take advantage of some external resources such as WordNet, sets of morphological rules or a previous alignment of general purpose catalogs (Yahoo and Google for instance). It is possible to use these resources as long as they have not been tuned to the task for the current benchmark (for instance, using a sub-lexicon which is dedicated to the domain considered by the tests). It is acceptable that the algorithms prune or adapt these resources to the actual ontologies as long as this is in the normal process of the algorithm. However this processing time must be considered within the running time of the algorithm.

In general, if human input is provided, the efficiency of systems can be expected to be better. In the current state, which is the absence of any consensus or valuable methods for handling and evaluating the contribution of this human input, this should be not taken into account in a first step.

Training on some samples is very often used by methods for matching ontologies and mapping schemas. However, this training sample is a particular alignment. The only situation in which this makes a lot of sense is when a user provides some example of aligned instances and the system can induce the alignment from this. This is thus quite related to user input. It should be an interesting characteristics to be considered in a second step.

Some parameters can be provided to the methods participating in the evaluation. However, these parameters must be the same for all tests. It can be the case that some methods are able to tune their parameters depending on the presented ontologies. In such a case, the tuning process is considered part of the method. However, this process

must be computed from the ontology input only, not from externally provided expected results.

It seems necessary, in competence benchmark, to have participants providing the best parameter set they found for the benchmark. This set must be the same for all tests. In competitive tests, especially when the expected result is not known from the participants, they will not change their parameters. However, auto tuning algorithms are perfectly acceptable.

2.1.4 Output alignment

The following possible constraints on the output alignment (A') of the algorithm can be identified:

Multiplicity: How many entities of one ontology can correspond to one entity of the others? Usual notations are 1:1, 1:m, n:1 or n:m. We prefer to note if the mapping is injective, surjective and total or partial on both side. We then end up with more alignment arities (noted with, 1 for injective and total, ? for injective, + for total and * for none and each sign concerning one mapping and its converse): ?:?, ?:1, 1:?, 1:1, ?:+, +:?, 1:+, +:1, +:+, ?:*, *:?, 1:*, *:1, +:*, *:+, *:*. These assertions could be provided as input (or constraint) for the alignment algorithm or be provided as a result by the same algorithm.

Justification: Is a justification of the results provided?

Relations: Should the relations involved in the correspondences be only equivalence relations or could they be more complex?

Strictness: Can the result be expressed with trust-degrees different than $\top$ and $\bot$ or should they be strictified before?

In real life, there is no reason why two independently developed ontologies should have a particular alignment multiplicity other than *:*. This should be the (non) constraint on the output alignment of the benchmark tests. However, if we say so and all our tests provide some particular type of alignment, it can be said that this introduces a bias. This bias can be suppressed by having each type of alignment equally represented. However, this is not easy to find and this is not realistic. What would be realistic would be to have a statistical evaluation of the proportion of each type of alignment. In the absence of such an evaluation, however, it remains reasonable to stick to the *:* rule. This could be revised later on.

Another worthwhile feature for users is the availability of meaningful explanations or justifications of the correspondences. However, very few algorithms are able to deliver them and there is no consensus either on the form in which they are expressed neither on the way to compare them. So, it is currently not possible to ask for explanations in the benchmark results.

All algorithms deliver pairs of entities (correspondences). However, some of them associate a relation between the entities different from equivalence (e.g., specificity) and some of them associate a strength to the correspondence (which can be a probability measure). A problem is that not all algorithms deliver the same structure.

Moreover, alignments must be used in tasks for which, most of the time it is necessary to know how to interpret a term of one ontology with regard to another ontology. For these reasons, and because each method can, at least, deliver equivalence statements with the maximum strength, in the first evaluation it seems better to avoid using any kind of other relation or measure (more exactly, to design the tests with alignments involving only equivalence relations and $\top$ confidence measure).

2.1.5 Matching process

The matching process (f) itself can be constrained by:

Resource constraints: Is there a maximal amount of time or space available for computing the alignment?

Language restrictions: Is the mapping scope limited to some kind of entities (e.g., only T-box, only classes)?

Property: Must some property be true of the alignment? For instance, one might want that the alignment be a consequence of the combination of the ontologies (i.e., $o, o' \models A'$) or that alignments preserve consequences (e.g., $\forall \phi, \phi' \in L, \phi \models \phi' \implies A'(\phi) \models A'(\phi')$) or that the initial alignment is preserved (i.e., $o, o', A' \models A$).

Resource constraints can be considered either as a constraint (the amount of resource is limited) or a result (the amount consumed is measured – see Chapter 3). It is a relatively important factor, at least for efficiency tests and must be measured. This can also be measured for competence tests (even if it is absolutely difficult to do because of the heterogeneity of the environments in which these algorithms can be run).

Constraints on the kind of language construct to be found in mappings can be designed. However, currently very few alignment algorithms can align complex expressions, most of them align the identified (named) entities and some of them are only restricted to concepts. With regard to its importance and its coverage by current alignment systems, it makes sense to ask for the alignment of named entities and consider complex expressions later.

The properties of the alignments provided by the alignment algorithms are not very often mentioned and they seem to be very heterogeneous depending of the implemented techniques. It seems thus difficult to ask for particular properties. As for the type of alignment, not asking for a property is a problem if the tests do not satisfy a variety of properties. Moreover, it is not obvious that in real life, there are any properties to be satisfied by alignments (because ontologies are made for different purposes). So, at this stage, we do not commit to a particular property.

In the next two sections, evaluation workflows representing the interaction between the components commented above are presented. They specify the sequence of activities carried out by matching and evaluation components.

2.2 Basic Evaluation Workflow

The basic evaluation workflow shows the interaction between components in a minimal evaluation setting. We restrict the evaluation experiment to one process that evaluates the compliance of one alignment with respect to a reference alignment (Chapter 3).

Figure 2.2[1] shows the basic evaluation workflow specified using BPEL (Business Process Execution Language [1]). BPEL defines a model and a grammar for describing the behavior of a business process based on interactions between the process itself and its partners [1]. The interaction with each partner occurs through Web Service interfaces, and the structure of the relationship at the interface level is encapsulated in what is called a partner link. A BPEL process defines how multiple service interactions with these partners are coordinated. BPEL standard seems to be suitable for describing the interaction between the several elements in a matching evaluation experiment.

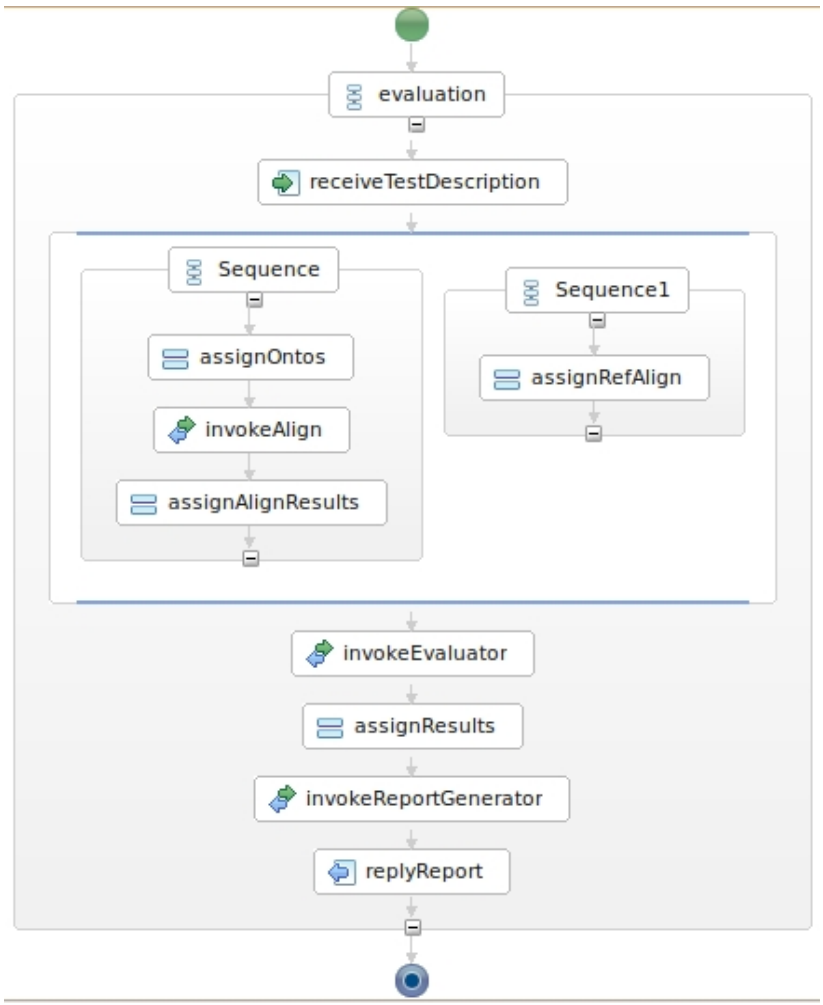


Figure 2.2: Basic BPEL evaluation workflow.

[1]Source code available on http://www.seals-project.eu/wiki/index.php/Example_workflows

The minimal interaction in an evaluation experiment is illustrated in Figure 2.2. The "evaluation" process starts by receiving from a third process the evaluation test description (information about the ontologies to be used, reference alignment, and matcher to be invoked). Such description is then used to initialize different variables in the process (*ontos* – two ontologies to be matched – and *refAlign*). This is done by an "assign" activity (*assignOntos* and *assignRefAlign*). The different assign activities are executed in independent sequences within the process.

Following the first sequence, a call to the service representing the matcher is performed by an "invoke" activity (*invokeAlign*), that returns the alignment between the input ontologies. Note that a matcher is seen as a partner link. The resulting alignment together with the reference alignment are used as input to invoke the evaluator process (partner link *evaluator*), that returns the evaluation results. These results are then assigned to a variable (*assignResults*), that will be used as input for the report generator service (partner link *ReportGenerator*). Finally, one report containing the interpretation of the alignment results is sent as reply to the process that had invoked the evaluation process.

2.3 Advanced Evaluation Workflow

The evaluation workflow above reflects the most basic sequence of activities in an evaluation experiment (one matcher and one test case). Due to the variability of alignment evaluation, different scenarios can be specified, by adding new components to the basic workflow:

Test generator can be used to generate test cases from a description of the kind of evaluation to be executed (for example, removing n% of the properties of the ontologies). A description of the desired test case must be provided and the output of the test generator service is then used as input to the matching process.

Batch tests Usually, a matcher is evaluated using a set of tests. In this way, an iterative activity must be provided to iterate each test. Moreover, several matchers can be evaluated in one evaluation experiment, what requires an iterative activity for the set of matchers.

No reference alignment It is not the case that all test cases have a complete reference alignment and alternative metrics of evaluation must be provided, such as measuring the consensus between the several matchers, intersection or union of results, and so on.

Usually, these components are combined together. For instance, we can have several matchers and test cases, as illustrated in Figure 2.3.

As shown in Figure 2.3, the evaluation test description should contain the list of test cases and matchers to be used. For each test case and matcher (iteration represented by the elements *ForEachTestCase* and *ForEachMatcher*), one main sequence of activities is carried out (as in the basic workflow). Within such a sequence, there are two sub sequences: the first one assigns the ontologies to be matched (*assignOntos* activity) to

local variables; invokes the respective matcher (*invokeAlign*); and assigns the results (*assignAlignResults*); while the second sequence stores the reference alignment in a local variable. Then, the resulting and reference alignments are used as input to invoke the evaluator process (partner link *evaluator*), that returns the evaluation results. These results are then assigned to a variable (*assignResults*), that will be used as input for the report generator service (partner link *ReportGenerator*), when the results of all matchers for all test cases have been generated. Finally, the report containing the interpretation of the alignment results is sent as reply to the process that has invoked the evaluation process.

2.4 Summary

This chapter presented the variability in the alignment task, discussing the parameters that must be controlled in its evaluation. Due to such high variability, the first SEALS campaign will focus on a simple kind of test demonstrating the feasibility of automating matching evaluation:

- comparing *two* ontologies written in the *same language*: OWL-DL,
- without input alignment,
- with any kind of fixed parameters and any kind of fixed and general purpose resources,
- without any kind of user input nor training samples.

Fortunately, this covers already several datasets offered in the current OAEI campaigns.

The aim of this chapter was also to show some possible evaluation workflows, specifying what happens in a matching evaluation experiment.

Next chapter presents the potential evaluation criteria and corresponding metrics that will be used in order to assess the matching algorithms and systems.

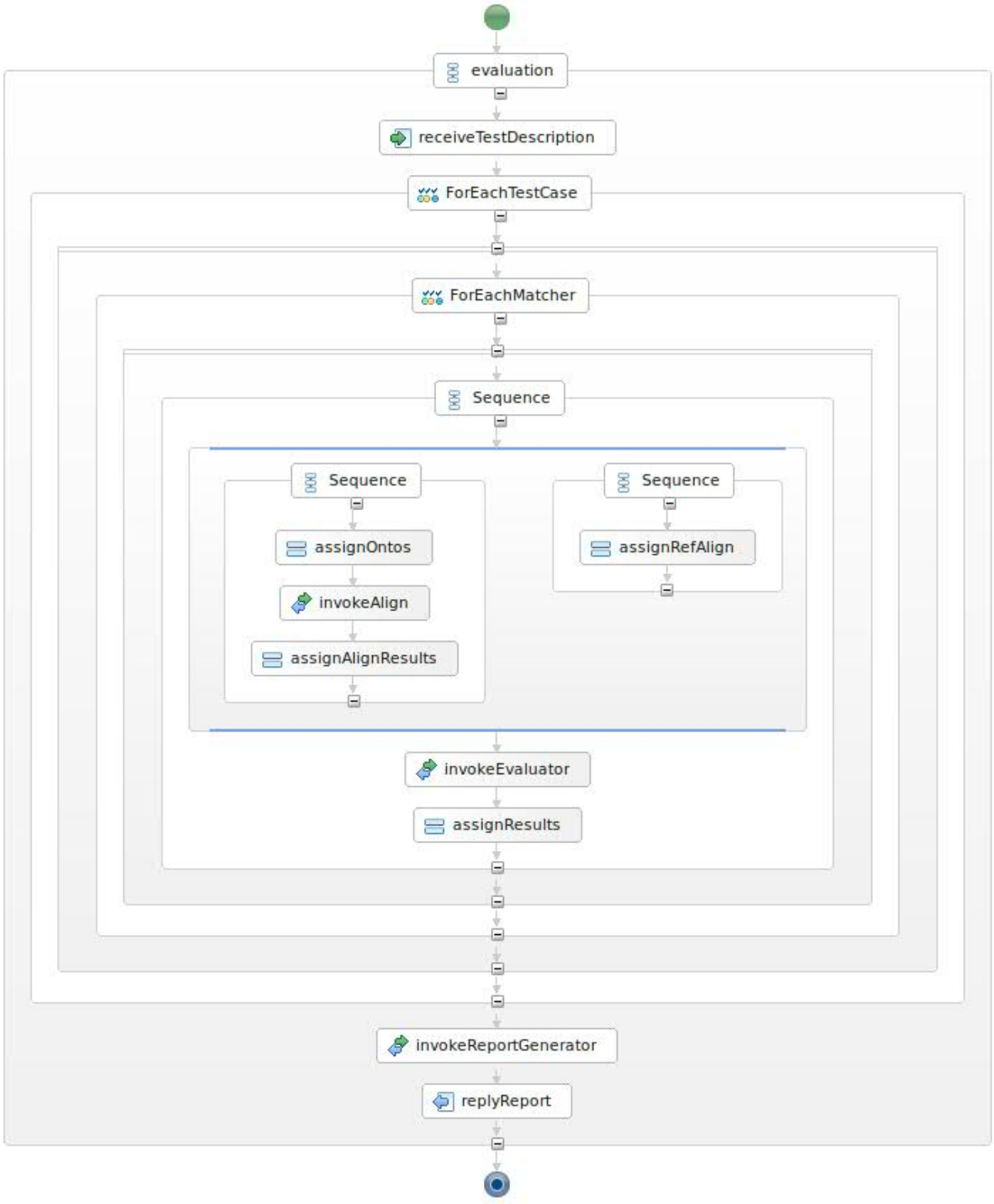


Figure 2.3: Advanced BPEL evaluation workflow.

3. Criteria and Measures

This chapter is concerned with the question of how to evaluate ontology matching algorithms and systems (evaluation targets). The aspects to be evaluated (criteria) and how to evaluate these aspects (metrics) are discussed.

Two groups of evaluations are considered. *Standard* evaluation (Section 3.1) concerns the first evaluation campaign and includes (a) compliance of matching systems with language standard (interoperability), (b) non functional but important features of the systems (such as efficiency, as ISO/IEC 9126-1 standard naming, and scalability), (c) the degree of conformance (or accuracy as as ISO/IEC 9126-1 naming) of the alignment methods to what is expected (precision and recall), and (d) alignment coherence.

For the second evaluation campaign, more elaborated criteria will be considered (Section 3.2). This includes (a) user-related measures focusing on user evaluation, (b) measures to evaluate specific tasks or applications, and (c) overall aggregating measures.

3.1 Standard Evaluation

3.1.1 Interoperability

Despite efforts on composing matchers [29] and [10] and on defining an Alignment API [12], ontology matching lacks interoperability benchmarks between tools. The first attempt to evaluate interoperability[1] between ontology matching systems is to measure their compliance to standards such as RDF(S) and OWL. In particular, we will test whether systems are able to correctly work on ontologies specified in the language standard RDF(S) and OWL. The criteria established in the deliverable D10.1 will be adapted to our case. Note that non-conformance to these standards sometimes can be detected through different criteria. For instance the inability to identify class names in a certain language will lead to a dramatic decrease of recall.

3.1.2 Efficiency and scalability

Efficiency measures the resource consumption for aligning two ontologies. Unlike the compliance measures, efficiency measures depend on the benchmark processing environment and the underlying ontology management system. Thus it is rather difficult to obtain objective evaluations. Metrics such as execution time (speed) and amount of required memory are usually considered to measure efficiency. For the first evaluation campaign we will use the Anatomy dataset (see Section 4.2), which is the largest dataset chosen for the first campaign, to measure these metrics.

Scalability criterion is also of prime importance. OAEI campaigns gave some preliminary evidence of the scalability characteristics of the ontology matching technology. In SEALS large tests involving 10.000, 100.000, and 1.000.000 entities per ontology

[1] `http://knowledgeweb.semanticweb.org/benchmarking_interoperability/owl/index.html`

(e.g., UMLS20 has about 200.000 entities) are to be designed and conducted to verify the behavior of matching systems.

Both, efficiency and scalability can depend on the nature of the ontology and more specifically on the complexity of the structures and definitions found in the ontology. Therefore there is a strong interaction between the hardness of tests wrt. efficiency and scalability and the complexity of the input ontologies.

Speed

Speed is measured by the amount of time taken by the algorithms for performing their alignment tasks. If user interaction is required, one has to ensure to effectively measure the processing time of the machine only.

We will measure time consumed by computing the time interval elapsed between the start and end of the matching process execution. It can be performed within the method implemented in the web service, which is invoked by *InvokeAlign* (Figures 2.2 and 2.3). Moreover, to have a more realistic measure, it should be averaged over several runs.

Memory

The amount of memory used for performing the alignment task marks another efficiency measure. Due to the dependency with underlying systems, it could also make sense to measure only the extra memory required in addition to that of the ontology management system (but it still remain highly dependent).

Measuring memory usage is similar to computing time, but in terms of required extra memory. It can be measured within the method invoked by the *InvokeAlign* activity in the BPEL process, with the help of specific system tools.

Scalability

There are two possibilities for measuring scalability, at least in terms of speed and memory requirements. First, it can be assessed by theoretical study. Second, it can be assessed by benchmark campaigns with quantified increasingly complex tests. From the results, the relationship between the complexity of the test and the required amount of resources can be represented graphically and the mathematical relationship can be approximated. In SEALS, scalability of matching tools will show how these tools are able to deal with complex problems in a given period of time.

3.1.3 Precision, recall, and others

There are many ways to qualitatively evaluate returned results [7]. One possibility consists of proposing a reference alignment (R) that is the one that the participants must find (a *gold standard*). The alignment A generated by the evaluated alignment algorithm can then be compared to that reference alignment.

Precision and recall

The most commonly used and understood measures are precision (true positive/retrieved) and recall (true positive/expected) which have been adopted for ontology alignment. They are commonplace measures in information retrieval.

Definition 2 (Precision) *Given a reference alignment R, the precision of some alignment A is given by*

$$P(A,R) = \frac{|R \cap A|}{|A|}.$$

Please note that precision can also be determined without explicitly having a complete reference alignment. Only the correct alignments among the retrieved alignments have to be determined ($R \cap A$), thus making this measure a valid possibility for ex-post evaluations.

Definition 3 (Recall) *Given a reference alignment R, the recall of some alignment A is given by*

$$R(A,R) = \frac{|R \cap A|}{|R|}.$$

The fallout measures the percentage of retrieved pairs which are false positive.

Definition 4 (Fallout) *Given a reference alignment R, the fallout of some alignment A is given by*

$$F(A,R) = \frac{|A| - |A \cap R|}{|A|} = \frac{|A \setminus R|}{|A|}.$$

F–measure

Precision and recall are the most widely and commonly used measures. But usually, when comparing systems one prefers to have only one measure. Unfortunately, systems are often not comparable based solely on precision and recall. The one which has higher recall has lower precision and vice versa. For this purpose, two measures are introduced which aggregate precision and recall.

The F-measure is used in order to aggregate the result of precision and recall.

Definition 5 (F-measure) *Given a reference alignment R and a number α between 0 and 1, the F-measure of some alignment A is given by*

$$M_\alpha(A,R) = \frac{P(A,R) \cdot R(A,R)}{(1-\alpha) \cdot P(A,R) + \alpha \cdot R(A,R)}.$$

If $\alpha = 1$, then the F-measure is equal to precision and if $\alpha = 0$, the F-measure is equal to recall. In between, the higher α, the more importance is given to precision with regard to recall. Very often, the value $\alpha = 0.5$ is used, i.e. $M_{0.5}(A,R) = \frac{2 \times P(A,R) \times R(A,R)}{P(A,R)+R(A,R)}$, the harmonic mean of precision and recall.

The overall measure (defined in [34] as accuracy) is an attempt of measuring the effort required to fix the given alignment (the ratio of the number of errors on the size of the expected alignment). Overall, it is always lower than the F-measure.

Definition 6 (Overall) *Given a reference alignment R, the overall of some alignment A is given by*

$$O(A,R) = R(A,R) \times \left(2 - \frac{1}{P(A,R)}\right).$$

It can also be defined as:

$$O(A,R) = \frac{|(A \cup R) - (A \cap R)|}{|R|}.$$

When comparing systems in which precision and recall can be continuously determined, it is more convenient to draw the precision/recall curve and compare these curves. This kind of measure is widespread in the results of TREC competitions.

Weighted Hamming distance

The Hamming distance measures the similarity between two alignments by counting the joint correspondences with regard to the correspondence of both sets.

Definition 7 (Hamming distance) *Given a reference alignment R, the Hamming distance between R and some alignment A is given by*

$$H(A,R) = 1 - \frac{|A \cap R|}{|A \cup R|}.$$

The Weighted Hamming distance pays attention not only to the correspondences but to their strengths as well. It requires that the strengths be the same in both sets of correspondences.

Definition 8 (Weighted Hamming distance) *Given a reference alignment R, the weighted Hamming distance between R and some alignment A is given by*

$$W(A,R) = \sum_{c \in A \cup R} \frac{|strength_A(c) - strength_R(c)|}{|A \cup R|}$$

in which $strength_X(c)$ is 0 if $c \notin X$.

However, since the semantics of strength is not well defined, it is hazardous to use them for comparing alignments. Moreover, it can be considered that some reference alignment is always achievable in each context. In such a case, it would be useful to compare an exact (hardened) version of each obtained alignment rather than a rough alignment unless the way it is used is known.

It can be more interesting to measure from how far the alignment missed the target. To that extent it is necessary to measure a distance from an obtained alignment and a reference alignment, as discussed in the next section.

3.1.4 Generalizations of precision and recall

It can happen that an alignment is very close to the expected result and another quite remote from it, however, both share the same precision and recall. The reason for this is that such metrics only compare two sets of correspondences without considering if these are close or remote to each other: if they are not the same exact correspondences, they score zero. They both score identically low, despite their different quality. It may be helpful for users to know whether the found alignments are close to the expected one and easily repairable or not. It is thus necessary to measure the proximity between alignments instead of their strict equality ([9], [13]). This section discusses two proposals to overcome this problem.

Relaxed precision and recall

[9] proposes to generalize precision and recall, measuring the proximity of correspondence sets rather than their strict overlap. Instead of taking the cardinal of the intersection of the two sets |R ∩ A|, they propose to measure their proximity (ω).

Definition 9 (Generalized Precision and Recall) *Given a reference alignment R and an overlap function ω between alignments, the precision and recall of an alignment A are given by*

$$P_\omega(A,R) = \frac{\omega(R \cap A)}{|A|}$$

$$R_\omega(A,R) = \frac{\omega(R \cap A)}{|R|}$$

There are different ways to design such a proximity given two sets. In [9] the authors propose to find correspondences matching each other and computing the sum of their proximity. This can be defined as an overlap proximity:

Definition 10 (Overlap Proximity) . *A measure that would generalize precision and recall, where M(A,R) is a matching between alignments*

$$\omega(A,R) = \sum_{<a,r> \in M(A,R)} \sigma(a,r)$$

To compute ω(A,R), it is necessary to measure the proximity between two matched correspondences (i.e., <a,r> ∈ M(A,R)) on the basis of how close the result is from the ideal one. Each element in the tuple a = <e_a,e'_a,r_a,n_a> will be compared with its counterpart in r = <e_r,e'_r,r_r,n_r>. For any two correspondences (the found a and the reference r), three similarities are computed: σ_{pair}, σ_{rel}, σ_{conf}:

- σ_{pair} How is one entity pair similar to another entity pair? In ontologies, it can follow any relation which exists (e.g., subsumption, instantiation), or which can be derived in a meaningful way. The most important parameters are the relations to follow and their effect on the proximity;

- σ_{rel} Often the alignment relations are more complex, e.g. subsumption, instantiation, or compositions. Again, one has to assess the similarity between these relations. The two relations of the alignment cell can be compared based on their distance in a conceptual neighborhood structure [21],[17];
- σ_{conf} Finally, one has to decide, what to do with different levels of confidence. The similarity could simply be the difference. Unfortunately, none of the current alignment approaches have an explicit meaning attached to confidence values, which makes it rather difficult in defining an adequate proximity.

Based on these three similarities, the correspondence proximity can be defined:

Definition 11 (Correspondence Proximity) *Given two correspondences* $<e_a,e_a^{'},r_a,n_a>$ *and* $<e_r,e_r^{'},r_r,n_r>$*, their proximity is:*

$$\sigma(< e_a, e_a^{'}, r_a, n_a >, < e_r, e_r^{'}, r_r, n_r >) = \sigma(< e_a, e_r >, < e_a^{'}, e_r^{'} >) \times \sigma(r_a, r_r) \times \sigma(n_a, n_r)$$

Three concrete measures based on the above definitions are proposed in [9]: *symmetric proximity*, *correction effort*, and *oriented proximity*. This section the first one, which have been used in OAEI evaluation campaigns, is presented in more detail. The *symmetric measure* is based on computing a distance δ on the ontological entities and to weight the proximity with the help of this distance: the higher the distance between two entities in the matched correspondences the lower their proximity. The distance is then characterized by σ_{pair}, σ_{rel}, σ_{conf}:

- $\sigma_{pair}(<e_a,e_a^{'}>,<e_r,e_r^{'}>)$ The distance is given by having a similarity inversely proportional to the distance in the sub-sumption tree (a class is at distance 0 of itself, at distance 0.5 of its direct sub- and superclasses, and at a distance 1 of any other class);
- $\sigma_{rel}(r_a,r_r)$ For the proximity between relations, it is only presented the similarity between equality (=) and other relations. It takes a similarity of 1.0 if correct relations (expected) are found in both r_a and r_r (i.e., $r_a = r_r$); and 0.5 if found relations are the same in both r_a and r_r (equivalence), but the correct relation involves a subsumption between the entities being mapped;
- $\sigma_{conf}(n_a,n_r)$ It is taken the complement of the difference between the two confidences.

Using the *correction effort* measure, the quality of alignments can be measured through the effort required for transforming the found alignment into the correct one. This measure can be implemented as an edit distance [30], which defines a number of operations by which an object can be corrected and assigns a cost to each of these operations (the effort required to identify and repair some mistake). The cost of a sequence of operations is the sum of their cost and the distance between two objects is the cost of the less costly sequence of operations that transform one object into the other one. Such a distance is then turned into a proximity measures. Finally, *oriented-effort* measure considers two different similarities depending of their application for evaluating either precision or recall. It associates different weights to compute the proximity measure in each case.

Semantic precision and recall

The measures above are based on syntactic generalizations of precision and recall. In order to design a generalization of precision and recall that is semantically grounded, [13] proposes *semantic precision* and *recall*. In such measures, those correspondences that are consequences of the evaluated alignments have to be considered as recalled and those that are consequence of the reference alignments as correct.

The semantic extension of precision and recall consists of using the set of α-consequences (or deductive closure on the prover side) instead of |A ∩ R|:

Definition 12 (α-consequence of aligned ontologies) *Give two ontologies o and o' and an alignment A between these ontologies, a correspondence σ is a α-consequence of o, o' and A (note $A \models \sigma$) if and only if for all models <m,m',γ> of o, o' and A, m, m' $\models \gamma$ (the set of α-consequence is noted by Cn(A)).*

In order to deal with the problems raised by the infinite character of the set of α-consequences, it is proposed to use a deductive closure bounded by a finite set so that the result is finite. It is based on different sets of true positives as:

$$TP_P(A,R) = \{\delta \in A; R \models \delta\} = A \cap Cn(R)$$
$$TP_R(A,R) = \{\delta \in R; A \models \delta\} = Cn(A) \cap R$$

The semantic precision and recall are based on these sets:

Definition 13 (Semantic Precision and Recall) *Given a reference alignment R, the precision of some alignment A is given by:*

$$P_{sem}(A,R) = \frac{|A \cap Cn(R)|}{|A|}$$
$$R_{sem}(A,R) = \frac{|Cn(A) \cap R|}{|R|}$$

Another possible way to implement semantic precision and recall is to distinguish between complex and non-complex correspondences. An alignment is said to be non-complex if it contains only non-complex correspondences. A non complex correspondence is a correspondence, which relates two named terminological entities (concepts or properties) via equivalence or subsumption.

The majority of current ontology matchers produces non-complex alignments and thus this restriction is only of little impact for current alignment evaluations. Therefore it is possible to directly compare the closures of both A and R to compute semantic precision and recall, since non of these sets becomes infinite as long as we restrict the closures of A and R to non-complex alignments. This approach is referred to as *restricted semantic precision and recall* and has been described and tested in ([40], [20]).

3.1.5 Alignment coherence

The term alignment (in)coherence has first been introduced in a paper concerned with the task of reasoning about ontology alignments in general [41].[2] Measuring

[2]More precisely, the authors referred to the corresponding notion as 'mapping inconsistency'.

the degree of (in)coherence of an alignment has been proposed in [33] for the first time. The authors argue that the incoherence of an alignment results in different kinds of problems depending on the specific application context. Thus, coherence of an alignment is an important quality, which has to be taken into account in the evaluation context.

The approach for measuring the degree of (in)coherence measuring is based on the notion of an aligned or merged ontology. Given two ontologies O_1 and O_2 and an alignment A between them, the merged ontology $O_1 \cup_A O_2$ is the union of O_1, O_2, and A where A is interpreted as a set of axioms. A correspondence expressing equivalence between two concepts, for example, is thus translated into an equivalence axiom in the context of the merged ontology. In [33] this approach is referred to as natural translation.

An alignment A between two ontologies O_1 and O_2 is called incoherent, if there exists an unsatisfiable concept $i\#C_{i\in\{1,2\}}$ in $O_1 \cup_A O_2$; its unsatisfiability must have (at least partially) been caused by A.

Definition 14 (Incoherence of an Alignment) *Given an alignment A between ontologies O_1 and O_2. If there exists a concept $i\#C$ with $i \in \{1,2\}$ such that $O_1 \cup_A O_2 \models \bot \sqsupseteq i\#C$ and $O_i \not\models \bot \sqsupseteq i\#C$ then A is incoherent with respect to O_1 and O_2. Otherwise A is coherent with respect to O_1 and O_2.*

It is possible to define alternative semantics for an alignment, which differ from the natural interpretation as axioms. However, the four measures proposed in [33] are independent of this choice. In the following we pick up two of these measures, namely the *Unsatisfiability Measure* and the *Maximum Cardinality Measure*.

The first measure is based on the idea of counting unsatisfiable concepts. It is derived from an ontology incoherence measure introduced in [36]. Contrary to measuring incoherences in ontologies, it has to be distinguished between two types of concept unsatisfiability in the merged ontology: There are unsatisfiable concepts in $O_1 \cup_A O_2$ which have already been unsatisfiable in O_1, respectively O_2, while there are unsatisfiable concepts which have been satisfiable in O_1, respectively O_2. These concepts have become unsatisfiable due to the impact of A. In particular, we compare the number of these concepts with the number of all named concepts satisfiable in O_1 or O_2.

Definition 15 (Unsatisfiability Measure) *Let A be an alignment between ontologies O_1 and O_2. Unsatisfiability measure m_{sat} is defined by*

$$m_{sat}(O_1, O_2, A) = \frac{|US(O_1 \cup_t AO_2) \setminus (US(O_1) \cup US(O_2))|}{|CO(O_1 \cup_t AO_2) \setminus (US(O_1) \cup US(O_2))|}$$

where $CO(O)$ refers to the set of named concepts in an ontology O and $US(O) = \{C \in CO(O) \mid O \models C \sqsubseteq \bot\}$ refers to the set of unsatisfiable concepts in O.

The *Maximum Cardinality Measure* is concerned with the effort of revising an incoherent alignment. We use the term revision to describe the process of removing correspondences from an incoherent alignment until a coherent subset of the alignment has been found. In particular, the *Maximum Cardinality Measure* is based on the idea to remove a minimum number of correspondences to achieve the coherence of the alignment.

Definition 16 (Maximum Cardinality Measure) *Let A be an alignment between ontologies O_1 and O_2. Maximum cardinality measure m_{card} is defined by*

$$m_{card}(O_1, O_2, A) = \frac{|A \setminus A'|}{|A|}$$

where $A' \subseteq A$ is coherent with respect to O_1 and O_2 and there exists no $A'' \subseteq A$ with $|A''| > |A'|$ such that A'' is coherent with respect to O_1 and O_2.

As shown in [33], this measure can be used to compute a strict upper bound for the precision of an alignment. In particular, we have $precision(A, R) \leq 1 - m_{card}(O_1, O_2, A)$. Thus, we are able to compute an upper bound for the precision of an alignment in the absence of a reference alignment R. This will be useful in many evaluation contexts where a reference alignment is missing or only partially available.

The coherency of an alignment is also a quality of its own. Thus, we would expect an automatically generated alignment to be coherent. However, first evaluations concerned with these measures revealed that the opposite is the case. The *Maximum Cardinality Measure* has been applied to the submissions of the OAEI conference track (see Section 4.3 for a description of the data set) and it turned out that none of the matching systems participating could ensure the coherence of the generated alignments. Even for systems as ASMOV (see Section 5.1) and Lily (see Section 5.4), systems with semantic verification component, a high degree of incoherence has been measured.

Measuring the degree of incoherence obviously requires full-fledged reasoning techniques. It is thus heavily linked to issues concerned with reasoning systems and is in particular a very interesting, but specific usecase for incoherence debugging.

3.2 Advanced Evaluation

The evaluation criteria described above involve standard ways to evaluate ontology matching systems (despite, for instance, some extended measures such as generalizations of precision and recall). However, the quality of a matcher can be assessed regarding its suitability for a specific task or application, as well as the user can be involved into the evaluation loop. In the following, such criteria are discussed.

3.2.1 Task-specific evaluation

Evaluation should help users to choose the best algorithm for their task. In terms of measurements, it would be useful to set up experiments which do not stop at the delivery of alignments but carry on with the particular task. This is especially true when there is a clear measure of the success of the overall task. Even without this, it can be useful to share corresponding aggregate measures associated to one *task profile*.

Different *task profiles* can be established to explicitly compare matching systems for certain tasks. The following list of possible applications gives hints on such scenarios [19]:

Ontology evolution uses matching for finding the changes that have occurred between two ontology versions;

Schema integration uses matching for integrating the schemas of different databases under a single view;

Catalog integration uses matching for offering an integrated access to online catalogs;

Data integration uses matching for integrating the content of different databases under a single database;

P2P information sharing uses matching for finding the relations of ontologies used by different peers;

Web service composition uses matching between ontologies describing service interfaces in order to compose web services by connecting their interfaces;

Multi agent communication use matching for finding the relations between the ontologies used by two agents and translating the messages they exchange;

Context matching in ambient computing uses matching of application needs and context information when application and devices have been developed independently and use different ontologies;

Query answering uses ontology matching for translating user queries about the web;

Semantic web browsing uses matching for dynamically (while browsing) annotating web pages with partially overlapping ontologies.

Based on the analysis of such tasks, the requirements of applications can be established with regard to matching systems (summarized in Table 3.1):

- input (for instance, applications require only a matching solution able to work without instances),
- some specific behavior of matching, such as requirements of (*i*) being *automatic*, i.e., not relying on user feed-back, (*ii*) being *correct*, i.e., not delivering incorrect matches, (*iii*) being *complete*, i.e., delivering all the matches, and (*iv*) having a good *run-time* efficiency.
- the use of the matching result. In particular, how the identified alignment is going to be processed, e.g., by merging the data or conceptual models under consideration or by translating data instances among them.

Regarding matcher profiles, the following data could be used to characterize the systems:

- input characteristics: size; use of external resources;
- approach: individual algorithms; hybrid and composite solutions; automatic, semi-automatic or manual execution; maximal time of execution, disc space, precision; recall;

Application	instances	run time	automatic	correct	complete	operation
Ontology evolution	*			*	*	transformation
Schema integration	*			*	*	merging
Catalog integration	*			*	*	data translation
Data integration	*			*	*	query mediation
P2P information sharing		*				query mediation
Web service composition		*	*	*		data mediation
Multi agent communication		*	*	*	*	data translation
Context matching in ambient computing		*	*	*		data translation
Query answering	*	*				query reformulation
Semantic web browsing	*	*		*		navigation

Table 3.1: Summary of applications requirements (from [19]).

- output features: complete or partial matching (match for all elements or not), cardinality, type of correspondence;
- usage features: local use, network use; internet use; application area (integration; transformation; query answering, etc.); human or machine applicable;
- documentation available or not;
- cost features: license.

The data source for matching profiles can be from literature analysis for finding the systems properties; exploitation of questionnaires as well as by intensive collaborations with developers of matching approaches; and results of evaluations, as performed in OAEI.

When the application requirements are known and the matcher profiles have been obtained, it is necessary to match them in order to decide which matcher to use. One naive method is based on weighted aggregation of the characteristics depending on the expressed needs of applications.

[8] provided an analysis of the different needs for evaluation depending on specific applications. His technique is applied to the requirement table (Table 3.1), as proposed by [19]. As a matter of fact, it can be rewritten in function of the measurements obtainable by evaluating the matchers. This technique is used to design Table 3.2. Therefore, different *application profiles* could be established to explicitly compare matching algorithms with respect to certain tasks.

Such table can be useful for aggregating the measures corresponding to each of these aspects with different weights or to have an ordered way to interpret evaluation results. For aggregating measures depending on a particular application, its is possible to use weights corresponding to the values of Table 3.2, and thus respecting the importance of each factor. Weighted aggregation measures (weighted sum, product, or average) can be used.

F-measure is already an aggregation of precision and recall. It can be generalized as a harmonic mean, for any number of measures. This requires to assign every

Application	speed	automatic	precision	recall
Ontology evolution	medium	low	high	high
Schema integration	low	low	high	high
Catalog integration	low	low	high	high
Data integration	low	low	high	high
P2P information sharing	high	low	medium	medium
Web service composition	high	high	high	low
Multi agent communication	high	high	high	medium
Context matching in ambient computing	high	high	high	medium
Query answering	high	medium	medium	high
Semantic web browsing	high	medium	high	low

Table 3.2: Application requirements of Table 3.1 reinterpreted as measurement weights (from [19]).

measurement a weight, such that these weights sum to 1. Obviously the weights have to be chosen carefully, again depending on the goal.

Definition 17 (Weighted harmonic mean) *Given a reference alignment R, a set of measures $(M_i)_{i\in I}$ provided with a set of weights $(w_i)_{i\in I}$ between 0 and 1 such that their sum is 1, the weighted harmonic mean of some alignment A is given by*

$$H(A,R) = \frac{\prod_{i\in I} M_i(A,R)}{\sum_{i\in I} w_i \cdot M_i(A,R)}.$$

3.2.2 User related evaluation

So far the measures have been machine focused. In some cases algorithms or applications require some kind of user interaction. This can range from the user utilizing the alignment results to concrete user input during the alignment process. In this case, it is even more difficult to obtain some objective evaluation. This subsection proposes measures to get the user into the evaluation loop.

Level of user input effort

In case algorithms require user intervention, this intervention could be measured in terms of some elementary information the users provide to the system. When comparing systems which require different input or no input from the user, it will be necessary to consider a standard for elementary information to be measured. This is not an easy task.

A first step towards evaluating the impact of user effort has been proposed in the OAEI anatomy track in 2008 (see Section 4 in [5]). Participating systems could not only use the information encoded in the ontologies, but could also take into account

a provided partial reference alignment as additional parameter. The additional information encoded in the partial reference alignment can be seen as a simulation of user input. Based on this approach it is possible to measure in how far this information can be exploited.

General subjective satisfaction

From a use case point of view it makes sense to directly measure the user satisfaction. As this is a subjective measure it cannot be assessed easily. Extensive preparations have to be made to ensure a valid evaluation. Almost all of the objective measures mentioned so far have a subjective counterpart. Possible measurements would be:

- input effort,
- speed,
- resource consumption (memory),
- output exactness (related to precision),
- output completeness (related to recall),
- and understandability of results (oracle or explanations).

Due to its subjective nature numerical ranges as evaluation result are less appropriate than qualitative values such as very good, good, satisfactory, etc.

3.2.3 Aggregating evaluation measures

Different measures suit different evaluation goals. If we want to improve our system, it is best to have as many indicators as possible. But if we want to single out the best system, it is generally easier to evaluate with very few or only one indicator. For the first case, different individual measurements have to be aggregated. This can be achieved by giving every measurement a weight (e.g., in form of a weighted linear aggregation function). Obviously the weights have to be chosen carefully, again dependent on the goal.

Definition 18 (Aggregated measure) *Given a set of evaluation measures $m_i \in M$ and their weighting $w_i \in W$, the aggregated measure Aggr is given by*

$$Aggr(M, W) = \sum_{m_i \in M} w_i \cdot m_i.$$

3.3 Evaluation and Results Metadata

In the SEALS platform matching algorithms and systems will be evaluated using an evaluation description, producing a result description upon which an interpretation can be made. This section describes the metadata to be used to describe evaluation and results in the respective repositories. Such metadata is based on the specification of the corresponding repository content provided in the deliverable D7.1.

The evaluation metadata will contain the following data:

- evaluation ID - an ID uniquely identifying the evaluation description, which should be a referable URI,
- checksum,
- evaluation name - for use in result presentation,
- description - a short informal description of the workflow,
- creator - a reference to the SEALS user, who created the workflow,
- classification - e.g. run, evaluation, campaign,
- relations to other entities:
  - subworkflows - a list of reference to all subworkflows,
  - used-in workflows - a list of reference to all superworkflows,
  - previous version - a reference to the previous version of the workflow,
  - set of test data - *can in principle be derived from the workflow itself*,
  - evaluation criteria/metric,
  - target (algorithms/systems),
- access rights.

At the moment it is not clear in how far some of the metadata can be derived from the worflow itself (see remarks above). Further considerations have to clarify this issue. Moreover, an evaluation should contain a workflow definition (XML File) which is the data described by the metadata.

The results of an evaluation experiment will be described by the following data:

- short name,
- result ID - an ID uniquely identifying the evaluation result,
- evaluation ID,
- timestamp - a stamp of the datetime when the result was generated,

The results have associated an interpretation:

- short name,
- description (criteria and metric),
- result data (alignment),
- creator contact,
- timestamp.

3.4 Summary

This chapter presented several criteria to evaluate matching systems, which are summarized in Table 3.3. Currently the most natural factors to measure system's quality are precision and recall, specially because they can be interpreted easily. However, it is one of the goals of SEALS to provide alternative criteria for evaluation, involving semantic measures and task-specific evaluations.

In the first evaluation campaign, matching systems will be evaluated with respect to the following criteria for which we will provide evaluation components implementing the corresponding measures

- Interoperability,

Evaluation	Criteria	Metric
Standard	Interoperability	compliance with RDF/OWL
	Efficiency	execution time and required memory
	Scalability	different test sizes (complex tests)
	Compliance with reference alignment	precision, recall, f–measure, and generalizations
	Coherence	minimal revision effort to achieve coherence
Advanced	User satisfaction	subjective satisfaction (qualitative values – very good, good, satisfactory, etc.)
	Task-oriented	based on matching system and task profiles (aggregated measures)

Table 3.3: Criteria and metrics for ontology matching evaluation.

- Compliance with reference alignment: standard precision and recall, restricted semantic precision and recall.
- Coherence.

Further, facilities for measuring the efficiency of matching systems in terms of speed and memory usage will be provided by the SEALS platform and included in the evaluation reports.

Measuring the scalability of matching approaches requires a more complex evaluation workflow in which the hardness of the evaluation problem is increased stepwise and the efficiency of the system is measured in each step. As these complex workflows will only be included in the second evaluation campaign, scalability is not an issue for the first campaign.

Task-specific evaluation have to be investigated in more detail before meaningful evaluation workflows can be defined. Therefore, we also do not consider them for the first evaluation campaign.

4. Test Data for Evaluation

Since 2004, a group of researchers on ontology matching, of which we belong to, have run several evaluation campaigns which are identified as Ontology Alignment Evaluation Initiative[1] (OAEI). The main goal of the OAEI is to compare systems and algorithms on the same basis and to allow anyone for drawing conclusions about the best matching strategies. From such evaluations, tool developers can learn and improve their systems. The OAEI campaigns provide the evaluation of matching systems on consensus test cases.

Two first evaluation events were organized in 2004, (*i*) the Information Interpretation and Integration Conference (I3CON) held at the NIST Performance Metrics for Intelligent Systems (PerMIS) workshop and (*ii*) the Ontology Alignment Contest held at the Evaluation of Ontology-based Tools (EON) workshop of the annual International Semantic Web Conference (ISWC) [42].

The first OAEI campaign occurred in 2005 and the results were presented at the workshop on Integrating Ontologies held in conjunction with the International Conference on Knowledge Capture (K-Cap) [2], in 2006 at the first Ontology Matching (OM) workshop collocated with ISWC [18], in 2007 at the second OM workshop collocated with ISWC+ASWC [16], and in 2008, OAEI results were presented at the third OM workshop collocated with ISWC, in Karlsruhe, Germany[2]. Finally, the OAEI 2009 results are presented at the fourth OM collocated with ISWC, in Virginia, USA.

Each campaign has a large variety of test cases that emphasize different aspects of ontology matching. The following test cases are proposed in the OAEI 2009:

Comparison track: benchmark The goal of this systematic benchmark series is to identify the areas in which each matching algorithm is strong and weak. The test is based on one particular ontology dedicated to the very narrow domain of bibliography and a number of alternative ontologies of the same domain for which alignments are provided.

Expressive ontologies track: offers ontologies using OWL modeling capabilities:

Anatomy: The anatomy real world case is about matching the Adult Mouse Anatomy (2744 classes) and the NCI Thesaurus (3304 classes) describing the human anatomy.

Conference track and consensus workshop: Participants were asked to freely explore a collection of conference organization ontologies (the domain being well understandable for every researcher). Organizers of this track offer diverse a priori and a posteriori evaluation of results.

Directories and thesauri track: proposed web directories, thesauri and generally less expressive resources:

[1] `http://oaei.ontologymatching.org`
[2] `http://om2008.ontologymatching.org`

test	formalism	relations	confidence	modalities	language
benchmark	OWL	=	[0 1]	open	EN
anatomy	OWL	=	[0 1]	blind	EN
conference	OWL-DL	=, ≤	[0 1]	blind+open	EN
directory	OWL	=	1	open	EN
library	SKOS, OWL	narrow-, exact-, broad	1	blind	EN+DU
oriented	OWL	=, ≤,	[0,1]	open	EN
eprints	RDF	=	[0,1]	open	EN
tap	RDF	=	[0,1]	open	EN
iimb	RDF	=	[0,1]	open	EN
vlcr	SKOS, OWL	exactMatch, closeMatch	[0,1]	blind+expert	EN+DU

Table 4.1: Characteristics of test cases.

Directory: The directory real world case consists of matching web sites directories (like open directory or Yahoo's). It is more than 4 thousand elementary tests.

Library: Two SKOS thesauri about books have to be matched using relations from the SKOS Mapping vocabulary. Samples of the results are evaluated by domain experts. In addition, application dependent evaluations are run.

Oriented matching track: This track focuses on the evaluation of alignments that contain other mapping relations than equivalences.

Instance matching track: The instance data matching track aims at evaluating tools able to identify similar instances among different datasets. It features Web datasets, as well as a generated benchmark.

Very large crosslingual resources: This real world test case requires matching very large resources (vlcr) available on the web, viz. DBPedia, WordNet and the Dutch audiovisual archive (GTAA), DBPedia is multilingual and GTAA is in Dutch.

Eprints-Rexa-Sweto/DBLP benchmark: Three datasets containing instances from the domain of scientific publications

TAP-Sweto-Tesped-DBpedia: Three datasets covering several topics and structured according to different ontologies

IIMB: A generated benchmark constituted using one dataset and modifying it according to various criteria.

Table 4.1 summarizes the variation in the results expected from these tests. Regarding the kind of evaluation (modalities), open evaluation is made with already published reference alignments; blind evaluation is made by organizers from reference alignments unknown to the participants; and consensual evaluation is obtained by reaching consensus over the found results.

In SEALS, specially for the first evaluation campaign, three test cases of OAEI will be considered as test data for evaluation: benchmark, conference, and anatomy. These sets are described in detail in the following.

4.1 Benchmark

The goal of the benchmark tests is to provide a stable and detailed picture of each algorithm. For that purpose, the algorithms run on systematically generated test cases.

The domain of this first test is Bibliographic references. It is based on a subjective view of what must be a bibliographic ontology. There can be many different classifications of publications, for example, based on area and quality. The one chosen here is common among scholars and is based on publication categories; as many ontologies (tests #301-304), it is reminiscent to BibTeX.

The systematic benchmark test set is built around one reference ontology and many variations of it. The ontologies are described in OWL-DL and serialized in the RDF/XML format. The reference ontology is that of test #101. It contains 33 named classes, 24 object properties, 40 data properties, 56 named individuals and 20 anonymous individuals. Participants have to match this reference ontology with the variations. Variations are focused on the characterization of the behavior of the tools rather than having them compete on real-life problems. They are organized in three groups:

Simple tests (1xx) such as comparing the reference ontology with itself, with another irrelevant ontology (the wine ontology used in the OWL primer) or the same ontology in its restriction to OWL-Lite;

Systematic tests (2xx) obtained by discarding features from some reference ontology. It aims at evaluating how an algorithm behaves when a particular type of information is lacking. The considered features were:

- *Name of entities* that can be replaced by random strings, synonyms, name with different conventions, strings in another language than English;
- *Comments* that can be suppressed or translated in another language;
- *Specialization hierarchy* that can be suppressed, expanded or flattened;
- *Instances* that can be suppressed;
- *Properties* that can be suppressed or having the restrictions on classes discarded;
- *Classes* that can be expanded, i.e., replaced by several classes or flattened.

Four real-life ontologies of bibliographic references (3xx) found on the web and left mostly untouched (there were added xmlns and xml:base attributes).

Since the goal of these tests is to offer some kind of permanent benchmarks to be used by many, the test is an extension of the 2004 EON Ontology Alignment Contest, whose test numbering it (almost) fully preserves. Participants are expected to deliver equivalence correspondences between named classes and properties.

4.2 Anatomy

The focus of the anatomy testdata is to confront existing matching technology with real world ontologies. Currently, we find such real world cases primarily in the biomedical

domain, where a significant number of ontologies have been built covering different aspects of medical research. Manually generating alignments between these ontologies requires an enormous effort by highly specialized domain experts. Supporting these experts by automatically providing alignment proposals is both challenging, due to the complexity and the specialized vocabulary of the domain, and relevant, due to the increasing number of ontologies used in clinical research.

The ontologies of the anatomy track are the NCI Thesaurus describing the human anatomy, published by the National Cancer Institute (NCI)[3], and the Adult Mouse Anatomical Dictionary[4], which has been developed as part of the Mouse Gene Expression Database project. Both resources are part of the Open Biomedical Ontologies (OBO). Both ontologies are more or less typical examples of large, carefully designed ontologies that are described in technical terms. The ontology describing the human anatomy contains for example a concept labeled *Abdominal esophagus*, which refers to the lower part of the gullet. The human ontology contains 3304 concepts and the mouse anatomy 2744 concepts. Besides their large size and a conceptualization that is only to a limited degree based on the use of natural language, they also differ from other ontologies with respect to the use of specific annotations and roles. For example, the extensive use of the *partOf* relation is an essential characteristic of anatomical ontologies.

The complex and laborious task of generating the reference alignment has been conducted by a combination of computational methods and an extensive manual evaluation with the help of domain experts. In addition to generating a reference alignment, the ontologies were extended and harmonized to increase the number of correspondences between both ontologies. A more elaborate description of creating the reference alignment can be found in [3]. The manual harmonization of the ontologies leads to a situation, where we have a high number of rather trivial correspondences that can be found by simple string comparison techniques. At the same time, we have a good share of non-trivial correspondences that require a careful analysis and sometimes also medical background knowledge.

The anatomy test data has been used in OAEI 2007, 2008 and 2009 within the anatomy track. Due to the importance of the biomedical domain it has attracted a constant number of 9-11 participating systems. Besides applying the classical measures of precision and recall, the evaluation process additionally focused on runtime aspects. However, due to the absence of an evaluation platform the runtime comparison was based on the information delivered by the participants. Only few systems have been manually installed and were run to verify the information delivered by the participants. The SEALS platform will allow to compare results related to runtime and memory consumption that are reliable and reproducible.

In summary, we can conclude that the anatomy data set is well suited to measure the characteristics of different matching systems with respect to the problem of matching biomedical ontologies. Due to its relatively large size, it also provides an interesting dataset for comparing runtime and memory consumption.

[3] `http://www.cancer.gov/cancerinfo/terminologyresources/`
[4] `http://www.informatics.jax.org/searches/AMA\_form.shtml`

Ontology	Type	Concepts	Datatype Prop.	Object Prop.	Expressivity	Ref
Ekaw	Insider	77	-	33	$\mathcal{SHIN}$	Yes
Sofsem	Insider	60	18	46	$\mathcal{ALCHIF}$(D)	Yes
Sigkdd	Web	49	11	17	$\mathcal{ALEI}$(D)	Yes
Iasted	Web	140	3	38	$\mathcal{ALCIN}$(D)	Yes
Micro	Web	32	9	17	$\mathcal{ALCOIN}$(D)	-
Confious	Tool	57	5	52	$\mathcal{SHIN}$(D)	-
Pcs	Tool	23	14	24	$\mathcal{ALCIF}$(D)	-
OpenConf	Tool	62	21	24	$\mathcal{ALCOI}$(D)	-
ConfTool	Tool	38	23	13	$\mathcal{SIN}$(D)	Yes
Crs	Tool	14	2	15	$\mathcal{ALCIF}$(D)	-
Cmt	Tool	36	10	49	$\mathcal{ALCIN}$(D)	Yes
Cocus	Tool	55	-	35	$\mathcal{ALCIF}$	-
Paperdyne	Tool	47	21	61	$\mathcal{ALCHIN}$(D)	-
Edas	Tool	104	20	30	$\mathcal{ALCOIN}$(D)	Yes
MyReview	Tool	39	17	49	$\mathcal{ALCOIN}$(D)	-

Table 4.2: Ontologies of the conference test set.

4.3 Conference

The conference test dataset consists of a collection of ontologies that describe the same domain, namely the domain of conference organization. This dataset has been developed by a group of researchers from the University of Economics, Prague. Its origin is described in [43]. Since 2005 is has continuously been refined, extended and used as test data of the OAEI conference/consensus track. The characteristics of the dataset are described in Table 4.2.

The conference ontologies feature three characteristics, which make them interesting as ontology matching dataset:

1. They share the same, generally understandable domain of conference organization.

2. They have been built by different groups, reflecting different naming conventions and conceptualizations.

3. They are described by the use of various types of axioms.

The first point makes it possible to generate a high quality reference without highly specialized domain experts. This has been done by extending step by step partial reference alignments, which have first been created in the context of the work reported in [32]. In addition, the correctness of many correspondences has been discussed within the consensus workshop, which has been part of the Ontology Matching workshop from 2006 to 2008. At the moment reference alignments are available for all pairs of ontologies from a subset of seven ontologies. The last column in Table 4.2 indicates for which ontologies reference alignments have been created.

Due to the second point, the dataset results for many combinations in hard matching problems. This is also based on the fact that the ontologies can be divided into three types with respect to their origin (second column in Table 4.2): Ontologies based on conferences and their web pages (Web), ontologies based on software tools for conference organization support (Tool), and ontologies which summarize the experience of people with personal participation in organization of conferences (Insider). The conference dataset has also been subject to many works concerned with the occurrence of diverse patterns relevant for ontology matching [44, 45] and with the generation of complex correspondences [38]. In particular, as part of OAEI 2009 the organizers decided to add a specific track that is concerned with the generation of subsumption correspondences. In addition to the benchmark test data, the conference dataset has been chosen due to its semantic heterogeneity.[5]

In [39] reasoning with alignments has been mentioned as one of the ten open challenges in ontology matching. As argued in [31], the role of semantics and in particular the role of reasoning in the context of ontology matching has been neglected for a long time. This can be explained by the fact that many ontologies typically used as test cases within the matcher community are hierarchies that do not contain expressive constructs such as disjointness or property restrictions. Thus, reasoning tasks related to alignments between lightweight ontologies do often not require full-fledged reasoning techniques. In opposite to this, the conference ontologies (compare column 'Expressivity' in Table 4.2) exceed simple subsumption hierarchies due to their high expressivity. The conference dataset is therefore very well suited for measuring alignment coherence, as proposed in Section 3.1.5.

4.4 Test Metadata

The test data collections used to evaluate a matching algorithm/system are registered in a Test Data Repository, that must contain the following metadata for each test case. Such metadata is based on the specification of the corresponding repository content provided in the deliverable D5.1.

- test data ID - an ID uniquely identifying the test data,
- checksum,
- short name - for use in result presentation displayed in tables and figures,
- long name - the complete name of the data set used in textual result descriptions,
- description - a short description of the dataset,
- creator - a reference to the SEALS user that uploaded the dataset,
- version,
- classification - e.g. ontology vs. alignment,
- language - e.g. OWL-lite,
- format - e.g. ASCII,
- alignment - null or a reference to the alignment which aligns the ontology,
- ontologies - null or references to the ontologies aligned by this alignment,

[5]Detailed information can be found at `http://people.kmi.open.ac.uk/marta/oaei09/orientedMatching.html`.

- published - a boolean value indicating wether the the dataset is available for the public (notice that some datasets are used in blind tests where the reference alignment is not open),
- test generator - null or a reference to the test generator which generated the data:
  - description - textual description of the configuration,
  - configuration - the configuration (or a reference to the configuration file) that has been used to generate the data.
- access rights.

Some of the metadata is related to the dataset in general, while other data might be better associated with the concrete version. Further considerations have to clarify this issue.

4.5 Summary

This chapter has presented the test data that will be used in the first SEALS evaluation campaign. Due to the diversity of the tests, which emphasize different aspects of ontology matching, several aspects of matching systems will be evaluated.

Table 4.3 provides the information about where (URL) OAEI datasets can be found.

Next chapter presents the matching systems that will be considered as target for the evaluation campaigns.

Track	Test Data	URL
Comparison	benchmark	http://oaei.ontologymatching.org/2009/benchmarks/
Expressive ontologies	anatomy	http://webrum.uni-mannheim.de/math/lski/anatomy09/
	conference	http://nb.vse.cz/~svabo/oaei2009/
Directories and thesauri	directory	http://www.disi.unitn.it/~pane/OAEI/2009/directory/
	library	http://www.few.vu.nl/~aisaac/oaei2009/index.html
Oriented matching	benchmark-subs	http://people.kmi.open.ac.uk/marta/oaei09/orientedMatching.html
Instance matching	Eprints	http://www.scharffe.fr/events/oaei2009/
	TAP	http://www.scharffe.fr/events/oaei2009/
	IIMB	http://www.scharffe.fr/events/oaei2009/
	vlcr	http://www.cs.vu.nl/~laurah/oaei/2009/

Table 4.3: Test cases and URLs.

5. Evaluation Target: Systems Generating Alignments for Evaluation

For the first evaluation campaigns, we focus on five systems as potential evaluation targets: ASMOV (Section 5.1), Falcon-AO (Section 5.2), SAMBO (Section 5.3), Lily (Section 5.4), and AROMA (Section 5.5). They have participated in previous OAEI campaigns. Although these systems provide a good starting point, we do not restrict participation to the systems described in the following. An extended list of potential participants can be found in Appendix A, where we focus on more technical aspects.

5.1 ASMOV

The ASMOV system has been developed by INFOTECH Soft (`http://www.infotechsoft.com/`), a software development company headquartered in Miami, Florida (US). The company is mainly specializing in the design and development of healthcare software solutions. ASMOV is an abbreviation for "Automated Semantic Matching of Ontologies with Verification". ASMOV is designed to combine a comprehensive set of element-level and structure-level measures of similarity with a technique that uses formal semantics to verify whether computed correspondences comply with desired characteristics. A detailed description of the approach implemented in ASMOV can be found in [25].

ASMOV made its debut at the OAEI in 2007 with very good results, in particular it was one of the top three systems in both the benchmark and the anatomy track. ASMOV also participated in 2008 and 2009. It has continuously been developed further over the years. In particular, it turned out in 2008 that the evaluation results helped to detect an erroneous configuration of the system. ASMOV shows that the matching of ontologies might play an important role as part of commercial software solutions.

5.2 Falcon-AO

Falcon is an infrastructure for Semantic Web applications, which aims at providing technology for finding, aligning and learning ontologies. The matching system Falcon-AO is a prominent component of this infrastructure and participated at OAEI 2005 to 2007 as one of the best systems in the benchmark track. It is available for download at `http://iws.seu.edu.cn/projects/matching/`. Falcon-AO is easy to use and delivered with a graphical user interface that displays the results of the matching process. It is implemented in Java, and, presently, it is an open source project under the Apache 2.0 license, developed by a the XObjects research Group at the Institute of Web Science in Southeast University (China).

Falcon-AO has been described in several publications, see for example [23]. Falcon, internally, makes use of different elementary matchers (V-Doc, GMO and PBM), which require coordination rules and a similarity combination strategy. V-Doc takes a linguistic approach to ontology matching by constructing virtual documents for matchable entities. Document similarity can then be calculated via traditional vector space

techniques. GMO [22] is an iterative structural matcher. It uses RDF bipartite graphs to represent ontologies and computes structural similarities between domain entities and between statements (triples) in ontologies by recursively propagating similarities in the bipartite graphs. PBM uses a divide-and-conquer approach for finding block mappings between large-scale ontologies [24], which decreases the execution time without loosing quality.

5.3 SAMBO

The SAMBO matching system, mainly aimed at aligning and merging biomedical ontologies, has first been described in [28]. SAMBO has been developed at the Department of Computer and Information Science at the Linkpings University (Sweden) and is available at `http://www.ida.liu.se/~iislab/projects/SAMBO/`. Sambo combines several matching systems to generate the final alignment. Beside the use of syntactic and structural methods, it additionally exploits the Metathesaurus in the Unified Medical Language System (UMLS). See `http://www.nlm.nih.gov/research/umls/` for more information about UMLS. Exploiting available background knowledge as well as specific aspects of the medical domain seems to be the main reason why SAMBO was one of the top performers at OAEI with respect to the anatomy track.

In 2008 the OAEI anatomy track has for the first time introduced subtask #4, which aims at simulating user interaction by providing a partial reference alignment. Among the systems participating at this subtrack SAMBO and SAMBOdtf (SAMBO with double-threshold filtering) achieved the best evaluation results. The developer of the system picked up the idea and elaborately discussed whether and how a partial reference alignment can be used in ontology alignment in an extensive experimental study [27]. First results of the OAEI 2009 evaluation indicate that none of the 2009 participants could generate better results for this specific test setting.

5.4 Lily

Lily is a matching system that participated at OAEI for the first time [46] in 2007. It has been developed by Peng Wang at the School of Computer Science and Engineering, Southeast University (China) and is available at `http://ontomappinglab.googlepages.com/lily.htm`. Lily can be used for solving generic ontology matching problems as well as for matching large scale ontologies. It provides a simple graphical user interface that displays ontologies and generated alignments to support semiautomatic matching processes.

Lily also comprises a component for mapping debugging described in detail in [47], which is similar to the verification component of ASMOV. One of the principles that distinguishes Lily from other matching system is its focus on the notion of semantic subgraphs. The meaning of an ontological entity is determined by its connection to other ontological entities. Therefore, the meaning of a concept can be captured by extracting a semantic subgraph which relates the concepts itself with all surrounding entities. Although Lily uses no medical background knowledge, it could generate a surprisingly high amount of correct non-trivial correspondences with respect to the

OAEI anatomy track. This result requires additional evaluations to gain a better understanding.

5.5 AROMA

AROMA (Association Rule Ontology Matching Approach) matching system has been developed by Jérôme David at INRIA Rhône-Alpes, Montbonnot Saint-Martin (France). It is an hybrid, extensional and asymmetric matching approach designed to find out relations of equivalence and subsumption between entities, i.e. classes and properties, issued from two textual taxonomies (web directories or OWL ontologies). This approach makes use of the association rule paradigm, and a statistical interestingness measure. AROMA relies on the following assumption: An entity A will be more specific than or equivalent to an entity B if the vocabulary (i.e. terms and also data) used to describe A, its descendants, and its instances tends to be included in that of B.

During the matching process, AROMA passes through three successive stages: (1) The pre processing stage represents each entity, i.e. classes and properties, by a set of terms, (2) the second stage consists of the discovery of association rules between entities, and finally (3) the post processing stage aims at cleaning and enhancing the resulting alignment.

AROMA participated at the OAEI in 2008 for the first time. The source-code of the 2008 version is available at `http://www.inrialpes.fr/exmo/people/jdavid/oaei2008/AROMAsrc_oaei2008.jar`.

5.6 Tools Metadata

Metadata must be provided whenever a matching system/algorithm (target) is registered in the SEALS platform. The description of each target should contain the following data, which is based on the specification provided in the deliverable D6.1:

- target ID - an ID uniquely identifying the system,
- checksum,
- short name - for use in result presentation (max. 8 characters displayed in tables and figures),
- long name - the complete name of the system used in textual result descriptions,
- description - a short system description,
- developer - a reference to the developer, which is registered as SEALS user,
- classification (e.g. Run, Evaluation, Campaign) - alternatively this can be derived from the evaluation description,
- current version,
- access rights.

The tool metadata also describes the different versions of the tool:

- tool version - a version number of the tool,
- published,

- capabilities - output format generated by the system, matchable entities (concepts, properties, instances),
- hardware platform - description of hardware requirements for running the system (e.g. required main memory),
- OS platform - a list of OS on which the system can be executed,
- execution requirements - additional requirements (e.g. external program/server installation),
- installation script - a textfile which contains a command line call to the installation script.

Some of the metadata is related to the tool in general, while other data might be better associated with the concrete version. Further considerations have to clarify this issue.

Moreover, each matching system must have associated a zip file including all dependencies (libraries, external data sources, etc), which are not listed in the metadata under *execution requirements*. The zip file should also contain an installation script, which can be called automatically.

5.7 Summary

The target evaluation in this deliverable is a matching system. This chapter described some of the most important targets to be considered in a first evaluation campaign in SEALS. Such candidate systems have participated in several OAEI campaigns in different tracks.

We also described the metadata which has to be associated with each matching system. Some of the points listed as metadatas require further refinement. In particular concerning the technical aspects related to installation and successful execution of the systems. Therefore, we added an extended list of potential evaluation targets in Appendix A, where we additionally list contact information as well as a more detailed description of runtime requirements.

6. Manipulation and Visualization of Evaluation Results

In OAEI campaigns, the evaluation results revolve around a few measures (precision, recall, time). Some of them can be further analyzed into smaller measures (correct answers, non correct answers, expected answers). From these measures, it is possible to draw more elaborate pictures such as recall/precision graphs, scaling plot or chronological evolution.

We have designed APIs for matching (widely used) and evaluation (less used), but we have no common way to display and manipulate these measures. With the development of SEALS, there is an opportunity to define in a more general way what it means to display and manipulate evaluation results. This is by no means a one-size-fits-all solution, but this may be a starting point for a common view of evaluation results.

This chapter presents how the evaluation results are displayed in OAEI report results (Section 6.1), how the results could be presented in a multidimensional view of data results (Section 6.2), as well as which operations are desirable in order to provide more elaborated ways to manipulate such results (Section 6.3).

6.1 OAEI Evaluation Reports

6.1.1 Benchmark Results Report

In the benchmark track, the evaluation results are visualized in tabular and plot formats. Summary and full tables are used to show the results of precision and recall of each participant, by group of tests. For sake of brevity, we show the summary table (Table 6.1) containing the results of the 2009 campaign. Full tables can be accessed directly on the web site, as referred in Section 6.4.

Some plots are used to show the results in a more visual way. Figures 6.1 and 6.2 show precision and recall graphs. The first plot (Figure 6.1) has been drawn with only technical adaptation of the technique used in TREC and it is computed by averaging the graphs of each of the tests (instead to pure precision and recall). The results given by the participants are cut under a threshold necessary for achieving $n\%$ recall and the corresponding precision is computed. Systems for which these graphs are not meaningful (because they did not provide graded confidence values) are drawn in dashed lines.

In the second plot (Figure 6.2), each point expresses the position of a system with regard to precision and recall.

6.1.2 Anatomy Results Report

In the anatomy track, besides precision and recall, runtime is also measured. The participants run the respective systems on their own machines and the resulting runtime measurements provide an approximate basis for a useful comparison. Such results are presented in a tabular format, as shown in Tables 6.2 and 6.3.

system	refalign		edna		aflood		AgrMaker		AROMA		ASMOV		DSSim		GeRoMe		kosimap		Lily		MapPSO		RiMOM		SOBOM		TaxoMap	
test	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.
2009																												
1xx	1.00	1.00	0.96	1.00	1.00	1.00	0.98	0.98	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00	0.98	0.97	1.00	0.34
2xx	1.00	1.00	0.41	0.56	0.98	0.74	0.98	0.60	0.98	0.69	0.96	0.85	0.97	0.62	0.92	0.71	0.94	0.57	0.97	0.86	0.73	0.73	0.93	0.81	0.97	0.46	0.90	0.23
3xx	1.00	1.00	0.47	0.82	0.90	0.81	0.92	0.79	0.85	0.78	0.81	0.82	0.94	0.67	0.68	0.60	0.72	0.50	0.84	0.81	0.54	0.29	0.81	0.82	0.92	0.55	0.77	0.31
H-mean	1.00	1.00	0.43	0.59	0.98	0.80	0.99	0.62	0.94	0.69	0.95	0.87	0.97	0.66	0.91	0.73	0.91	0.59	0.97	0.88	0.63	0.61	0.93	0.82	0.98	0.44	0.86	0.26
Symmetric relaxed measures																												
H-mean	1.00	1.00	0.73	1.00	0.99	0.81	0.99	0.62	0.98	0.72	0.99	0.90	1.00	0.67	0.92	0.74	0.99	0.64	0.99	0.89	0.99	0.96	0.99	0.88	1.00	0.44	0.99	0.30
2008																												
1xx	1.00	1.00	0.96	1.00	1.00	1.00			1.00	1.00	1.00	1.00	1.00	1.00	0.96	0.79			1.00	1.00	0.92	1.00	1.00	1.00			1.00	0.34
2xx	1.00	1.00	0.41	0.56	0.96	0.69			0.96	0.70	0.95	0.85	0.97	0.64	0.56	0.52			0.97	0.86	0.48	0.53	0.96	0.82			0.95	0.21
3xx	1.00	1.00	0.47	0.82	0.95	0.66			0.82	0.71	0.81	0.77	0.90	0.71	0.61	0.40			0.87	0.81	0.49	0.25	0.80	0.81			0.92	0.21
H-mean	1.00	1.00	0.43	0.59	0.97	0.71			0.95	0.70	0.95	0.86	0.97	0.67	0.60	0.58			0.97	0.88	0.51	0.54	0.96	0.84			0.91	0.22

Table 6.1: Means of results obtained by participants on the benchmark test case (corresponding to harmonic means). The symmetric relaxed measure corresponds to the relaxed precision and recall measures of [10].

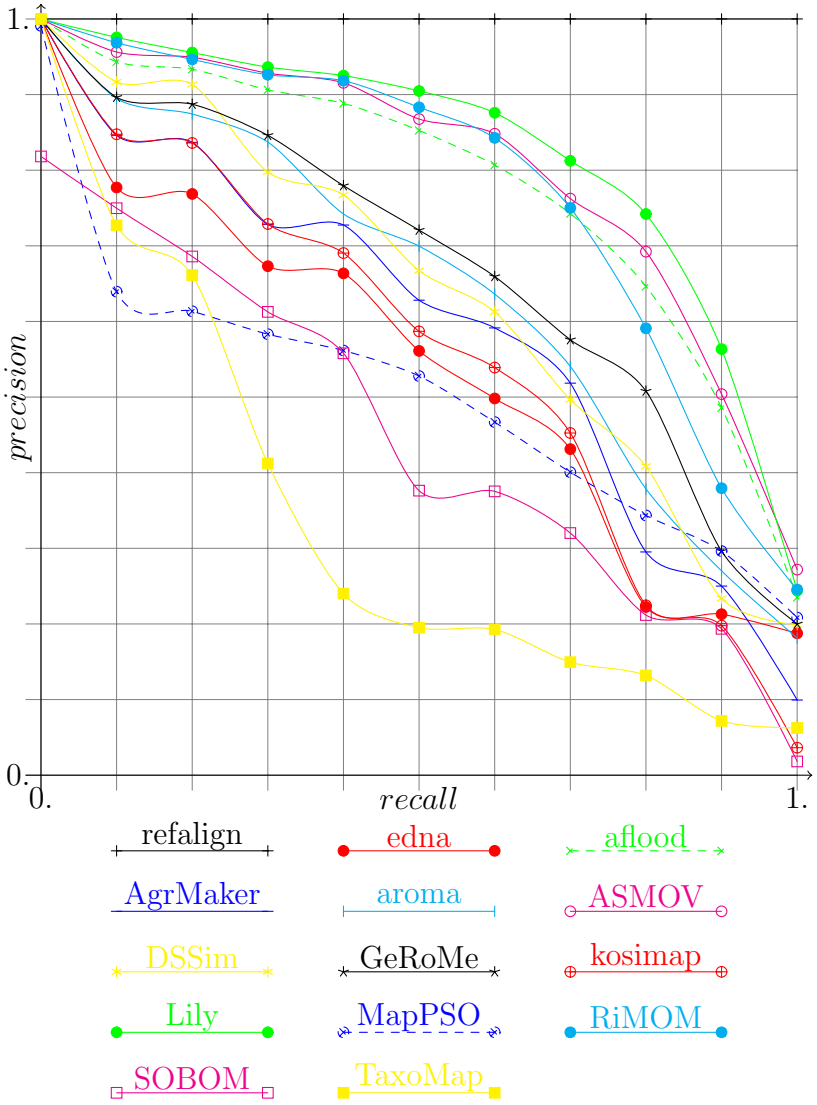


Figure 6.1: Precision/recall graphs for benchmarks.

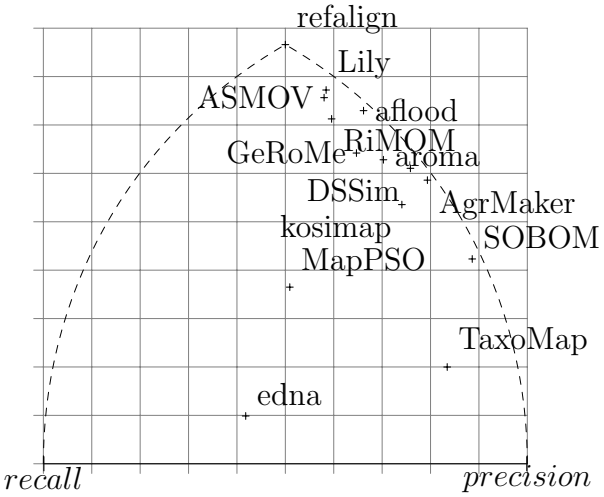


Figure 6.2: Expressing the position of a system with regard to precision and recall (benchmark).

Table 6.2 list the results of the participants in descending order with respect to the F-measure achieved for subtrack #1, where recall+ is defined as recall restricted to the subset of non trivial correspondences in the reference alignment (details in [16]). Table 6.3 refers to an alignment generated for task #1 resp. #4 as A_1 resp. A_4. So, the comparison of $A_1 \cup R_p$ resp. $A_4 \cup R_p$ with the reference alignment R is performed. The situation where the partial reference alignment is added after the matching process has been conducted against the situation where the partial reference alignment is available as additional resource used within the matching process.

6.1.3 Conference Results Report

Similar to what is done in the benchmark, conference track evaluates the results of participants against a reference alignment, generating values of precision, recall, and f–measure, which are computed for three different thresholds (t) – Table 6.4. The organizers also provide visualization of the results for an optimal threshold (Table 6.5). A dependency of F-measure on a threshold can be seen from the Figure 6.3

6.2 Multidimentional View

The matching results reported above can be represented through multidimensional views, as advocated by the proposer of OLAP [35] technology. We can reduce these dimensions to three (non ordinal) dimensions:

- System: the tool that is evaluated,
- Test: the test against which it is evaluated,
- Measure: the evaluation measure.

System	Task #1				Task #2			Task #3			Recall+	
	Runtime	Prec.	Rec.	F	Prec.	Rec.	F	Prec.	Rec.	F	#1	#3
SOBOM	≈ 19 min	0.952	0.777	0.855	-	-	-	-	-	-	0.431	-
AgrMaker	≈ 23 min	0.865	0.798	0.831	0.967	0.682	0.800	0.511	0.815	0.628	0.489	0.553
RiMOM	≈ 10 min	0.940	0.684	0.792	-	-	-	-	-	-	0.183	-
TaxoMap	≈ 12 min	0.870	0.678	0.762	0.953	0.609	0.743	0.458	0.716	0.559	0.222	0.319
DSSim	≈ 12 min	0.853	0.676	0.754	0.973	0.620	0.757	0.041	0.135	0.063	0.185	0.061
ASMOV	≈ 5 min	0.746	0.755	0.751	0.821	0.736	0.776	0.725	0.767	0.745	0.419	0.474
aflood	≈ 15 sec / 4 min	0.873	0.653	0.747	0.892	0.712	0.792	0.827	0.763	0.794	0.197	0.484
Lily	≈ 99 min	0.738	0.739	0.739	0.869	0.559	0.681	0.534	0.774	0.632	0.477	0.548
Aroma	≈ 1 min	0.775	0.678	0.723	-	-	-	-	-	-	0.368	-
kosimap	≈ 5 min	0.866	0.619	0.722	0.907	0.446	0.598	0.866	0.619	0.722	0.154	0.154

Table 6.2: Anatomy track participants and 2009 results with respect to runtime, precision, recall, recall+ and f-value.

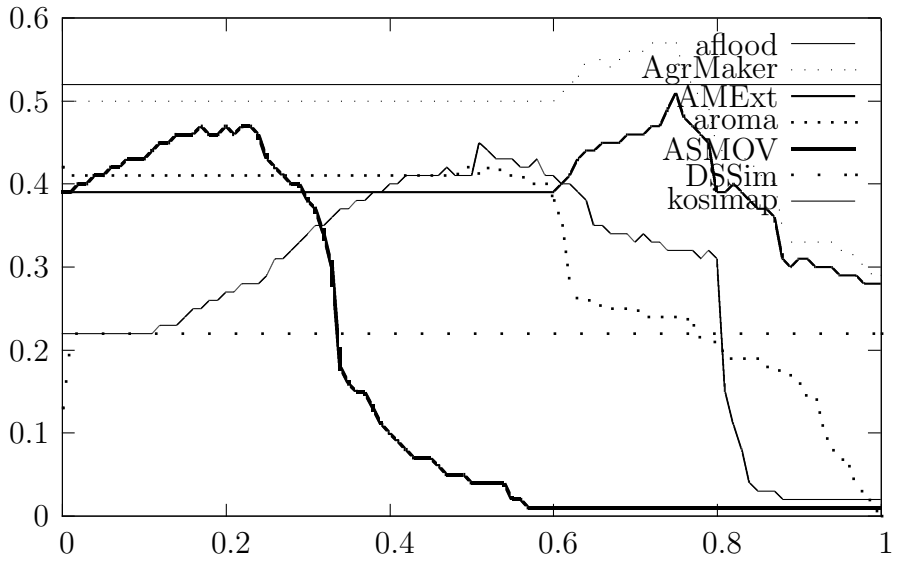


Figure 6.3: F-measures for each threshold (conference track).

System	Δ-Precision	Δ-Recall	Δ-f-Measure
SAMBOdtf_{2008}	$+0.020_{0.837\rightarrow0.856}$	$+0.003_{0.867\rightarrow0.870}$	$+0.011_{0.852\rightarrow0.863}$
ASMOV	$+0.034_{0.759\rightarrow0.792}$	$-0.018_{0.808\rightarrow0.790}$	$+0.009_{0.782\rightarrow0.791}$
$\text{aflood}_{\#3}$	$+0.005_{0.838\rightarrow0.843}$	$+0.003_{0.825\rightarrow0.827}$	$+0.004_{0.831\rightarrow0.835}$
TaxoMap	$+0.019_{0.878\rightarrow0.897}$	$-0.026_{0.732\rightarrow0.706}$	$-0.008_{0.798\rightarrow0.790}$
AgrMaker	$+0.128_{0.870\rightarrow0.998}$	$-0.181_{0.831\rightarrow0.650}$	$-0.063_{0.850\rightarrow0.787}$

Table 6.3: Changes in precision, recall and F-measure based on comparing $A_1 \cup R_p$, resp. $A_4 \cup R_p$, against reference alignment R (anatomy track).

	t=0.2			t=0.5			t=0.7		
	P	R	F-meas	P	R	F-meas	P	R	F-meas
aflood	48%	61%	52%	48%	61%	52%	48%	61%	52%
AgrMaker	45%	61%	50%	45%	61%	50%	6%	55%	56%
AMExt	30%	60%	39%	30%	60%	39%	41%	53%	46%
aroma	37%	49%	41%	38%	49%	42%	40%	19%	25%
ASMOV	58%	40%	47%	22%	3%	4%	5%	1%	1%
DSSim	15%	51%	22%	15%	51%	22%	15%	51%	22%
kosimap	18%	56%	27%	41%	43%	41%	70%	23%	33%

Table 6.4: Recall, precision and F-measure for three different thresholds (conference track).

matcher	threshold	P	R	F-meas
aflood	*	48%	61%	52%
AgrMaker	0.75	69%	51%	57%
AMExt	0.75	54%	50%	51%
aroma	0.53	39%	48%	42%
ASMOV	0.23	68%	38%	47%
DSSim	*	15%	51%	22%
kosimap	0.51	52%	42%	45%

Table 6.5: F-measure, Precision, and Recall for an optimal threshold for each matcher (conference track).

Each cell of this multidimensional table would contain the measure of the efficiency of one tool against one test. It is important that the data in this table be as precise as possible: the test should be clearly qualified and if there are 10 runs, for instance they should be described as "run 1", "run 2", etc. Similarly, the tool should denote one precise tool (version) and not a generic object like "Protégé" for instance. A rule of thumb may be that each index on a dimension cannot be further decomposed: a jar file for the system, a test file for the test and one particular measure for the measures.

The requirements that the dimensions be as precise as possible entail that the matrix will be very sparse. In counterpart, along each dimension, the indexes can be

refined. This should be described through extensive metadata:

- System: version, type, operating system, license, language, parameterization;
- Test: variant (benchmarks, 2009, 2xx, 203, 203-8), run, size, description;
- Measure: aggregate measure (sum, average, F-measure), parameters, normalization.

Fortunately, we have such metadata within the SEALS platform (see specifications above).

6.3 Operations

We do not want to display a huge detailed data cube. Some operations are needed to organize the data and make sense out of it. In a first approximation, we need the following operations:

- Selecting individually, or on criteria, the data that we want to consider (the results of a system, the results of a dataset, the results according to a particular measure);
- Projecting data on more dimensions according to particular split and aggregation functions;
- Ordering the data along a dimension according to a particular criterion (the year of the test, the name of the system, etc.);
- Grouping the data along a dimension according to a particular criterion (all the test in the same year, all the tests of that system, etc.);
- Aggregating the data of a group (that can be a unit) with a particular function (e.g., sum, average, harmonic means, variance, standard deviation).

This is basically SQL: SELECT, GROUP BY, ORDER BY, functions applied to dimensions instead of columns.

In addition, there are other functions that would be genuinely useful:

- Display as table (2 or three dimensional);
- Plot (according to time, scale);
- Save as... LaTeX, HTML, gnuplot, CSV, XML, etc.

We may also need "second-order" functions, i.e., functions which are computed against the dataset. This includes:

- generating a rank for a particular plan,
- normalizing the values with regard to the others,
- computing a distance from average.

Certainly, this simple presentation does not exhaust the data manipulation needs and we would have some specific measures or display to integrate in this landscape. So it would be useful to provide some standard way to integrate them.

For instance, in OAEI we produce our triangle view (see Figure 6.2) which should be relatively easy to produce and precision/recall graphs which are more tricky because they rely on the content of alignments.

6.4 Summary

This chapter presented the OAEI evaluation reports which comprise different types of tables and figures. These reports can be found in the OAEI results reports of the last years [16, 18, 5, 15] and they are also displayed as part of the online available reports.

- Results of the OAEI 2009 Benchmark Track
  `http://oaei.ontologymatching.org/2009/results/benchmarks.html`
- Results of the OAEI 2009 Anatomy Track
  `http://webrum.uni-mannheim.de/math/lski/anatomy09/results.html`
- Results of the OAEI 2009 Conference Track
  `http://nb.vse.cz/~svabo/oaei2009/eval.html`

Most of these reports have been generated by the use of spreadsheet software combined with applying evaluation scripts on the raw data.

The SEALS platform will allow a more flexible, simplified, and less error-prone generation of these reports, as well as will provide more elaborated ways to visualise and manipulate the results, as stated in Sections 6.2 and 6.3. For the first evaluation campaign, similar reports to those used in OAEI tracks must be provided, while improvements in terms of displaying and operations to manipulate results will be implemented for the second campaign.

7. Conclusions

In this report, we have documented the initial step of the SEALS methodology for designing systematic evaluation for ontology matching tools. According to this first step (referred to as 'Preliminary' in the general methodology), we have identified evaluation goals and assumptions, criteria and metrics as well as tools, datasets, and requirements for the evaluation campaigns to be carried out in the context of the SEALS project. In the following, we summarize the decisions made for the first evaluation campaign to be carried out in fall 2010. The goal of work in WP12 is to be able to support the identified evaluations in terms of evaluation components and workflows that run on the SEALS platform.

7.1 Goals and Assumptions

The goal of the first evaluation campaign is to evaluate the competence of matching systems with respect to isolated aspects and to compare matching systems on single criteria. For this purpose simple evaluations will be implemented that can apply a single matching system on a single criterion and store the result for further aggregation with other results.

Assumptions for this first campaign are that the matching systems can run independently and that it is possible and useful to compare systems based on different criteria separately.

7.2 Criteria and Metrics

For the first evaluation campaign a limited set of criteria will be used that can be tested using simple workflows as described in this deliverable. Criteria and measures to be considered are:

- Efficiency: runtime, memory consumption;
- Interoperability: compliance to the standard language RDFS and OWL-DL;
- Compliance with reference alignment: standard precision and recall, restricted semantic precision and recall;
- Coherence.

Scalability and Task-based evaluations will not be considered in the first evaluation campaign.

7.3 Tools and Datasets

For the first campaign, we have selected a subset of the datasets and systems that have been involved in past OAEI campaigns. Tools have been selected based on maturity and availability. Based on these criteria, we have identified the following tools as potential candidates to participate in the first evaluation campaign

- ASMOV,
- Falcon-OA,
- SAMBO,
- Lily,
- AROMA.

The datasets were selected based on the existence of reliable reference alignments and experiences with using the datasets in evaluation campaigns. These criteria are met by the following datasets:

- Benchmark,
- Conference,
- Anatomy.

Other datasets and dataset generators will not be considered for the first evaluation campaign.

7.4 Requirements

We specify the following requirements for matching systems and algorithms to perform evaluations and participate in the evaluation campaign:

- input ontologies written in the same language (and without syntax errors).
- without input alignment,
- with any kind of fixed parameters and any kind of fixed and general purpose resources (and corresponding libraries being provided),
- without any kind of user input nor training samples.
- providing an implementation for the API that will be specified for runnning systems in the SEALS platform,
- providing an output in the correct format.

Finally, we provided some examples of how evaluation results can be displayed, taking as basis OAEI evaluation reports. We also discussed how such results can be represented through multidimensional views and presented the set of operations that should be needed to better organize the data and make sense out of it. For the first evaluation campaign, similar reports to those used in OAEI campaigns will be provided, while for the second campaign visualization and manipulation of results require further improvements, as discussed in this deliverable.

References

[1] Alexandre Alves, Assaf Arkin, Sid Askary, Ben Bloch, Francisco Curbera, Yaron Goland, Neelakantan Kartha, Sterling, Dieter König, Vinkesh Mehta, Satish Thatte, Danny van der Rijn, Prasad Yendluri, and Alex Yiu. Web services business process execution language version 2.0. OASIS Standard Committee, April 2007.

[2] Benjamin Ashpole, Marc Ehrig, Jérôme Euzenat, and Heiner Stuckenschmidt, editors. *Proc. K-CAP Workshop on Integrating Ontologies*, Banff (CA), October 2005.

[3] Oliver Bodenreider, Terry F. Hayamizu, Martin Ringwald, Sherri De Coronado, and Songmao Zhang. Of mice and men: Aligning mouse and human anatomies. In *AIMA 2005 Symposium Proceedings*, pages 61–65, 2005.

[4] 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.

[5] Caterina Caracciolo, Jérôme Euzenat, Laura Hollink, Ryutaro Ichise, Antoine Isaac, Véronique Malaisé, Christian Meilicke, Juan Pane, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, and Vojtech Svátek. Results of the ontology alignment evaluation initiative 2008. In *OM*, 2008.

[6] Raúl Garcí Castro, Diana Maynard, Holger Wache, Doug Foxvog, and Rafael Gonzá Cabero. Specification of a methodology, general criteria, and benchmark suites for benchmarking ontology tools. Deliverable D2.1.4, Knowledge Web, 2005.

[7] Hong-Hai Do, Sergei Melnik, and Erhard Rahm. Comparison of schema matching evaluations. In *Proc. Workshop on Web, Web-Services, and Database Systems*, volume 2593 of *Lecture notes in computer science*, pages 221–237, Erfurt (DE), 2002.

[8] Marc Ehrig. *Ontology alignment: bridging the semantic gap*. PhD thesis, Universität Fridericiana zu Karlsruhe, Karlsruhe (DE), 2006.

[9] Marc Ehrig and Jérôme Euzenat. Relaxed precision and recall for ontology matching. In Benjamin Ashpole, Marc Ehrig, Jérôme Euzenat, and Heiner Stuckenschmidt, editors, *Proceedings of the Workshop on Integrating Ontologies*, volume 156, page 8. CEUR-WS.org, August, 2005 2005.

[10] Marc Ehrig, Steffen Staab, and York Sure. Bootstrapping ontology alignment methods with APFEL. In *Proc. 4th International Semantic Web Conference (ISWC)*, volume 3729 of *Lecture notes in computer science*, pages 186–200, Galway (IE), 2005.

[11] Jérôme Euzenat. Towards composing and benchmarking ontology alignments. In *Proc. ISWC Workshop on Semantic Integration*, pages 165–166, Sanibel Island (FL US), 2003.

[12] Jérôme Euzenat. An API for ontology alignment. In *Proceedings of the 3rd International Semantic Web Conference (ISWC)*, pages 698–712, Hiroshima (JP), 2004.

[13] Jérôme Euzenat. Semantic precision and recall for ontology alignment evaluation. In *Proc. 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 248–253, Hyderabad (IN), 2007.

[14] Jérôme Euzenat, Marc Ehrig, and Raúl Garcí Castro. Specification of a benchmarking methodology for alignment techniques. Deliverable D2.2.2, Knowledge Web, 2005.

[15] Jérôme Euzenat, Alfio Ferrara, Laura Hollink, Véronique Malaisé, Christian Meilicke, Andriy Nikolov, Juan Pane, Francois Scharffe, Pavel Shvaiko, Vassilis Spiliopoulos, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, and Vojtech Svátek.

[16] Jérôme Euzenat, Antoine Isaac, Christian Meilicke, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Svab, Vojtech Svatek, Willem Robert van Hage, and Mikalai Yatskevich. Results of the ontology alignment evaluation initiative 2007. In Pavel Shvaiko, Jrme Euzenat, Fausto Giunchiglia, and Bin He, editors, *Proceedings of the 2nd ISWC international workshop on Ontology Matching, Busan (KR)*, pages 96–132, 2007.

[17] Jérôme Euzenat, Nabil Layaïda, and Victor Dias. A semantic framework for multimedia document adaptation. In *Proc. 18th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 31–36, Acapulco (MX), 2003.

[18] Jérôme Euzenat, Malgorzata Mochol, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Svab, Vojtech Svatek, Willem Robert van Hage, and Mikalai Yatskevich. Results of the ontology alignment evaluation initiative 2006. In Pavel Shvaiko, Jérôme Euzenat, Natalya Noy, Heiner Stuckenschmidt, Richard Benjamins, and Michael Uschold, editors, *Proceedings of the ISWC international workshop on Ontology Matching, Athens (GA US)*, pages 73–95, 2006.

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

[20] Daniel Fleischhacker and Heiner Stuckenschmidt. Implementing semantic precision and recall. In *Proceedings of the ISWC 2009 Workshop on Ontology Matching*, Washington, DC, US, 2009.

[21] Christian Freksa. Temporal reasoning based on semi-intervals. *Artificial Intelligence*, 54(1-2):199–227, 1992.

[22] Wei Hu, Ningsheng Jian, Yuzhong Qu, and Yanbing Wang. Gmo: A graph matching for ontologies. In *Proceedings of the K-CAP Workshop on Integrating Ontologies*, 2005.

[23] Wei Hu and Yuzhong Qu. Falcon-ao: A practical ontology matching system. *Journal of Web Semantics*, 6:237–239, 2008.

[24] Wei Hu, Yuzhong Qu, and Gong Cheng. Matching large ontologies: A divide-and-conquer approach. *Data & Knowledge Engineering*, 67:140–160, 2008.

[25] Yves R. Jean-Mary, E. Patrick Shironoshitaa, and Mansur R. Kabuka. Ontology matching with semantic verification. *Web Semantics: Science, Services and Agents on theWorldWideWeb*, 158, 2009.

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

[27] Patrick Lambrix and Qiang Liu. Using partial reference alignments to align ontologies. In *Proceedings of the European Semantic Web Conference*, 2009.

[28] Patrick Lambrix and He Tan. Sambo - a system for aligning and merging biomedical ontologies. *Web Semantics: Science, Services and Agents on the World Wide Web*, 4:196–206, 2006.

[29] Yoonkyong Lee, Mayssam Sayyadian, AnHai Doan, and Arnon S. Rosenthal. etuner: tuning schema matching software using synthetic scenarios. *The VLDB Journal*, 16(1):97–122, 2007.

[30] Vladimir Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Doklady akademii nauk SSSR*, 163(4):845–848, 1965. In Russian. English Translation in Soviet Physics Doklady, 10(8) p. 707–710, 1966.

[31] Christian Meilicke. The relevance of reasoning and alignment incoherence in ontology matching. In *Proccedings of the ESWC 2009 PhD Symposium*, Heraklion, Greece, 2009.

[32] Christian Meilicke and Heiner Stuckenschmidt. Analyzing mapping extraction approaches. In *Proceedings of the ISWC 2007 Workshop on Ontology Matching*, Busan, Korea, 2007.

[33] Christian Meilicke and Heiner Stuckenschmidt. Incoherence as a basis for measuring the quality of ontology mappings. In *Proc. of the ISWC 2008 Workshop on Ontology Matching*, Karlsruhe, Germany, 2008.

[34] 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.

[35] Carl Nolan. Manipulate and query olap data using adomd and multidimensional expressions. *Microsoft Systems Journal*, (63):51–59, 1999.

[36] Guilin Qi and Anthony Hunter. Measuring incoherence in description logic-based ontologies. In *Proceedings of the 6th International Semantic Web Conference*, Busan, Korea, 2007.

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

[38] Dominique Ritze, Christian Meilicke, Ondrej Svab-Zamazal, and Heiner Stuckenschmidt. A pattern-based ontology matching approach for detecting complex correspondences. In *Proceedings of the ISWC 2009 workshop on ontology matching*, 2009.

[39] Pavel Shvaiko and Jérôme Euzenat. Ten challenges for ontology matching. In *Proceedings of the 7th international conference on ontologies, dataBases, and applications of semantics*, Monterrey, Mexico, 2008.

[40] Pavel Shvaiko, Jérôme Euzenat, Fausto Giunchiglia, and Heiner Stuckenschmidt, editors. *Proceedings of the 3rd International Workshop on Ontology Matching (OM-2008) Collocated with the 7th International Semantic Web Conference (ISWC-2008), Karlsruhe, Germany, October 26, 2008*, volume 431 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[41] Heiner Stuckenschmidt, Luciano Serafini, and Holger Wache. Reasoning about ontology mappings. In *Proceedings of the ECAI workshop on contextual representation and reasoning*, 2006.

[42] York Sure, Oscar Corcho, Jérôme Euzenat, and Todd Hughes, editors. *Proc. 3rd ISWC Workshop on Evaluation of Ontology-based tools (EON)*, Hiroshima (JP), 2004.

[43] Ondrej Svab, Vojtech Svatek, Petr Berka, Dusan Rak, and Petr Tomasek. Ontofarm: Towards an experimental collection of parallel ontologies. In *Poster Track of ISWC*, Galway, Ireland, 2005.

[44] Ondrej Svab, Vojtech Svatek, and Heiner Stuckenschmidt. A study in empirical and 'casuistic' analysis of ontology mapping results. In *Proceedings of the 4th European Semantic Web Conference*, 2007.

[45] Ondrej Svab-Zamazal, Vojtech Svatek, Christian Meilicke, and Heiner Stuckenschmidt. Testing the impact of pattern-based ontology refactoring on ontology matching results. In *Proceedings of the ISWC 2008 Workshop on Ontology Matching*, Karlsruhe, Germany, 2008.

[46] Peng Wang and Baowen Xu. Lily: The results for the ontology alignment contest oaei 2007. In *Proceedings of the ISWC 2007 workshop on ontology matching*, Busan, Korea, 2007.

[47] Peng Wang and Baowen Xu. Debugging ontology mapping: A static method. *Computing and Informatics*, 27:21–36, 2008.

A. List of Tools

In Chapter 5 we picked out five matching tools from the large set of all available tools. In the following we present a list of tools to be considered as potential evaluation targets for a first evaluation campaign. The description of the tools was collected from a questionary we had sent to the developers.

It can be expected that most developers of these tools are interested in participating in a SEALS evaluation or in using the SEALS platform, respectively. Tools are ordered lexically due to their acronym. The property *Tool feature* can have the values *Tbox, Tbox+, Abox*. *Tbox* means that the system can match concepts and properties by equivalence, + includes subsumption correspondences, and *Abox* refers to matching systems capable of matching instances.

AgrMaker

Fullname: AgreementMaker (a.k.a. AgrMaker, amaker)

Developer Contact: Ulas Keles, ukeles@cs.uic.edu

Tool Feature: Tbox

Implementation Language: Java 1.6

Execution Environments:

Operating System/Version: tested on Windows XP & Vista

Non standard libraries: None

External program install: None

External server install: None

Webpage: `http://www.agreementmaker.org/index.html`

License: not specified

Anchor-Flood

Fullname: Anchor-Flood (a.k.a. aflood)

Developer Contact: Hanif Seddiqui, hanif@kde.ics.tut.ac.jp

Tool Feature: Tbox

Implementation Language: Java 1.6

Execution Environments:

Operating System/Version: Windows Vista Business 32bit (Service pack 1)

Non standard libraries: None

External program install: None

External server install: None

Webpage: `http://www.kde.ics.tut.ac.jp/~hanif/res/2009/`

License: not specified

AROMA

Fullname: AROMA

Developer Contact: Jerome David, jerome.david@inrialpes.fr

Tool Feature: Tbox

Implementation Language: Java

Execution Environments:

Operating System/Version: any.

Non standard libraries: n.a.

External program install: n.a.

External server install: n.a.

Webpage: `http://aroma.gforge.inria.fr/`

License: not specified

ASMOV

Fullname: ASMOV

Developer Contact: Yves R. Jean-Mary, reggie@infotechsoft.com

Tool Feature: Tbox+, Abox

Implementation Language: Java 6

Execution Environments:

Operating System/Version: Platform independent, tested on Windows and a FreeBSD systems

Non standard libraries: UMLS library (2009AA release), Jena v2.5.7 (ARP parser), all in package with ASMOV

External program install: WordNet (database only [version 2.1]) & UMLS (file indexing option)

External server install: None

Webpage: `http://www.infotechsoft.com/`

License: not specified

DSSim

Fullname: DSSim

Developer Contact: Miklos Nagy, mn2336@student.open.ac.uk

Tool Feature: Tbox+, Abox

Implementation Language: Java 1.6

Execution Environments:

Operating System/Version: MacOS X
Non standard libraries: COLT, JAVAWS, JUNG, SECONDSTRING, STAX
External program install: None
External server install: None

Webpage: tool webpage is unknown / does not exist

License: not specified

Falcon-AO

Fullname: Falcon-AO (a.k.a. Falcon)

Developer Contact: Wei Hu, whu@seu.edu.cn

Tool Feature: Tbox

Implementation Language: Java (version unknown)

Execution Environments:

Operating System/Version: Platform independent
Non standard libraries: n.a.
External program install: n.a.
External server install: n.a.

Webpage: `http://iws.seu.edu.cn/projects/matching/`

License: Apache 2.0

GeRoMe

Fullname: GeRoMeSuite

Developer Contact: Christoph Quix, quix@cs.rwth-aachen.de

Tool Feature: Tbox

Implementation Language: Java 1.5 (or newer)

Execution Environments:

Operating System/Version: Tested on Windows XP and Vista
Non standard libraries: JWNL for wordnet.
External program install: None (Wordnet-Files ?)
External server install: None

Webpage: `http://dbis.rwth-aachen.de/cms/projects/GeRoMeSuite`

License: not specified

HMatch

Fullname: HMatch - The ISLab Ontology Matching System

Developer Contact: Alfio Ferrara, ferrara@dico.unimi.it

Tool Feature: Tbox, Abox

Implementation Language: n.a.

Execution Environments:

Operating System/Version: n.a.
Non standard libraries: n.a.
External program install: n.a.
External server install: n.a.

Webpage: `http://islab.dico.unimi.it/hmatch/`

License: not specified

KOSIMap

Fullname: KOSIMap

Developer Contact: Quentin H. Reul, q.reul@abdn.ac.uk

Tool Feature: Tbox

Implementation Language: Java 1.5 for Mac

Execution Environments:

Operating System/Version: MacOS Version 10.5.8.

Non standard libraries: OWL API, FaCT++ API, Pellet, SimMetrics API

External program install: None

External server install: None

Webpage: n.a.

License: not specified

Lily

Fullname: Lily

Developer Contact: Peng Wang, pwangseu@gmail.com

Tool Feature: Tbox

Implementation Language: Java, C++ / JDK 1.6 (C++ source has been compiled as a DLL file and is called via Java)

Execution Environments:

Operating System/Version: Windows XP/2000

Non standard libraries: Jena, Dom4j,

External program install: None

External server install: None

Webpage: `http://ontomappinglab.googlepages.com/lily.htm`

License: not specified

MapPSO

Fullname: MapPSO - Mapping by Particle Swarm Optimisation

Developer Contact: J´urgen Bock, bock@fzi.de

Tool Feature: Tbox

Implementation Language: Java 1.5

Execution Environments:

- Operating System/Version: Platform independent, since based on Java. Successfully tested on Windows, Linux and MacOS..
- Non standard libraries: So far only the JWNL shipped with the Alignment API is used in order to access WordNet.
- External program install: WordNet. Location of the WordNet dictionary is provided via a parameter file with all other parameters used by the tool.
- External server install: None

Webpage: n.a.

License: not specified

OLA

Fullname: OLA (a.k.a. OLA 2)

Developer Contact: Jrme Euzenat, Jerome.Euzenat@inrialpes.fr

Tool Feature: Tbox

Implementation Language: Java 1.5

Execution Environments:

- Operating System/Version: Independent
- Non standard libraries: JWNL
- External program install: WordNet
- External server install: none

Webpage: n.a.

License: not specified

RiMOM

Fullname: RiMOM

Developer Contact: Jie Tang, tangjie@keg.cs.tsinghua.edu.cn

Tool Feature: Tbox+, Abox

Implementation Language: n.a.

Execution Environments:

- Operating System/Version: Window XP, Vista and Ubuntu Server 8.10
- Non standard libraries: third party libraries used in RiMOM are already contained in the lib folder.
- External program install: WordNet 2.0 be installed in the system and the install path be correctly configured in the etc/file_properties.xml
- External server install: None

Webpage: tool webpage is unknown / does not exist.

License: not specified

SAMBO

Fullname: SAMBO - System for Aligning and Merging of Biomedical Ontologies

Developer Contact: Patrick Lambrix, patla@ida.liu.se

Tool Feature: Tbox

Implementation Language: n.a.

Execution Environments:

- Operating System/Version: n.a.
- Non standard libraries: n.a.
- External program install: n.a.
- External server install: n.a.

Webpage: `http://www.ida.liu.se/~iislab/projects/SAMBO/index.html`

License: not specified

SOBOM

Fullname: SOBOM

Developer Contact: Peigang Xu, xpg0312@163.com

Tool Feature: Tbox

Implementation Language: Java 1.5 or 1.6

Execution Environments:

- Operating System/Version: Windows Vista Ultimate 32bit
- Non standard libraries: Jena2.5
- External program install: None
- External server install: None

Webpage: n.a.

License: not specified

TaxoMap

Fullname: TaxoMap

Developer Contact: Faycal Hamdi, Faycal.Hamdi@lri.fr

Tool Feature: Tbox+

Implementation Language: Java (all versions ?)

Execution Environments:

- Operating System/Version: Platform independent
- Non standard libraries: n.a.
- External program install: TreeTagger (but can also be delivered as part of the tool)
- External server install: MySQL server

Webpage: tool webpage is unknown / does not exist

License: not specified

Table A.1 summarizes the description of the tools, focusing on the main features we need to know in order to run them in the SEALS platform.

Tool	Language	OS	Ext. Libraries
AgrMaker	Java 1.6	Windows	None
Anchor-Flood	Java 1.6	Windows	None
AROMA	Java 1.6	Independent	None
ASMOV	Java 1.6	Independent	UMLS (2009 AA release), Jena v2.5.7
DSSim	Java 1.6	MacOS	COLT, JavaWS, JUNG, SECONDSTRING, STAX
Falcon-AO	Java	Independent	n.a
GeRoMe	Java 1.5	Windows	JWNL
HMatch	n.a	n.a	n.a
KOSIMap	Java 1.5	MacOS	OWL API, FaCT++, Pellet, SimMetrics API
Lily	Java 1.6/C++	Windows	Jena, Dom4j
MapPSO	Java 1.5	Independent	JWNL
OLA	Java 1.5	Independent	JWNL
RiMOM	n.a	Windows/Linux	n.a.
SAMBO	n.a	n.a	n.a
SOBOM	Java 1.6	Windows	Jena v2.5
TaxoMap	Java	Independent	n.a

Table A.1: Runtime features of evaluation targets.