RSDL Workbench Results for OAEI 2015*

Simon Schwichtenberg and Gregor Engels

University of Paderborn, s-lab – Software Quality Lab, Germany
{simon.schwichtenberg,engels}@upb.de

Abstract The vision of automatic service composition is to automatically combine single services to a software solution that satisfies certain requirements. Comprehensive service specifications are needed to receive suitable compositions. The Rich Service Description Language (RSDL) has been developed and can be used to specify ontological and behavioral semantics of services comprehensively. Part of a service's RSDL specification is its domain ontology that comprises concepts to describe, e.g., the service's input and output parameters. The RSDL Workbench (RSDLWB) is a platform that provides tools for the specification, matching, and composition of services. In particular, RSDLWB matches ontologies that are part of RSDL specifications. In this paper, we present that ontology matcher and the evaluation results as determined by the Ontology Alignment Evaluation Initiative (OAEI). Compared to the last campaign, we improved the runtime while maintaining the quality level of the produced alignments.

1 Presentation of the system

RSDLWB is a collection of tools for the specification, matching, and automatic composition of services. On the one hand, service requesters need to specify *service requests*, i.e., the requirements for services they need. On the other hand, service providers need to specify their *service offers*, i.e., the services they provide. Comprehensive, multi-faceted specifications that describe structural as far as behavioral aspects are needed to determine proper service compositions. A RSDL specification of a service defines its individual ontology and operation signatures. Besides these structural aspects of a service, the specifications also comprises behavioral aspects as pre- and postconditions of operations and operation protocols.

The ontologies describe the concepts and relations that appear in the domain of a service, e.g. to describe parameter types of operations. Within this paper, only ontologies that are part of service specifications are in the focus. Comprehensive specifications can be created in languages like RSDL [4], which is similar to the Web Ontology Language for Services (OWL-S).

The task of matching requests and services is called *Service Discovery*. For the matching of multi-faceted specifications, multiple matchers are needed, while each is specialized for either the matching of ontologies, operations, or protocols [4].

Since service specifications are created independently, the ontologies they contain are most likely to be heterogeneous in terms of their terminology or conceptualization.

* This work was partially supported by the German Research Foundation (DFG) within the Collaborative Research Centre "On-The-Fly Computing" (SFB 901)

Two ontologies might contain equivalent concepts, while both use different labels or logical hierarchies. The task of ontology matching is to find correspondences between concepts in the ontologies of service requests and offers. Ontology matchers produce *ontology alignments*, i.e., sets of mappings.

RSDLWB also enables the transformation of individuals from one ontology to another, based on the previously calculated ontology alignment. In this context, individuals are instances of the classes defined in an ontology. Within the RSDL specification of a service, the pre- and postconditions of its operations are denoted by Visual Contracts (VCs) [2], i.e., a variant of graph grammar rules. Each rule consists of a Left-Hand Side (LHS) and a Right-Hand Side (RHS). The LHS and RHS of the graph grammar rules are instance graphs that conform to the service's individual ontology. The LHS is the precondition that must hold before the operation can be executed, whereas the RHS describes the effects of the execution. In the short notation of VCs, instances that only appear on the LHS are deleted and marked in red, instances that only appear on the RHS are created and marked in green, and instances that appear on both sides are preserved and marked in black.

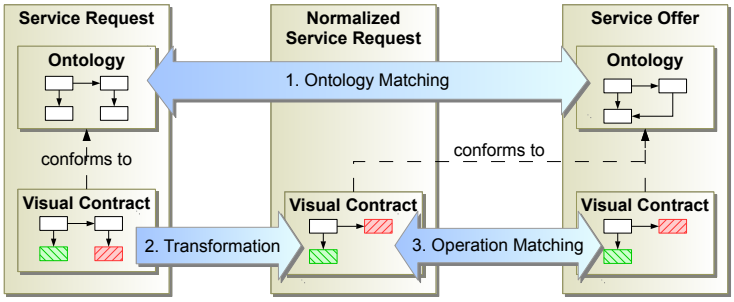

Figure 1: Matching Process [8]

Ontology matching is a prerequisite for the operation matcher that is described in [4]. This matcher requires that specifications conform to the same ontology. Consequently, the heterogeneous ontologies that are contained in request and offer specifications have to be normalized so that operation matching can be applied. A unique feature of RSDLWB is that it produces ontology alignments in terms of relational Query View Transformation (QVT) model transformation scripts. These transformation scripts can be used as a basis to normalize specifications, i.e., to reconcile VCs so that they conform to the same ontology. The relationship between ontology and operation matching is shown in Fig. 1: In a first step, two ontologies are matched and a transformation script is produced. The input of the transformation are the VCs of the request and the output are the corresponding VCs that conform to the ontology of the offer. These normalized VCs are used for the operation matching.

1.1 State, purpose, general statement

The purpose of RSDLWB's ontology matcher is to match ontologies that are part of (RSDL) service specifications. RSDLWB is still under development and continuous improvement. In its current shape, the matcher supports the following OAEI tracks:

benchmark, *anatomy*, *conference*, and *largebio*. The focus for this year's OAEI campaign was to improve the runtime performance of the matcher.

1.2 Specific techniques used

For the OAEI 2015 campaign, a new version of RSDLWB's ontology matcher was introduced that is specialized for OAEI. This version includes the following major changes: (1) Unnecessary time for the conversion of different model representations was eliminated. In particular, the abstraction layer that translates Web Ontology Language (OWL) ontologies to their Ecore representation was removed, so that OWL can be processed directly without an adapter. Furthermore, the matcher does not implement the EMFCompare API[1] anymore, but implements the Semantic Evaluation At Large Scale (SEALS) API[2] directly. (2) In order to avoid quadratic runtime complexity, the matching algorithm does not create a complete similarity matrix to check all possible concept pairs anymore. Instead, a simple heuristic was used as explained in Sect. 1.2. (3) Machine learning techniques were applied to obtain a classifier that can match concept pairs. This classifier was trained on the basis of reference alignments provided by the OAEI tracks.

Algorithm The RSDL Workbench matches `Classes`, `DataProperties`, and `ObjectProperties` independently. At first, a pre-processing normalizes the labels of the concepts. The labels are split into tokens at uppercase characters (CamelCase) or special delimiters like underscores. Each single token is *normalized* by lowercasing and suppression of non-alphabetical characters. These single tokens are concatenated with an underscore to form the normalized label. For example, the concept `OrganizingCommittee` becomes `organizing_committee` or `Positive_Review` becomes `positive_review`.

In a next step, *seed pairs* are selected, i.e., concept pairs that are likely to match. Seed pairs are selected by two different heuristics. The first heuristic selects seed pairs that have identical normalized labels. When identical normalized labels are not available by a sufficient amount, a different heuristic is used. This second heuristic selects seed pairs of concepts that are on the same hierarchy level in respect to the ontology they are defined in. Roots are on level 0, the roots' subclasses are on level 1, and so forth.

Next, the seed pairs are classified by means of the classifier that is described in the following sections. A post-processing ranks the positively classified pairs descending by an aggregated similarity value, i.e., the Euclidean distance of the feature vectors that are described in the following.

Starting from the most similar pairs, these pairs are added to the output alignment in a greedy manner. When a pair $\langle c_1, c_2 \rangle$ is added to the alignment, all other pairs that contain either c_1 or c_2 are discarded. Consequently, the matcher produces only 1:1 mappings.

Machine Learning Features Machine learning classification relies on statistical patterns found in features of example objects. RSDLWB's classifier classifies whether pairs of

[1] `https://www.eclipse.org/emf/compare/`
[2] `http://www.seals-project.eu/`

concepts do match or not. The classification is conducted on the basis of feature vectors. In particular, these feature vectors comprise several similarity values. We experimented with different features that were developed with runtime performance in mind. The features that had been used to train the classifier are described in this paragraph. Further features based on background knowledge are planned for future work.

Hierarchy level similarity relates the hierarchy position of two concepts with respect to the ontology they are defined in. The intuition is that concepts that are located on the similar hierarchy level are similar.

Outdegree similarity relates the outdegree of two concepts. An ontology is a directed triple graph, where each triple (s, p, o) consists of a subject s, predicate p, and an object o. The outdegree of concept c is the number of triples where c is the subject and p and o are free variables. The intuition is that concepts that have a similar number of outgoing edges are similar.

Property count similarity relates the number of properties of two concepts. The intuition is that concepts that have a similar number of properties are similar.

Shared token similarity relates the set of tokens from the labels of two concepts. The Jaccard coefficient is calculated for these token sets. The intuition is that concepts whose labels share many tokens are similar.

Property shared token similarity relates the set of tokens of the labels of all direct property labels of two concepts: At first, all direct properties of a concept are determined. Their labels are split into token sets. Then the Jaccard coefficient is calculated for these token sets. The intuition is that the property labels of two similar concepts share many tokens.

Neighborhood shared token similarity relates the set of tokens of the labels of all direct neighbors of two concepts. A neighbor n of concept c is determined by the triple (c, p, n), where p is a free variable. The Jaccard coefficient is calculated for these token sets. The intuition is that the token sets created from the labels of all adjacent neighbors of two similar concepts have a high overlap.

Token count similarity relates the number of tokens of the labels of two concepts. The intuition is the labels of similar concepts have a similar amount of tokens.

Substring length similarity relates the string length of two concept labels. If the label of a concept c_1 is contained in label of another concept c_2, this similarity is defined as the quotient of c_1's and c_2's label length. Otherwise, the similarity is 0. The intuition is that the longer the common character sequence is, the more similar the concepts are.

Equivalent shared token similarity relates the set of tokens of the labels of all equivalents of two concepts: At first, all equivalents of a concept are determined according to the *#equivalentClass* relation. Their labels are split into token sets for which the Jaccard coefficient is calculated afterwards. The intuition is that the token sets created from all equivalent classes of similar concepts have a high overlap.

Corpus and Classifier Creation In order to train classifiers with machine learning techniques, positive and negative examples were needed, in which statistical patterns are found that allow distinguishing correct from incorrect matches. A corpus is a set of positive and negative examples and is divided into a training and a validation set. An

example is a vector of feature values for the concept pair $\langle c_1, c_2 \rangle$ plus a matching class, which determines if the concept pair is a correct mapping or not.

Two corpora had been created for each of the OAEI tracks *benchmark*, *anatomy*, *conference*, and *largebio*: One corpus for class and another for property matching. The set of positive examples $I_\oplus$ is the set of mappings that are included in the given reference alignment R_{O_1,O_2}:

$$I_\oplus := R_{O_1,O_2}$$

In contrast to $I_\oplus$, the set of negative examples $I_\ominus$ had to be generated. Randomly generated incorrect pairs are likely to differ a lot from correct pairs, i.e., the values of their feature vectors deviate a lot. Consequently, correct and incorrect pairs can be easily distinguished. However, it is more meaningful to train a classifier on examples that show the subtle differences between correct and incorrect pairs. That is the reason why the set of incorrect pairs $I_\ominus$ was generated depending on $I_\oplus$: Originating from a correct pair $\langle c_1, c_2 \rangle \in I_\oplus$, incorrect pairs were selected from the Cartesian product of c_1's and c_2's direct subclasses. The idea is that c_1's and c_2's subclasses are similar, because c_1 and c_2 form a correct pair. Let $\langle c_1, c_2 \rangle \in I_\oplus$. S_{c_i} is the set of direct subclasses of c_i and $S'_c := S_{c_i} \cup \{c_i\}$. Incorrect pairs are selected from the Cartesian product $S'_{c_1} \times S'_{c_2}$.

$$I_\ominus := S'_{c_1} \times S'_{c_2} \backslash I_\oplus \quad \text{for all } \langle c_1, c_2 \rangle \in I_\oplus$$

The number of generated negative examples was limited by the number of the given positive examples in order to receive balanced sets of positive and negative examples. All examples were distributed by a 66/33 percentage ratio over the training and validation set.

Tab. 1 shows a short evaluation of the quality of the previously described features. In particular, the information gain metric [6] was calculated on the basis of *anatomy*, *benchmark*, *conference*, and *largebio* corpora for class matching. In addition, the average score across all tracks is given.

Feature	Information gain				
	anatomy	benchmark	conference	largebio	∅
#Examples	6958	3032	346	51844	
Substring length similarity	.1224	.3276	.3768	.1484	.2438
Shared token similarity	.1227	.1670	.2812	.1426	.1784
Equivalent shared token similarity	.1227	.1670	.2812	.1426	.1784
Neighborhood shared token similarity	.0957	.1504	.0713	.0810	.0996
Outdegree similarity	.0235	.0830	0	.0229	.0852
Token count similarity	.0749	.0234	.0857	.0251	.0523
Hierarchy level similarity	0	.0968	0	.0476	.0361
Property count similarity	0	.1395	0	0	.0349
Property shared token similarity	0	.0437	.0338	0	.0194

Table 1: Information Gain of Features for Class Matching

In its current shape, RSDLWB uses a Random Forest classifier [1] that was trained on the benchmark corpora. An inclusion of other classifiers trained on the other corpora is planned for future work. The tool suite WEKA [3] was used to create the classifiers.

2 Results

This section first describes the experimental set-up of the different OAEI tracks, in which RSDLWB has been evaluated. The evaluation results regarding RSDLWB are summarized in Tab. 2. The values for precision, F-measure, and recall were calculated with respect to the reference alignments specified in the second column. A detailed explanation of the reference alignments can be found in the respective paragraphs. The harmonic mean of all test cases is stated for *conference* and *multifarm*. Regarding *anatomy* and *largebio*, results for the single test cases are provided particularly.

benchmark The test cases of the *benchmark* track are systematically generated from two seed ontologies – biblio and IFC4 – by modifying or discarding several ontology features. Due to unverified technical difficulties during the execution performed by the organizers, RSDLWB did not produce any alignments for the *benchmark* track.

anatomy The task of the *anatomy* track is to match the Adult Mouse Anatomy and a part of the National Cancer Institute Thesaurus (NCI) describing the human anatomy. With regard to precision, F-measure, and recall, RSDLWB performs similar to the baseline algorithm StringEquiv. RSDLWB achieved high precision but low recall. Compared to the last year's evaluation [7], the quality of the produced alignments stayed approximatively the same. The runtime was improved from 1337 to 22 seconds.

conference This track consists of 16 heterogeneous ontologies in the domain of conference organization. There are three kinds of reference alignments for each test case: ra1, ra2, and rar2. The reference alignment ra2 is the transitive closure of ra1, in which conflicting correspondences had been eliminated by the organizers. The reference alignment rar2 is a refinement of ra2 in which violations had been removed by logical reasoning. Three evaluation modalities are provided for each reference alignments: M1 contains only classes, M2 only properties, and M3 is the union of M1 and M2.

RSDLWB showed its best accuracy regarding the M1 reference alignments. Regarding F-measure and ra1-M1, RSDLWB is better than the baseline algorithm StringEquiv. For ra2-M1 and rar2, RSDLWB is even better for the baseline algorithm edna regarding F-measure. The results for the M2 reference alignments show that RSDLWB matching of properties is improvable. This has also a negative effect on the results for the M3 reference alignments: Compared to the the OAEI 2014 campaign, RSDLWB's accuracy was significantly reduced in respect to ra2-M3. In particular, precision decreased by 0.53, recall increased by 0.02, and F-measure decreased by 0.24. The modalities M1 and M2 cannot be compared, because they were not available for the OAEI 2014 campaign.

multifarm The goal of the *multifarm* track is to evaluate the ability of a matcher to deal with ontologies in different languages. This track has two kinds of tasks: The first kind matches the same ontology in different languages (same) and the second matches different ontologies in different languages (diff).

RSDLWB does not support other languages than English yet. For *multifarm* RSDLWB uses the hierarchy level heuristic as described in Sect. 1.2. This heuristic works

Track	Reference Alignment	Runtime [h:m:s]	Precision	F-measure	Recall
anatomy	Mouse-NCI	00:00:22	.959	.732	.592
conference	H-Mean (ra1-M1)	n/a	.88	.66	.53
conference	H-Mean (ra1-M2)	n/a	.03	.05	.24
conference	H-Mean (ra1-M3)	n/a	.25	.33	.49
conference	H-Mean (ra2-M1)	n/a	.82	.61	.48
conference	H-Mean (ra2-M2)	n/a	.03	.05	.24
conference	H-Mean (ra2-M3)	n/a	.23	.3	.44
conference	H-Mean (rar2-M1)	n/a	.82	.63	.51
conference	H-Mean (rar2-M2)	n/a	.03	.05	.22
conference	H-Mean (rar2-M3)	n/a	.23	.31	.46
multifarm	H-Mean (diff)	00:00:14	.01	.01	.01
multifarm	H-Mean (same)	00:00:14	.20	.11	.08
largebio	FMA-NCI (small)	00:00:17	.964	.482	.321
largebio	FMA-NCI (whole)	00:03:31	.798	.443	.307
largebio	FMA-SNOMED (small)	00:00:36	.98	.226	.128
largebio	FMA-SNOMED (whole)	00:06:53	.933	.224	.127
largebio	SNOMED-NCI (small)	00:03:41	.967	.418	.267
largebio	SNOMED-NCI (whole)	00:07:16	.894	.408	.265

Table 2: RSDL Workbench Results for OAEI 2015

better for the tasks with same ontologies in different languages (same), because their hierarchies are identical. This is in contrast to the tasks with different ontologies (diff), where the ontologies have also different hierarchies. This explains the better quality of the produced alignments for the tasks with same ontologies in different languages.

largebio The data set of this track comprises the large biomedical ontologies Foundational Model of Anatomy (FMA), SNOMED CT, and NCI. These ontologies are semantically rich and contain a huge amount of concepts. *Largebio* consists of six test cases over three input ontologies. For each ontology pair, there are two tasks where whole ontologies (whole) or smaller fragments (small) are matched.

RSDLWB completed all the test cases in the given time frame of 10 hours. This is opposed to the OAEI 2014 campaign, when only the smaller FMA-NCI test could be completed [7]. In addition, the runtime was significantly improved. RSDLWB and LogMapLite [5] were the fastest systems altogether. Furthermore, RSDLWB and LogMapC [5] were the best systems in terms of precision across all test cases. In regard to the FMA-NCI test case, RSDLWB improved F-measure by 0.102.

2.1 Discussions on the way to improve the proposed system

As explained above, we plan to introduce features that exploit background knowledge in order to find non-trivial correspondences. It is also planned to use multilingual background knowledge from auxiliary ontologies like DBpedia to translate labels into different languages. This would enable support for the *multifarm* track.

Until now, the RSDLWB's classifiers were trained exclusively on the *benchmark* corpora. The integration of further classifiers that were trained on the other corpora might improve the results in regard to the different OAEI tracks.

The evaluation showed that RSDLWB's accuracy for class matching is much better than for property matching. One idea to improve the accuracy of property matching is to factor the similarity of their owning classes.

As explained in Sect. 1.2, the creation of a complete similarity matrix was replaced by heuristics to select seed pairs of concepts. Apparently, the fact that the matcher does not consider all concept pairs facilitates low recall. At the moment, the heuristic is too restrictive and only allows finding trivial correspondences with identical normalized labels. In the future, we want to explore further mapping candidates starting from the seed pairs.

3 Conclusion

The OAEI 2015 campaign showed a significant improvement of RSDLWB's runtime performance. This improvement was achieved by heuristics to select concepts pairs that are likely to match. The better runtime has enabled to complete all test cases of the *largebio* track, which is opposed to last year's OAEI campaign, when only one of six test cases could be completed. RSDLWB is one of the best systems in the OAEI 2015 campaign regarding the runtime. In general, RSDLWB has high precision when matching classes, but can be improved in regard to the matching of properties. In the future, we would to further improve RSDLWB's performance regarding recall.

Acknowledgments Special thanks to Henning Wachsmuth (Bauhaus-Universität Weimar) and Stefan Heindorf (University of Paderborn) for their support regarding machine learning.

References

1. Breiman, L.: Random Forests. Machine learning 45(1), 5–32 (2001)
2. Engels, G., Güldali, B., Soltenborn, C., Wehrheim, H.: Assuring Consistency of Business Process Models and Web Services Using Visual Contracts. In: Schürr, A., Nagl, M., Zündorf, A. (eds.) AGTIVE, LNCS, vol. 5088, pp. 17–31. Springer (2007)
3. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. SIGKDD Explor. Newsl. 11(1), 10–18 (Nov 2009)
4. Huma, Z., Gerth, C., Engels, G., Juwig, O.: Towards an Automatic Service Discovery for UML-based Rich Service Descriptions. In: France, R., Kazmeier, J., Breu, R., Atkinson, C. (eds.) MODELS. LNCS, vol. 7590, pp. 709–725. Springer (2012)
5. Jiménez-Ruiz, E., Grau, B.C.: LogMap: Logic-Based and Scalable Ontology Matching. In: 10th International Semantic Web Conference (ISWC). pp. 273–288 (2011)
6. Kullback, S., Leibler, R.A.: On Information and Sufficiency. The annals of mathematical statistics pp. 79–86 (1951)
7. Schwichtenberg, S., Gerth, C., Engels, G.: RSDL Workbench Results for 2014. In: Proceedings of the 9th International Workshop on Ontology Matching collocated with the 13th International Semantic Web Conference (ISWC). pp. 155–162 (2014)
8. Schwichtenberg, S., Gerth, C., Huma, Z., Engels, G.: Normalizing Heterogeneous Service Description Models with Generated QVT Transformations. In: Cabot, J., Rubin, J. (eds.) Modelling Foundations and Applications, LNCS, vol. 8569, pp. 180–195. Springer (2014)