TOWARDS STANDARD TEST COLLECTIONS FOR THE
EMPIRICAL EVALUATION OF SEMANTIC WEB SERVICE
APPROACHES

Ulrich Küster and Birgitta König-Ries
Institute for Informatics, Friedrich-Schiller-University Jena
07743 Jena, Germany
ukuester@informatik.uni-jena.de
koenig@informatik.uni-jena.de
http://fusion.cs.uni-jena.de

Received (Day Month Year)
Revised (Day Month Year)
Accepted (Day Month Year)

Semantic web services have received a significant amount of attention in the last years and many frameworks, algorithms and tools leveraging them have been proposed. Nevertheless surprisingly little effort has been put into the evaluation of the approaches so far. The main blocker of thorough evaluations is the lack of large and diverse test collections of semantic web services. In this paper we analyze requirements on such collections and shortcomings of the state of the art in this respect. Our contribution to overcoming those shortcomings is OPOSSum, a portal to support the community to build the necessary standard semantic web service test collections in a collaborative way. We discuss, how existing test collections have been integrated with OPOSSum, showcase the benefits of OPOSSum by an illustrative use case and outline next steps towards better standard test collections of semantic web services.

Keywords: semantic web services; evaluation; test collections.

1. Introduction

In recent years semantic services research has emerged as an application of the ideas of the semantic web to the service oriented computing paradigm. Semantic web services (SWS in the following) have received a significant amount of attention and research spending since their beginnings roughly six years ago [16]. We found that from 2002 to 2006 within the sixth EU framework program alone at least 20 projects with a combined funding of more than 70 million Euro dealt directly with semantic services which gives a good impression of the importance being currently put on this field of research. Research efforts so far have resulted in the proposal of numerous competing frameworks for SWS descriptions such as OWL-S [15],

\[http://cordis.europa.eu/fp6/projects.htm\]
WSMO [2], WSDL-S [1], or SAWSDL [3] and dozens of different matchmaking and discovery algorithms (see for instance [9] for an overview). Despite of this wealth of theoretical work surprisingly little effort has been spent towards the comparative evaluation of the competing approaches [14].

This corresponds to a trend that seems to exist in computer science in general. Tichy et. al. have found that computer scientists publish relatively few papers with experimentally validated results compared to other sciences [19]. Tichy argues that this trend is harmful for the progress of the science [18]. His arguments directly apply to the SWS domain. Without relying on the results of objective and well-founded evaluations, on what basis should researchers decide which SWS approach to enhance and base their following work on? Clearly, the gain achieved by the development of new alternative approaches is marginal, if it remains unclear whether or under which conditions the new approaches improve the old ones. Furthermore, without experiments that prove the applicability and relative advantage of SWS to real world problems, why should industry pick up a particular approach to SWS or the idea of SWS in general? There are striking parallels to this situation in the history of computer science:

"[in the experiments] . . . there have been two missing elements. First [...] there has been no concerted effort by groups to work with the same data, use the same evaluation techniques, and generally compare results across systems. The importance of this is not to show any system to be superior, but to allow comparison across a very wide variety of techniques, much wider than only one research group would tackle. [...] The second missing element, which has become critical [...] is the lack of a realistically-sized test collection. Evaluation using the small collections currently available may not reflect performance of systems in large [...] and certainly does not demonstrate any proven abilities of these systems to operate in real-world [...] environments. This is a major barrier to the transfer of these laboratory systems into the commercial world."

This quote by Donna Harman [6] addressed the situation in text retrieval research prior to the establishment of the series of TREC conferences in 1992 but seems to perfectly describe the current situation in SWS research. Harman continued:

"The overall goal of the Text REtrieval Conference (TREC) was to address these two missing elements. It is hoped that by providing a very large test collection, and encouraging interaction with other groups in a friendly evaluation forum, a new thrust in information retrieval will occur."

From the perspective of today, it is clear that her hope regarding the positive influence of the availability of mature evaluation methods to the progress of information retrieval research was well justified.

http://trec.nist.gov/
Towards Standard Test Collections for the Empirical Evaluation of SWS Approaches

We argue that a similar effort for SWS related research is necessary today for the advancement of this field. Similar to the situation in IR in the early nineties, the main problem in evaluating SWS technology is the need for realistic test data, i.e. semantically annotated web services. Studies by Klusch et al. [12] have shown that these are not readily available on the Web today. Therefore, there is a strong need to assemble suitable test collections. In this paper we investigate what the requirements towards these collections are, how they could be established, and present our contribution towards this end, OPOSSum.

The rest of this paper is organized as follows. In Section 2, we take a closer look at the requirements towards standard SWS test collections. We then (Section 3) discuss existing datasets and their shortcomings. In Section 4, we present the design and implementation of OPOSSum, a portal to support the building of better test collections. In Section 5, we describe how we leveraged existing work by integrating the most prominent existing collections into OPOSSum. We continue with brief reports on an experiment that illustrates the benefits of OPOSSum in Section 6 and ongoing work towards a new test collection in Section 7. Related work is presented in Section 8 before we conclude and summarize in Section 9.

2. Requirements for SWS Test Collections

As mentioned above, serious evaluation of semantic web services frameworks crucially depends on high quality test collections. In this section, we are going to investigate what requirements towards such collections exist and what the ideal test collection should look like.

2.1. Use Cases and Dimensions of Evaluation

First, it should be noted that the SWS use case at hand will imply important context assumptions and determine the main goal of the corresponding evaluation. The requirements towards the data required for evaluation will vary depending on which evaluation goal is pursued and which context assumptions are made. Typical evaluation goals are:

- For a given description language, compare the performance of different matchmaking algorithms with respect to precision, recall, scalability, ...
- For a given framework, test its usability, ease of learning, ...
- Compare different frameworks with respect to expressivity, matchmaking or composition capabilities, usability, ease of learning, ...

If one looks at the requirements implied by such cases more systematically, one can identify a number of dimensions that need to be spanned by any evaluation of semantic service frameworks and thus by the underlying test collections:

- The expressivity of the formalism employed influences how well the web service descriptions represent the service instances offered by a provider and
how well the needs of the client can be represented in a goal description. Similarly, the *correctness* measures the quality of the support offered by the framework, i.e. to which degree a framework acts precisely like the user it acts on behalf of.

- The *supported scope and automation* determines which of the different phases during service consumption are supported by a framework, which level of guarantee is attached to the results produced by a framework, and which level of automation can thus be supported for which tasks.
- The *usability* of the framework regards the ease to create semantic descriptions, the tool support and the compliance with standards. It also covers the aspect whether the modeling of services follows the intuition of a user, which is not always the case [5]. This dimension also includes the *effort* of creating and maintaining the framework, for example to agree on common ontologies or update existing ones.
- *Performance and scalability* regard the runtime properties of a system.
- Finally, the *level of decoupling* measures the openness of a system, i.e. how closely the client and the provider are linked in the overall process and to what extent and how easily the goal and web service descriptions can be created independently of each other, still yielding correct results.

Obviously, these dimensions are correlated, some of them negatively. If natural language is used as the description formalism, expressivity will be high, usability excellent, the effort minimal, and the level of decoupling very good. On the other hand, runtime performance will be poor and automation will nevertheless be impossible to achieve. If, on the other hand, a controlled vocabulary of business classification identifiers is used, the effort, usability, level of decoupling, performance, and scalability will be very good but the expressivity will be poor only supporting very few and simple use cases. SWS frameworks balance in this design space and evaluations should make the chosen tradeoffs and the resulting consequences explicit, transparent and comparable. This is only possible, if evaluations are not limited to single dimensions but cover as many of them as possible.

Furthermore, it is important to keep in mind that the results of the evaluation will depend on assumptions made about the usage context of a use case at hand. Thus, evaluations should be done in different such contexts, i.e. for different types of services, in different business settings (e.g., end-user vs. B2B), with different assumptions about the choreographies of the services (one step vs. complex), the involved transaction types, the required degree of automation (fully automated vs. only supporting the user), etc. This will make results reliable and universal by preventing them from depending on choices of context parameters.

### 2.2. Desirable Characteristics

The above considerations allow to derive a number of desirable characteristics of employed test collections from the evaluation dimensions above.
• Expressivity: In order to evaluate expressivity of a framework, natural language descriptions of services and goals need to be provided, which are then expressed using the different frameworks under evaluation. The provided descriptions need to be as unambiguous as possible, in order to leave as little room as possible for interpretation. Otherwise the results of the evaluation will be tampered by different such interpretations by different people. Since different approaches might be good in modeling different types of services, the collection needs to contain services from different domains and with different characteristics.

• Supported scope and automation: To illustrate and evaluate which approaches are advantageous in which settings requires a big but in particular diverse collection of scenarios and services covering as many as possible of the different envisioned use cases.

• Usability: Natural language scenario descriptions are required in order to evaluate the difficulty to semantically encode the involved services and goals within a particular framework. A set of pre-defined ontologies may be provided for some scenarios, while for others the necessary ontologies may need to be created from scratch.

• Performance and scalability: To compare the performance of different frameworks, they need to run against the same set of services. Thus, offer and goal descriptions in different languages for the same set of services and requests are needed to compare systems across formalisms. To effectively evaluate the scalability of approaches, a large testbed is needed. However, caution has to be paid with regard to automatically generated testbeds. Depending on how well these reflect the variety encountered in real-world settings, scalability measures may or may not reflect real-world circumstances accurately. The composition of a testbed, for example whether the contained services are very similar to each other or vary greatly, may have strong effects on the performance of certain approaches.

• Decoupling: The test collections need to reflect different people’s viewpoints and modeling approaches in order to effectively support testing for this aspect. Thus, preferably many people should contribute to their development. This will also prevent any unintended bias.

In summary, an ideal test collection needs to be fairly big, be composed of contributions by many different people, cover different domains and services with a variety of characteristics, and contain both, unambiguous, natural language descriptions of services and semantic descriptions for these services in different formalisms.

3. Publicly Available SWS Test Data

In this section we review the existing publicly available SWS test data with the requirements identified above in mind. We first cover SWS generally visible on the Web and then analyze existing SWS collections explicitly created for evaluation.
purposes.

3.1. Semantic Web Services Visible on the Web

The spread of publicly visible SWS accessible on the Web has recently been subject of a study by Klusch and Zhing. They investigated the question “Where are all the semantic Web services today?” They used a specialized crawler to search for Semantic Web Services and found less than seventy semantic service descriptions (38 OWL-S, 12 WSML, 11 WSDL-S, and 6 SAWSDL descriptions) in the surface Web as well as in the scientific archive citeseer” [12]. Even if one considers that their search did only find a minor share of the SWS descriptions available in total, it becomes clear that the number of SWS easily accessible on the Web is much too tiny to support any meaningful evaluation of SWS frameworks. Generally, as the authors remark, “the reported preliminary experimental result does not reflect the strong research efforts carried out in the SWS domain world wide in the past few years, independent from the status of maturity of SWS technology and implied low adoption by end users yet.” It seems very likely, that more SWS descriptions exists than are visible on the surface Web. Klusch and Zhing remark in this aspect: “Additional personal communication with few selected research groups at universities […] revealed that, if semantic Web service descriptions do exist at their site, the public retrieval from specific project related repositories is prohibited, hence invisible to any search engine.” It seems obvious that, given the amount of effort involved in creating semantic web services as test data, the level of reuse and sharing of such data among researchers needs to be significantly improved.

3.2. Explicitly Created Test Collections

Besides SWS descriptions generally scattered on the Web, there is also a number of comparably larger SWS collections that have been explicitly created for evaluation purposes and are publicly available. These are OWLS-TC 2.2\(^c\), SWS-TC 1.1\(^d\), SAWSDL-TC 1.1\(^e\), the Semantic Web Service Discovery Data Set by University Koblenz, Germany\(^f\), the Assam collection by Andreas Hess\(^g\), and the DIANE Benchmark\(^h\). Table 1 shows an overview of these collections.

OWLS-TC is the only collection frequently used and thus also regularly cited in the literature. OWLS-TC is also the base of two of the other collections (SAWSDL-TC and the Koblenz dataset). It can clearly be viewed as the current de facto standard of SWS test collections. Its popularity is due to its size and the fact that, unlike all other collections (except for SAWSDL-TC), it does not only contain

\(^{c}\)http://projects.semwebcentral.org/projects/owls-tc/
\(^{d}\)http://projects.semwebcentral.org/projects/sws-tc/
\(^{e}\)http://projects.semwebcentral.org/projects/sawsdl-tc/
\(^{f}\)https://www.uni-koblenz.de/FB4/Institutes/IFI/AGStaab/Projects/xmedia/dl-tree.htm
\(^{g}\)http://www.andreas-hess.info/projects/annotator/owl-ds.html
\(^{h}\)http://fusion.cs.uni-jena.de/DIANE/benchmark/
Formalism Size Released Comment
OWLS-TC OWL-S > 1000 2005 frequently used and updated
SWS-TC OWL-S 241 2006 not used recently
SAWSDL-TC SAWSDL/OWL 894 2008 based upon OWLS-TC
Koblenz DL (ACL/ALE-T) 96 2007 based upon OWLS-TC
Assam OWL-S 164 2004 not used recently
DIANE NL + DSD 195 2005 not used recently

Table 1. Overview of publicly available SWS test collections.

Advertisements, but also sample requests and a complete set of binary relevance judgments of the advertisements with respect to the requests. It can thus be readily used in particular to test and evaluate OWL-S matchmakers. This was also the purpose for which it has been developed [10].

SWS-TC and the Assam collection are alternative OWL-S datasets, but are much smaller, do not contain sample requests and relevance judgments, are not widely used and have not been updated recently. The Koblenz dataset shares the same restrictions. Furthermore, while services are described in description logics similar to the OWL-S collections the concrete modeling approach does not follow OWL-S conventions.

The DIANE Benchmark is the only collection that contains natural language request descriptions. However, the majority (145) of the 195 descriptions come from only two domains. They cover either requests to purchase a train ticket or a book. Thus, the collection is quite narrow in scope. Semantic descriptions for the services are only available in a non-standard formalism called DSD [8].

SAWSDL-TC has been released only recently to start development of a test collection for the new W3C standard SAWSDL. With respect to this standard, the current initial release has some limitations: there is only one single interface with a single operation per service description, model references point to OWL-DL concepts exclusively, and only automatically derived lifting schema mappings in XSLT are provided. Since SAWSDL-TC is a translation of OWLS-TC 2.2 from OWL-S to SAWSDL, SAWSDL-TC inherits the main features of OWLS-TC (in particular sample requests and relevance judgments).

Overall, the status with respect to available test collections may seem noncritical at the first glance, but a closer look reveals a number of issues.

- Despite of significant research efforts outside the OWL-S community (in particular on WSMO/WSML), there is very little publicly available test data for tools and algorithms relying on formalisms other than OWL-S.
- To the best of our knowledge only SAWSDL-TC and OWLS-TC are currently being actively developed.
- Each of the collections mentioned above has been created by a single group whereas a standard test collection should contain contributions by many different groups to be balanced with respect to the characteristics of the
contained services and to generally avoid any unintended bias towards certain approaches.

- All collections focus on a specific modeling approach in a specific formalism. While there are differences among the collections, services within one collection are generally quite similar to each other. None of the collections displays a large variety of service characteristics.
- The DIANE collection is the only one that contains natural language descriptions besides semantic service descriptions. Unfortunately, its scope is very narrow.
- Except for the translation of OWLS-TC to SAWSDL there is no collection with different descriptions in different formalisms for the same set of services.
- Even though most collections state that the contained descriptions were modeled after real services, only the Assam dataset provides information about these original services by including the original WSDL files. In case of the other collections information from the original services that was not captured by the contained description is not available anymore. It is not verifiable whether the service semantics are appropriately captured by the provided descriptions and whether the descriptions indeed reflect realistic services.

Obviously, none of the collections meets a significant subset of the requirements listed in Section 2 and is thus ready to be used for a broad and general comparative evaluation of different SWS approaches. Even though OWLS-TC (and the recently derived SAWSDL-TC) is more advanced than the other collections, it was not meant to serve as the standard test collection as it practically does: "Please note, that no standard test collection for OWL-S service retrieval does exist yet. As a consequence, OWLS-TC can only be considered as one possible starting point for any activity towards achieving such a standard collection by the community as a whole." (OWLS-TC 2.2 manual). Despite of clearly being the collection which comes closest to a standard test collection, it nevertheless contains a number of peculiarities and practical weaknesses which needs to be improved [14].

3.3. Conclusions

The previous section has shown that the requirements towards test collections are quite daunting. Our analysis above illustrates, that the possible candidates for a standard test collection are all far from meeting them yet. Furthermore, the number of SWS available on the web outside of dedicated test collections is clearly insufficient to form the basis of meaningful evaluations.

Building a collection that meets even only a significant subset of the listed requirements will require a lot of effort, to be more precise, it will require more effort that can be supplied by any single group. Even if there were a particularly resource-rich group, the need for impartiality would still make it undesirable to
have this one group develop the test collection alone. As a consequence, community involvement is crucial for successfully building high-quality test collections.

However, even though some of the existing collections have been publicly available for quite some time now, they have not been improved and updated other than by their original authors. This is particularly evident in the case of OWLS-TC. This collection has been downloaded from the SemWebCentral Portal more than 7000 times but according to the authors of OWLS-TC there have been only two occasions where external people also contributed to its further improvement. This illustrates that it is not easy to achieve community involvement. In our opinion, one prerequisite for obtaining the necessary contributions from the community is to offer appropriate tools that make contributing as easy and effortless as possible while offering a significant and obvious gain. Such tools have been largely lacking so far. In the following section, we will describe our contribution targeted at resolving this problem.

4. OPOSSum: Tool Support for Community Involvement

In order to provide the necessary tool support making the collaborative development of standard SWS test collections feasible, we have developed OPOSSum, the Online Portal for Semantic Services. In this section, we will motivate and explain the design of OPOSSum and provide information about its implementation.

4.1. Design Goals

According to the discussion in the previous sections, the design of OPOSSum has been motivated by three main objectives:

Goal 1: Promote exchange, reuse and collaborative improvement of existing data. As mentioned above, there must be many more SWS descriptions around than were found by the experiment by Klusch and Zhing. Despite of major projects in the field in Canada, Asia, or the pacific rim, for instance, Klusch and Zhing did not find any public semantic web services outside of the US, Europe and Iran. Apparently, most SWS descriptions developed within research projects remain hidden in private repositories. Similarly, three of the six test collections introduced in the previous section are available on project web sites only and thus hard to find.

Given the amount of effort involved in creating SWS descriptions, existing data has to be shared more efficiently. Besides this practical reason, using independent third-party data is also essential to avoid unintended biases and increase the objectivity and thus relevance of an evaluation. Therefore, sharing, reusing, and editing existing data must become easy.

Goal 2: Improve structure, documentation, and usability. The few public as well as private SWS collections that the authors know of are generally poorly docu-

---

1As of September 2008: http://projects.semwebcentral.org/top/toplist.php?type=downloads
2http://fusion.cs.uni-jena.de/OPOSSum
mented, poorly structured, and usually come in form of collections of flat files, which do not support convenient browsing nor powerful search. This limits their usability and keeps people from actually reusing them. Future collections must be improved in this aspect. To make this happen, this must be supported by tools.

**Goal 3: Support reuse and comparisons across formalisms.** Besides the effort to semantically annotate a given service, it is also far from trivial to come up with meaningful, rich, and diverse services in the first place (one of the obvious lessons learned from existing collections). Building a SWS test collection requires to gather (potentially fictitious) services and to semantically annotate them. Both steps are similarly challenging and time-consuming. Thus a collection of meaningful services in any description format including natural language is of great value when constructing a collection in a particular formalism. Furthermore, testbeds for different SWS description formalisms should not be isolated, as they currently are, but instead be closely interlinked. This supports the direct comparison of different description approaches for the same set of services, thereby allowing to investigate the trade-offs of the various approaches more easily.

### 4.2. Data Model

The internal data model of OPOSSum has been designed to support the three goals listed above. As a result of Goal 2, unlike existing file-centered collections, OPOSSum is built on top of a relational database. Figure 1 shows a slightly simplified picture of the data model of OPOSSum’s database.

![Fig. 1. Simplified data model of OPOSSum](image-url)
Unlike all existing collections, OPOSSum’s data is structured around the notion of a Service, independent from a particular service description. This promotes the reuse and comparison of service descriptions written in different formalisms (Goal 3). Accordingly, a service in OPOSSum is first described by a natural language text. Services can be classified in possibly overlapping Categories.

To add more structure and support more precise searching, a service’s Parameters (inputs and outputs) should be declared explicitly and described in natural language in addition to the general description of the service (Goal 2). To add more semantics without binding to a particular formalism, parameter types are mapped to WordNet synsets\(^k\), thus providing a kind of semantics that ensures an excellent compromise between being unambiguous, flexible, easily usable and language/formalism independent. We report on some experiences using WordNet in Section 5.

While some approaches to SWS model service offers and requests alike (e.g., OWL-S), others model them differently (e.g., WSMO). This enables to distinguish between more generic offers (like a flight booking service) and concrete requests (like a booking request for a particular flight). To accommodate both views, OPOSSum explicitly distinguishes between service Offers and Requests. OPOSSum’s data model allows to store different user-based Relevance judgments between an offer and a request and uses a multi-valued metric for those judgments, which better reflects the capabilities of current matchmaking approaches than the binary relevance judgments used so far in the state of the art [13].

Pointers to Service implementations (e.g., a web service) may be listed for an OPOSSum service.

Service descriptions written in any Formalism (WSDL, OWL-S, SAWSDL, WSML, ...) are collected by attaching them to the service (request or offer) that they describe. As mentioned above, this should ease the creation of descriptions in different formalisms and support the comparisons of different descriptions for the same service (Goal 3). Service descriptions may be grouped to Service collections.

Resources like ontologies or schemas can be added to the system as an independent entity. Descriptions may then refer to the resources that they import. This allows to compute the set of necessary resources for a given set of service descriptions and thus to automatically assemble test collections with all necessary resources on the fly.

To enable users to mark services, OPOSSum allows adding Tags to services as well as service descriptions in a Web 2.0 fashion. To support more powerful searching, those tags may optionally be linked to WordNet sense keys to disambiguate their semantics.

\(^k\)WordNet (http://wordnet.princeton.edu/) is a semantic lexicon for the English language developed at Princeton University. It uses the notion of synsets to collect synonyms and disambiguates homonyms. Sense keys are used to reference synsets, thus providing a unambiguous identifier of a particular semantic meaning.
4.3. Implementation

To enable easy sharing, reusing, and editing of existing data (Goal 1), OPOSSum has been implemented as a PHP-based web application online at http://fusion.cs.uni-jena.de/OPOSSum. This way – unlike with the existing collections – anybody willing can easily contribute to the collection hosted in OPOSSum in a wiki-like fashion. OPOSSum is fully functional. Users may search for services or descriptions based on their properties, compose and download collections based on search results, add new services and service descriptions to the database, store relevance judgments for offers and requests, improve the documentation of existing data, update descriptions, fix errors and inconsistencies in referenced ontologies, etc.

5. Integration of Existing Data with OPOSSum

To leverage existing work we completely imported OWLS-TC, SWS-TC, and SAWSDL-TC to OPOSSum. In this section, we will present some more information about the imported data and describe how it has been integrated with OPOSSum.

5.1. OWLS-TC

As mentioned previously, OWLS-TC is by far the largest and best established publicly available collection of semantic web services. The current version 2.2 of OWLS-TC contains 1003 services written in OWL-S 1.1, 69 of which are also provided in OWL-S 1.0. Additionally, it contains 29 OWL-S 1.1 service request descriptions, 28 of which are also provided in OWL-S 1.0. OWLS-TC is organized as a collection of files, each containing a single service description. The OWL-S 1.1 services are classified into seven domains (communication, economy, education, food, medical, travel, and weapon). Reference relevance judgments are provided through query folders that contain the set of advertisements judged binary relevant to the corresponding query by human judges. We parsed the service offer and query description files and extracted the following data from each of them:

- domain the service was listed under
- resources (ontologies) imported by each description
- service name as specified in the profile
- textual description as specified in the profile
- service inputs and outputs:
  - parameter name
  - parameter type (as an ontological reference)
  - parameter description as specified in the label attribute of a parameter (although next to none were provided).

Overall, the 1032 services used 3020 parameters with 417 different parameter types. These were manually mapped to WordNet sense keys as used in OPOSSum.
Complete information about the performed preprocessing and the resulting type mappings is available online\(^1\).

From the data obtained from each of the descriptions we inserted a corresponding service offer or request in OPOSSum and attached the original description for that service to the new service entry. The ontologies imported by those descriptions were inserted as well and linked to the descriptions that referenced them. The OWL-S 1.0 service descriptions additionally contained in OWLS-TC 2.2 were later added in the same way to the corresponding service entry. All generated service entries were categorized according to the domains under which they were listed in OWLS-TC 2.2. As a last step, the OWLS-TC relevance set folders were parsed and the explicit positive as well as the implicit negative relevance judgments were added to OPOSSum’s database.

OWLS-TC was thus completely integrated, but unfortunately the generated entries do contain errors, mostly related either to incorrect mappings of parameter types to WordNet sense keys or to flaws in the original OWLS-TC 2.2 data. We will illustrate this by some examples. OWLS-TC 2.2 contains numerous services that were apparently created by slightly varying existing services. This results in some copy and paste errors. The service "award_scholarshipduration_SwissGovservice.owls" for instance, has a service name "GermanGovernmentAwardScholarshipService" and is described as a service that provides information about scholarships by the German government. The service "price_Fishservice.owls" is described as a service that returns the price of dried fish offered by a company, but is named in the profile as "CanonCameraPriceService". Such trivial errors are aggravated by the fact that the services in OWLS-TC 2.2 are described with little detail. The above mentioned service delivering the price of dry fish, for instance, is described solely by its single output parameter of type price. A more complete analysis of such problems is available in [14]. While we created the WordNet type mappings, we encountered some parameter types which referred to non-existing concepts in the ontologies, mostly caused by confusing references to the SUMO and MILO ontologies\(^m\). We fixed those errors when we encountered them but we did not explicitly search for them. Most likely, the data still contains some erroneous references.

In addition to the flaws contained in the original OWLS-TC data, more errors were likely inserted during the process of mapping ontological parameter types to WordNet sense keys. These result from the fact that those mappings were made based on the referenced ontological types independent from the actual services. As a result they do not always describe the parameter types very well. First, the ontological types did not always have a perfectly matching entry in WordNet. Second, the ontological types themselves did not always capture the semantics of the parameter types precisely in the first place. Obviously, these two sources of errors can

\(^1\)http://hnsp.inf-bb.uni-jena.de/Opossum/index.php?action=faq
\(^m\)http://www.ontologyportal.org/
potentially combine in an unfortunate way. We report on some of the problems encountered when creating the WordNet mappings in Section 5.4. The automatically generated service entries were added to a specific category ("Incomplete - OWLS-TC") to mark that they could be considerably improved, if a human took some time to manually edit them. However, due to the size of OWLS-TC 2.2 this can only be done by the community as a whole. It is one of the achievements of OPOSSum, that this is now easily possible online.

OWLS-TC 2.2 contains roughly two dozen different domain ontologies that are referenced by the services. During the process of creating the type mappings, we realized that these contain concepts which are defined multiple times in the different ontologies without being explicitly related to each other. For some settings, this situation may be more realistic than one with a single unified ontology, thus, this is not a flaw of OWLS-TC, but it is a feature worth noting.

5.2. SWS-TC

SWS-TC 1.1 is another test collection of 241 OWL-S services. While SWS-TC 1.1 is much smaller than OWLS-TC, it seems that the average quality of the descriptions is somewhat better (e.g., the level of documentation) and that it contains fewer services created by only slight variations of existing ones.

The services from SWS-TC 1.1 have been parsed and added to OPOSSum exactly like those from OWLS-TC 2.2. The WordNet mappings created for the SWS-TC 1.1 service’s parameter types are available together with those created for OWLS-TC 2.2. Overall, the 241 services in SWS-TC 1.1 used a total of 594 parameters with 192 different parameter types, which illustrates the relatively higher variety of these services compared to OWLS-TC 2.2. Unlike OWLS-TC 2.2, SWS-TC 1.1 is based on a single unified domain ontology. Also, services in SWS-TC 1.1 are not classified in domains. Therefore, after their insertion to OPOSSum, a volunteer classified them manually in 12 overlapping categories (in contrast, the categorizations of the services from OWLS-TC 2.2 derived from the domain classification in that collection are disjoint). Neither sample requests nor relevance judgments are available for the SWS-TC services.

5.3. SAWSDL-TC

SAWSDL-TC is a recently released translation of OWLS-TC 2.2 from OWL-S to SAWSDL. The current first release contains 894 semantically annotated WSDL service offer descriptions and 26 semantically annotated WSDL request descriptions. Additionally, 1607 XSLT files with automatically generated lifting schema mappings are provided. Since SAWSDL-TC is a translation of OWLS-TC and the described services were thus already listed in OPOSSum, the descriptions from SAWSDL-TC could easily be integrated with OPOSSum by simply attaching them automatically to the corresponding existing service entry. OPOSSum allows arbitrary resources to be references by service descriptions. However, to keep the listed data as simple as
possible the 1607 XSLT files were not added as independent resources. Instead, the contained XSLTs were copied into the corresponding WSDL files and the SAWSDL liftingSchemaMapping references were changed to local references within the WSDL file.

Since most additional data like service classifications or reference judgments is handled in OPOSSum on the level of a service entry and not on the level of service descriptions, all of the corresponding information from SAWSDL-TC was already contained in OPOSSum and no further work was necessary to finish the import of SAWSDL-TC.

5.4. Mapping Parameter Types to WordNet Sense Keys

The main task to integrate OWLS-TC, SWS-TC, and SAWSDL-TD to OPOSSum consisted of mapping the service’s parameter types to WordNet sense keys. These mappings had to be created manually since this information was not contained in the original releases but constitutes an advancements compared to the original collections. When creating the mappings, we looked up the concepts in the ontologies, tried to estimate an English term which captured their semantics best and identified the corresponding WordNet synset using OPOSSum’s WordNet API. Although WordNet is likely the most complete and thorough ontology available, a couple of terms were difficult to map to WordNet synsets:

- Many compound terms are missing in WordNet, even though some of them are extremely common: email address, credit card number, account name, user name, dvd player, airport code, airline code, area code, arithmetic computation, full-time respectively part-time position, . . .
- Many technical terms are missing, too: SLR (single lens reflex camera), APS (advanced photo system), analog SLR, . . .
- A common feature in ontologies is a hierarchy of concepts where general concepts are specialized by adding restrictions. Such concepts generally did not map well to WordNet which often contains only the base concept. Examples are: cell phone with camera, recommended price, price in Euro, price in Dollar, tax free price, taxed price, . . .

In particular the absence of some very common compound terms (like email address or credit card number) is astonishing. It seems that Wordnet is surprisingly weak in this aspect. In some cases, however, it may also be, that we simply did not find the best matching term. The OPOSSum WordNet API performs a quite basic search in the WordNet dictionary, thus, it is often not trivial to come up with the right search term. The term “CryptographyKey” used in SWS-TC’s concepts ontology for instance, was mapped to the WordNet sense 106492320 (“a word that is used as a pattern to decode an encrypted message”) which is listed under the lemma

http://hnsp.inf-bb.uni-jena.de/opossum/index.php?action=wordnetAPI
"key word". A search for "key" lists 21 senses but does not list that particular one because the current API does not perform a substring search if precise matches are found. Also the search terms "cryptographic" or "cryptography" do not yield the above mentioned sense. We are currently working on improving OPOSSum’s Wordnet API to better support such cases.

5.5. Other Collections

While OWLS-TC, SWS-TC, and SAWSDL-TC have been completely integrated with OPOSSum, the other collections mentioned in Section 3.2 have not been integrated yet. However, we plan to do so in the future. The main difficulties are as follows.

The Koblenz database consists of a single RDF-serialized ontology which would have to be splitted in descriptions for the single services in order to add these to OPOSSum. Furthermore, it does not contain any natural language documentation regarding the semantics of the services. In order to add meaningful service entries to OPOSSum, those semantics would have to be derived manually from the description logic formalization of the services which is a very time-consuming task.

Somewhat the opposite is the case for the DIANE Benchmark. It consists primarily of natural language descriptions of the services, but formal descriptions are only available in a graphical format which can not be parsed easily. This makes it very hard to extract certain structure, like the inputs and outputs of the services. Again, these would have to be added completely manually.

The Assam collection appears to be severely outdated and many of the referenced ontologies are not available at the specified URLs. It requires further manual investigation whether the missing ontologies can be located elsewhere or replaced by new ontologies.

Finally, we would appreciate any advice regarding other collections that we are not yet aware of.

6. A Showcase of the Benefits of OPOSSum

In order to showcase the potential of OPOSSum we briefly report on an experiment that was performed with data hosted in OPOSSum and took advantage of functionality offered by OPOSSum.

The evaluation of the retrieval performance of matchmaking approaches (e.g., in the spirit of the S3 Contest on Semantic Service Selection\(^9\)) is probably the most prominent use case that requires large-scale SWS test collections. Test collections used for this purpose need a set of sample requests and reference relevance judgments identifying the set of services to be retrieved for each request. As mentioned, OWLS-TC 2.2 and the derived SAWSDL-TC are the only SWS test collection avail-

\(^9\)http://www-ags.dfki.uni-sb.de/~klusch/s3/index.html
able that include such, binary, relevance judgments. Using binary relevance judgments, an advertisement is considered to be either relevant or irrelevant to a request but no further distinction is made.

However, one of the core problems of SWS matchmaking is that it is unrealistic to expect advertisements and requests to be either a perfect match or a complete fail. Thus, virtually all SWS matchmakers support multiple degrees of match, i.e. they classify the set of advertisements into a hierarchy of different match levels or even assign a continuous degree of match to each offer. Nevertheless, existing approaches for the evaluation of the retrieval effectiveness of matchmaking approaches have so far been based exclusively on binary relevance. This remarkable discrepancy may distort evaluation results and compromise their reliability.

Therefore, OPOSSum was designed to support multi-valued and user-based relevance judgments. This means a configurable hierarchy of graded match levels is available and different judges do not necessarily have to agree how to judge a specific query-advertisement pair. Furthermore, OPOSSum provides a powerful Web interface that allows human assessors to comfortably judge a large number of services, thus making the creation and management of large numbers of multi-valued relevance judgments feasible.

This allowed to conduct a preliminary experiment that investigated the feasibility of using multi-valued relevance judgments and the properties of different retrieval performance measures based on such multi-valued relevance. Details on the experiment are available in [13].

To prepare the experiment, three human assessors judged the education domain subset of OWLS-TC. This subset contains six requests and 276 advertisements. Nearly 5000 multi-valued relevance statements were created by the three assessors in addition to the existing 1656 binary judgments that had been imported from OWLS-TC. Ranked service output lists of different matchmakers for the given six queries were provided by the organizers of the S3 Contest. A Java-based evaluation framework was created that processes those output lists and computes different binary and graded recall and precision measures. The necessary relevance judgments were directly retrieved from OPOSSum’s relational database by using the publicly available programmatic access. This allowed to easily investigate the effects of switching from binary to graded relevance measures or using different strategies to deal with conflicting judgments by different assessors. It was found that using multi-valued relevance in SWS retrieval evaluation is feasible and that evaluation results are sensitive towards the choice of employed relevance levels and retrieval measures [13].

We would like to stress that this experiment would not have been feasible without the support offered by OPOSSum which made it easy to create and manage the necessary large number of relevance judgments. Furthermore, other groups have now direct access to the test data used and created within this experiment. Thus, other researchers can use the data to confirm or revise the reported results. Or they may use the data as a base to create additional data, as the OWLS-TC data was
used as the base for the data created for the reported experiment.

Overall, it is expected that the community will immediately benefit from the support offered by OPOSSum and it is hoped that, over time, an increasingly large and versatile collection will evolve in OPOSSum.

7. Directions for Future Work on Standard SWS Test Collections

OPOSSum already contains over 2400 descriptions for nearly 1500 services. However, the largest share of this data was imported from OWLS-TC, SWS-TC and SAWSDL-TC. As discussed in Section 3.2, this data is far from meeting all requirements for a standard SWS test collection. In particular it has been criticized to not represent realistic services appropriately [14]. In this section we report on initial efforts to create additional data which comes closer to meeting the requirements identified in Section 2.

One of the difficulties when building a test collection of SWS is to ensure for a sufficient variety of services and, at the same time, make sure that the contained services are still somewhat related and similar. For many evaluation purposes it is particularly important to have services which are similar to each other in varying degrees, i.e. services which are very similar to each other, others, that are completely different, and as many in between as possible.

We therefore examined public web service repositories to identify a domain with many public services that meet this requirement. For the start we chose the geocoding/mapping/geographic-information domain since it appears to be the domain with the most publicly accessible web services available. We analyzed and collected services within this domain from sources like http://seekda.com, http://xmethods.com, http://webservicelist.com/, http://www.programmableweb.com/, or http://www.geonames.org/. So far we have already added 170 such services to OPOSSum. All of these services are real, working web services, a significant number of them commercial. Most are implemented as SOAP web services but a large number is also REST-based. The services were added to OPOSSum with complete information, including the WSDL (where available), pointers to the implementation of the services and natural language documentation of what the service delivers. Work on further enlarging the dataset and defining sample requests and relevance judgments is still ongoing. While the collection is still comparatively small, to the best of our knowledge it is already the largest collection with real, handpicked services from a single domain. We also found, that the characteristics of these real services differ significantly from the ones of services found in current test collections. The average number of service input parameters, for instance, is roughly twice as high as in the case of OWLS-TC or SWS-TC (2.6 versus 1.3 and 1.4). This may be due to the chosen domain, but it illustrates the need for versatile data.

In order to turn the new dataset into a test collection that meets the requirements listed in Section 2, semantic descriptions in different formalisms are needed
Towards Standard Test Collections for the Empirical Evaluation of SWS Approaches

19

for the services. To avoid an unintended bias, to make sure that the descriptions reflect different modeling styles, and to generally increase the objectivity of the collection, these descriptions should ideally be contributed by different groups.

We are currently attempting to initiate such efforts in collaboration with the organizers of the two main evaluation initiatives in the field, i.e. the SWS-Challenge and the S3 Contest (see Section 8). It is hoped that this will contribute to the creation of better and more realistic test collections than are currently available.

8. Related Work

Relatively few work on evaluation of SWS is available. A number of publications report the evaluation of SWS matchmakers with the existing collections, for example [4, 7, 11]. However, they do not discuss the properties of these collections or how to improve them. While some of the collections have been available for years now, there was nearly no scientific discussion about them so far. In previous work we provided a discussion of the properties of OWLS-TC, the most prominent existing SWS test collection [14].

The work most closely related to the OPOSSum portal presented in this paper is SWS-TC-Wiki. SWS-TC-Wiki is a Wiki dedicated to foster community participation in improving existing test collections (originally OWLS-TC). It has been set up by the authors of OWLS-TC and organizers of the S3 Contest. This contest aims at establishing an evaluation initiative of SWS matchmakers that is modeled after the successful series of TREC conferences in information retrieval. Participating matchmakers are evaluated based on the classic retrieval performance metrics recall and precision, additionally, their runtime performance is measured. The initial 2007 edition of the contest was based on OWLS-TC and thus limited to OWL-S matchmakers. The 2008 edition will use the newly created SAWSDL-TC to extend the scope to SAWSDL matchmakers, too. Further extensions of the scope are planned but depend on the availability of corresponding test collections.

SWS-TC-Wiki has raised the awareness of this need, however, SWS-TC-Wiki is a fairly lightweight approach. While OPOSSum is based on a relational database, SWS-TC-Wiki is designed as a collection of files. Therefore, SWS-TC-Wiki is agnostic to the contents or structure of service descriptions and does not provide more structuring than a file system would. As a consequence, it enables easy sharing of semantic descriptions but does not allow to edit existing descriptions easily. It also does not offer elaborate search capabilities, the means to manage user-based relevance judgments or programmatic access to the data. OPOSSum was designed as an improvement of SWS-TC-Wiki in these aspects.

The Semantic Web Service Challenge is the best established initiative in the area of SWS evaluation. The Challenge provides a certification of the capabilities of

\[\text{http://www-ags.dfki.uni-sb.de/swstc-wiki/} \]
\[\text{http://www-ags.dfki.uni-sb.de/~klusch/s3/} \]
\[\text{http://sws-challenge.org} \]
SWS approaches via a set of elaborate problem scenarios. It currently consists of two tracks, one focusing on data and process mediation aspects, the other one targeted at service discovery. The Challenge evaluates primarily two aspects. It provides a certification of the functional scope of a framework by assessing that certain problem scenarios can be solved with the framework. Additionally, it attempts to provide an indication about the flexibility of the approaches by testing how difficult it is to adapt a given solution to a change in the underlying problem scenario. The Challenge methodology is described in detail in a W3C expert group report [17].

Quantitative measurements that require large test collections as performed by the S3 Contest have so far been beyond the scope of the Challenge.

Apart of the SWS-Challenge and the S3 Contest there is next to no work dedicated specifically to discuss issues of SWS evaluation. However, a comprehensive survey of more remotely related work is available in [14].

9. Summary

Doing science means producing reproducible results that can be independently evaluated. This is an indispensable requirement for scientific progress and industrial adoption of research results. This article argued that more efforts regarding the evaluation of SWS technology are necessary for the further advancement of the field. A mandatory prerequisite for such activities is the establishment of large and high quality SWS test collections in various formalisms. Despite of important initial efforts, such test collections are still largely lacking.

In this article, requirements on SWS test collections were investigated and the state of the art examined. It was found that there is too little test data, that the few existing collections are far from meeting all requirements on good test collections and that more community involvement is essential in order to build the desired better collections. The main prerequisite to obtain the necessary contributions from the community is to offer appropriate tools that make contributing as easy and effortless as possible and allow the community to benefit from such contributions immediately. Such tools had been lacking so far.

A solution to this problem was presented: OPOSSum, a web portal that supports the community effort of building up high-quality SWS test collections. To get started and leverage previous efforts, the major existing public collections have been fully integrated with OPOSSum. OPOSSum’s web interface enables the community to collaboratively update, correct, and improve this data. Additionally, everybody can share his work and extend OPOSSum’s dataset by adding new services and service descriptions online with as little effort as possible. The community benefits from a more efficient use of existing resources by having access to a large, public repository of semantic web services, that is well structured and can be much more efficiently searched and browsed than previous collections.

These benefits were illustrated by means of an experiment on evaluating SWS matchmaking performance with graded instead of binary relevance judgments. This
experiment relied on data and functionality provided by OPOSSum and would not have been feasible without such tool support.

Finally, preliminary results from efforts to build a new higher quality dataset were presented. The next steps towards an initial collection that meets the requirements on a standard SWS test collection were outlined. However, this work will not be feasible without community participation and the creation of true standard test collections clearly needs to be a continuous and collaborative process. The community is invited to join corresponding efforts and it is hoped that the work presented in this article will help facilitating them.

Acknowledgements

We would like to thank Andreas Krug and Christian Ruhmer who implemented essential parts of OPOSSum and helped with providing graded relevance judgments for the OWLS-TC Education subset. We would also like to thank the authors of the existing test collections for sharing them and making them publicly available. In particular we are thankful to Patrick Kapahnke and Matthias Klusch for their help regarding the integration of OWLS-TC and SAWSDL-TC in OPOSSum.

References


