Ulrich Lampe, Stefan Schulte, Melanie Siebenhaar, Dieter Schuller, Ralf Steinmetz:
Adaptive Matchmaking for RESTful Services based on hRESTS and MicroWSMO.
In: Walter Binder and Heiko Schuldt: Proceedings of the 5th Workshop on Enhanced Web Service Technologies (WEWST 2010), p. 10-17, The Association for Computing Machinery (ACM), December 2010.

Adaptive Matchmaking for RESTful Services based on hRESTS and MicroWSMO

Ulrich Lampe
ulrich.lampe@kom.tu-darmstadt.de

Stefan Schulte
stefan.schulte@kom.tu-darmstadt.de

Melanie Siebenhaar
melanie.siebenhaar@kom.tu-darmstadt.de

Dieter Schuller
dieter.schuller@kom.tu-darmstadt.de

Ralf Steinmetz
ralf.steinmetz@kom.tu-darmstadt.de

Multimedia Communications Lab (KOM)
Technische Universität Darmstadt
Rundeturmstr. 10, 64283 Darmstadt, Germany

ABSTRACT

Matchmaking – i.e., the task of finding functionally suitable service offers based on a service request – has only been addressed in the context of WS-* Web services. However, RESTful services are gaining increasing attraction and have been adopted by major companies, thus increasing the need for suitable matchmaking solutions. This paper introduces XAM4SWS, an adaptive matchmaker for semantic Web services that supports multiple service description formats, including hRESTS and MicroWSMO for RESTful services. XAM4SWS adapts existing methodologies from WS-* matchmaking and extends them through the inclusion of REST-specific service features. A prototypical implementation of the matchmaker is evaluated with respect to multiple information retrieval metrics using an adapted semantic Web service test collection.

1. INTRODUCTION

In the past few years, the paradigm of Service-oriented Architecture (SOA) has received significant attention from both practitioners and researchers. At the core of SOA are services that represent certain (business) functionalities and are exhibited through a well-defined interface. Key characteristics of services include interoperability, composability, and loose coupling. That is, fine-granular services can be combined into more powerful, coarse-granular composite services, and individual services in such composition can be substituted through functionally equivalent counterparts [1]. In the envisioned Internet of Services (IoS), service consumers will be able to purchase services from providers by the means of public marketplaces in order to realize certain IT functionality. Such a scenario requires effective and efficient *matchmaking*, i.e., the ability to identify suitable service offers based on functional requirements, as defined in a service request.

Today, *Web services* have become the de facto implementation of the SOA paradigm. Web services are based on a stack of so-called *WS-** standards, most importantly the SOAP communication protocol and the *Web Services Description Language* (WSDL). Matchmaking for WS-* Web services has been a prominent field of research recently [2–6]. It is mostly based on the notion of semantic Web services (SWS), i.e., syntactic service descriptions that have been augmented with machine-processable semantic information [7, 8]. Meanwhile, in many application scenarios, Web services are perceived as too complex and heavyweight. As a result, so-called RESTful Web services have gained increasing interest. RESTful Web services constitute a lightweight alternative to SOAP-based Web services. They are based on standard Web technologies, such as Hypertext Transfer Protocol (HTTP), Uniform Resource Identifier (URI), and Extensible Markup Language (XML) [9]. RESTful services have been adopted by major companies, such as *Google*, *Facebook*, or *Flickr*. However, the problem of matchmaking has not been addressed for RESTful services so far, notably due to a lack of common description standards.

In the work at hand, we present *XAM4SWS* ("Cross-Architectural Matchmaker for Semantic Web Services"), an adaptive matchmaker that supports semantic matchmaking for both SOAP-based and RESTful Web services. It extends LOG4SWS.KOM, a matchmaking approach we presented in [6], to support the HTML for RESTful Services (hRESTS) description format in conjunction with MicroWSMO semantic annotations [10, 11]. XAM4SWS uses a generic foundation which is complemented by specific components for each supported service description format.

The remainder of this paper is structured as follows: In Section 2, we briefly introduce description formats for Web services. A focus lies on the hRESTS and MicroWSMO formats for RESTful services. This provides the background for Section 3 which discusses related work. In Section 4, the matchmaking algorithm in XAM4SWS is introduced in detail. In Section 5, we present the evaluation results for XAM4SWS. Section 6 concludes the paper and provides a brief outlook on future research.

.

2. (SEMANTIC) SERVICE DESCRIPTION

The issue of (machine-readable) description of SOAP-based services has been a field of research for almost a decade now. The most popular result is WSDL, a World Wide Web Consortium (W3C) specification; its latest version is WSDL 2.0 [12]. WSDL allows to syntactically describe a Web service. Both standards have been augmented by the *Semantic Annotations for WSDL and XML Schema* (SAWSDL) recommendation [13]. SAWSDL constitutes a lightweight framework to add semantic annotations to syntactical WSDL files. Matchmaking for SOAP-based Web services is often based on SAWSDL documents [2].

With respect to RESTful services, there exists no commonly accepted description standard yet [9]. RESTful Web services are often described using non-normative, human-readable documents, e.g., in the form of Web pages [11][1]. Fairly recently, however, two machine-readable service description formats have been proposed. *Web Applications Description Language* (WADL) constitutes a full-fledged approach that adapts some of the ideas of WSDL, most notably the utilization of XML Schema (XSD) for the description of input and output messages [14]. However, WADL only provides syntactic service descriptions and does not define any semantic annotation mechanism to date (an exception is the ability to leverage SAWSDL for the annotation of XSD definitions within a WADL document).

In contrast to WADL, hRESTS provides a lightweight mechanism to augment existing service descriptions. Service descriptions in hRESTS are based on textual descriptions of RESTful Web services in the form of Hypertext Markup Language (HTML) Web pages. hRESTS assumes a minimal service model, which essentially defines a service as a set of operations that in turn relate to a distinct set of input and output parameters [10]. For each of these service components and their related attributes, hRESTS defines classes. These classes can be used within an (existing) HTML page, thus marking certain elements in that page as specific service component types. Thus, hRESTS provides a machine-readable service description within a human-readable document. Due to the fact that a multitude of plain HTML service descriptions already exists on the Web, we believe that hRESTS currently constitutes a more practical approach than WADL. Thus, the matchmaking approach presented in this paper focuses on hRESTS.

As such, hRESTS documents only carry syntactic information. In order to include semantic information, the use of the MicroWSMO microformat has been proposed [10,11]. In accordance with the SAWSDL specification, MicroWSMO introduces three types of semantic annotations. Most importantly, the *model* construct adds a link to one or more arbitrary semantic concepts, which semantically describe a service component. The *lifting* and *lowering* constructs define how to transform data. For an overview of the hRESTS component model, we refer to Kopecký et al. [10].

In order to facilitate the automated processing of semantic information, we assume that all referenced concepts are represented in the Web Ontology Language (OWL); more specifically, OWL's description logic (DL) variant. This is a common assumption in the context of matchmaking.

[1]E.g., Flickr's *App Garden* `http://www.flickr.com/services/api/`

3. RELATED WORK

Matchmaking for semantic Web services (SWS) has been a dynamic field of research in recent years, resulting in the creation of numerous approaches for different Web service description standards. The wide range of participants in the annual *International Semantic Service Selection (S3) Contest* resemble this variety [2]. However, to the best of our knowledge, no specific matchmaking approach for hRESTS has been proposed so far. XAM4SWS builds on a matchmaking approach we presented previously, LOG4SWS.KOM, which operates on service descriptions in the SAWSDL format.

LOG4SWS.KOM [6] is based on "traditional" subsumption matching, as introduced by Paolucci et al. [8]. However, it maps the discrete Degrees of Match (DoM) into numerical equivalents using an *Ordinary Least Squares* (OLS) estimator to allow for a combination with additional numerical similarity measures. As proof-of-concept, the inverse path length between two classes in an ontology is utilized. The approach is complemented by a WordNet-based fallback strategy.

The general applicability of adaptation mechanisms to SWS discovery has been demonstrated by Kiefer and Bernstein [15]. Their approach, iSPARQL, implements a query mechanism for OWL-S based service descriptions based on the SPARQL Query Language for RDF (SPARQL). Kiefer and Bernstein apply both string-based and vector-based strategies for the determination of similarities. The optimal aggregation of these similarity measures over a whole service is consecutively determined using various machine learning techniques.

Different SWS description formats have been addressed by the "MX" family of matchmakers by Klusch et al. [3,4]. *OWLS-MX* is a hybrid matchmaker for OWL-S based service descriptions. It combines logic-based matching with additional similarity metrics from the field of text information retrieval (IR). *SAWSDL-MX*, which operates on SAWSDL descriptions, follows a comparable approach to OWLS-MX. In its latest version, SAWSDL-MX additionally features a component for structural analysis of WSDL documents and applies machine learning techniques for an optimal aggregation of the applied similarity measures.

Battle and Benson [16] introduce the *Semantic Bridge for Web Services* (SWBS). It provides means to access services and their underlying data in a unified manner using an extended version of SPARQL. For that matter, SWBS wraps existing Web services based on their WSDL or WADL description files. SWBS automatically distributes single queries across multiple endpoints and thus offers rudimentary service discovery and composition features. However, its matchmaking capabilities are limited to semantic information.

In summary, XAM4SWS adapts many ideas from our previously presented matchmaker LOG4SWS.KOM. Most notably, this concerns the mapping of discrete DoMs into numerical equivalents using OLS. In contrast to other matchmaking approaches, XAM4SWS uses syntactical information only as a *substitute*, not as a *complement* to semantic information. Our matchmaker further leverages the special features of hRESTS service descriptions, e.g., the explicit definition of operation types. By supporting multiple service description formats, including (SA)WSDL and hRESTS, XAM4SWS permits the discovery of services across architectural boundaries.

4. MATCHMAKING ALGORITHM

4.1 Operations-focused Matching

The matchmaker we previously presented for SAWSDL, LOG4SWS.KOM, applies the notion of *operations-focused matching* [6]. This approach is based on the observation that operations provide the essential functionality a service requester is looking for. Thus, similarity values from all levels of abstractions in a service are aggregated at the operation level. Subsequently, pairs of operations are matched, regardless of how these operations are organized into interfaces. In XAM4SWS, the same notion is applied. In fact, the approach is even more intuitive, because hRESTS does not provide an equivalent to SAWSDL's interface component. As a substitute, we utilize the information available from the topmost level of abstraction in hRESTS documents, namely the service level.

An overview of the matchmaking process is provided in Figure 1. For each pair of operations in service request and offer, XAM4SWS determines the similarity for the respective input (sim_{in}), output (sim_{out}), native operation (sim_{op}), and service (sim_{ser}) levels. These individual similarities are combined using the weights w_{in}, w_{out}, w_{op}, and w_{ser}, resulting in an aggregated similarity value sim_{agg} for each pair of operations. Formally expressed, with both a and b denoting an operation, the following holds:

$$w_{ser} + w_{op} + w_{in} + w_{out} = 1 \tag{1}$$

$$\begin{aligned} sim_{agg}(a,b) = \quad & sim_{ser}(a,b) * w_{ser} \\ & + sim_{op}(a,b) * w_{op} \\ & + sim_{in}(a,b) * w_{in} \\ & + sim_{out}(a,b) * w_{out} \end{aligned} \tag{2}$$

Based on these aggregated similarities, the overall service similarity sim_{ov} is given by computing an optimal matching of operations. Formally, let A and B be the sets of operations in the service request R and service offer O respectively. Let x_{ab} be a binary variable, indicating whether $a \in A$ has been matched with $b \in B$. The overall service similarity is computed as follows:

$$sim_{ov}(R,O) = \frac{1}{|A|} * \sum_{a \in A, b \in B} x_{ab} * sim_{agg}(a,b) \tag{3}$$

The matching process for sets of components (i.e., operations, inputs, and outputs) is conducted using bipartite graphs, as initially suggested by Guo et al. [17] and further elaborated by Bellur et al. [18]. The sets of components from service request and offer each constitute a partition of nodes in the graph. Each node in the first partition is connected to each node in the second partition through a weighted edge. The edge weight corresponds to the similarity between the two components. Using the well-known Kuhn-Munkres algorithm [19], optimal 1-to-1 assignments of nodes are determined. Because cardinalities may differ between the two partitions, we apply an adapted version of the algorithm by Nedas that may potentially leave some operations unmatched [20].

Following the matching process, the edge weights of all matched edges are summed up and divided by the cardinality of the original component sets. If these cardinalities differ, the following strategy is employed: In general, the cardinality of the service request's component set is decisive. In the case of inputs, however, the cardinality of the service offer's component set is decisive. Thus, if an offer lacks requested operations or outputs, its overall similarity diminishes. The same is true if an offer requires more inputs than the request provides. This procedure does not exclude any service offers because of a mismatch in the number of operation or parameters a priori. Instead, these offers are implicitly punished by a reduction in similarity. As tie-breaker for sorting service offers with identical overall similarity, we use the alphabetical order of their associated hRESTS description URIs.

4.2 Assignment of Similarities

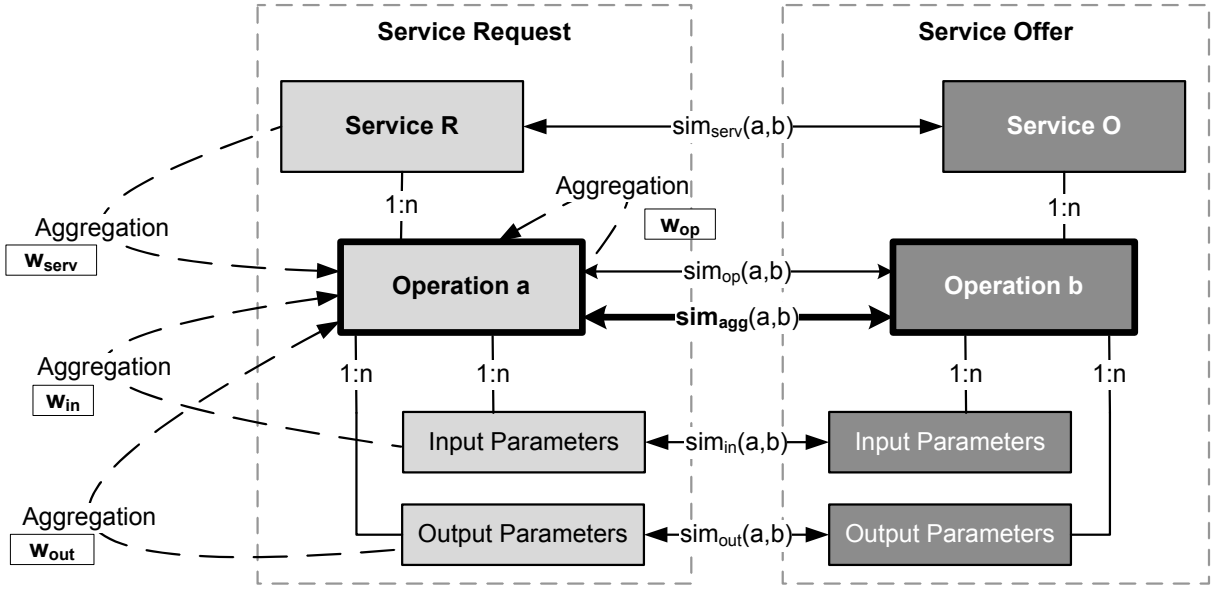


Figure 1: Matchmaking Process

The shortcomings of "traditional" subsumption reasoning in the context of matchmaking have been extensively discussed in our previous work on LOG4SWS.KOM [6]. In the same work, we presented and successfully evaluated a novel approach which maps discrete DoMs onto numerical equivalents. The same procedure is applied in the work at hand. We will thus limit our discussion to the most fundamental aspects and refer the interested reader to our previous work for additional details.

Essentially, subsumption matching based on discrete DoMs, as suggested by Paolucci et al. [8], and applied in similar form in related work [3, 4], has three significant disadvantages. First, traditional DoMs are discrete in nature and thus only permit a coarse-grained ranking of services. Second, discrete DoMs are non-numerical, which complicates the combination with complimentary continuous numerical measures. Third, the ranking of DoMs is based on certain assumptions that do not necessarily hold true in every application domain.

Our approach specifies four generic types of matches between semantic concepts. These DoMs can be applied to any service abstraction level and type of parameter. Given two arbitrary semantic concepts, A (from the service request) and B (from the service offer), the DoM is defined as

$$DoM(A,B) = \begin{cases} exact & if A \equiv B \\ super & if A \sqsubseteq B \\ sub & if A \sqsupseteq B \\ fail & else \end{cases} \quad (4)$$

We map these four discrete DoMs onto a continuous numerical scale, ranging from 0 (no similarity at all) to 1 (perfect similarity). This approach allows the combination with other numerical measures and a much more fine-grained ranking of services. On each individual matching level $L \in \{iface, op, in, out\}$, each DoM $D \in \{exact, super, sub, fail\}$ is assigned a numerical equivalent, $d_{L.D}$, in the range $[0;1]$. Formally,

$$d_{L.D} \in [0;1] \quad \forall \quad L \in \{iface, op, in, out\}, \; D \in \{exact, super, sub, fail\} \quad (5)$$

To allow for a more fine-grained similarity assessment, in the case of a super or sub match, the DoM's numerical equivalent is merged with the path length between two concepts. One intuitive approach to conduct such merging is simply dividing the numerical equivalent by the path length, based on the assumption that the similarity between two concepts (linearly) shrinks with their distance in an ontology.

Formally, let $PL(A,B)$ denote the shortest path between the two concepts A and B in an ontology. Furthermore, let L be the level on which the matching of components that point to these concepts is conducted. Then, the similarity $cs(A,B)$ between A and B (and thus, the two underlying components) is given by:

$$cs(A,B) = \begin{cases} d_{L.exact} & if A \equiv B \\ d_{L.super}/PL(A,B) & if A \sqsubseteq B \\ d_{L.sub}/PL(A,B) & if A \sqsupseteq B \\ d_{L.fail} & else \end{cases} \quad (6)$$

The task of defining numerical equivalents for DoMs is ambigious and would require significant manual tuning effort. To address this problem, we apply an OLS estimator for the determination of optimal numerical DoM equivalents.

The process is based on the notion that a dependent variable $y_L^{a.b}$ – corresponding to the similarity of two operations, a and b, on a certain matching level, L, – can be derived through the linear combination of a set of independent variables $x_{L.D}^{a.b}$, corresponding to the frequency of a certain DoM D when matching a and b on that level. That is, we assume that the weighted linear combination of the different DoM's frequencies predicts the similarity of two operations.

The OLS estimation is independently conducted for each matching level. That is, the numerical weights differ for inputs, outputs, operations, and services. As training data, a set of services is required along with a predefined similarity (or relevance) rating. A subset of a test collection, such as employed in the evaluation of XAM4SWS (cp. Section 5.1), fulfills this requirement.

In the training phase, XAM4SWS computes similarities between all pairs of operations in all service requests and offers. For each pair and each abstraction level, it stores the types of subsumption matches in the assigned components along with the path length. XAM4SWS then determines the predefined similarity between the two operations (or, if the similarity is unavailable at the operations level, of the parent services). The predefined similarity rating constitutes the vectors of predictors (y_{ser}, y_{op}, y_{in}, and y_{out}) for the OLS process. Each entry corresponds to a pair of operations.

The design matrices (X_{ser}, X_{op}, X_{in}, and X_{out}) are derived in the following manner: Each pair of operations yields one row with four entries, where each entry corresponds to the frequency of a certain type of DoM with respect to all matched components on a certain level. In detail, the frequency count is incremented by 1 for an exact and fail match between two components. For super and sub matches, it is incremented by 1 divided by the path length. The row is finally divided by the total number of matched components on the current level.

Given a design matrix and vector of predictors, the standard OLS estimator can be applied in the following manner on each matching level L:

$$\hat{\beta}_L = (X'_L X_L)^{-1} X'_L y_L \quad (7)$$

$\hat{\beta}_L$ corresponds to the optimal estimate of numerical weights. To derive the actually utilized vectors of weights, d_L, all entries are mapped to the range $[0;1]$. For that matter, the minimum value in the vector is added to all entries. Then, all entries are divided by the new maximum value. This ensures that all similarity values will also be in the specified range, i.e., $[0;1]$.

4.3 Consideration of REST-specific Features

In contrast to WSDL, OWL-S, or WADL, the hRESTS format provides a rather rudimentary service description. This makes it difficult to identify specific features which can be exploited in the context of matchmaking.

However, in hRESTS, each invocation of an operation is associated with a command in the underlying HTTP transfer, the so-called *HTTP verb*. These HTTP verbs are *POST*, *GET*, *PUT*, and *DELETE*. With respect to RESTful services, the verbs can be mapped to the four elementary operations of data storage or retrieval systems, namely *Create*, *Read*, *Update*, and *Delete* (the so-called CRUD operations) [16]. Thus, in theory, operations that exhibit a differ-

ent HTTP verb for invocation can be deemed incompatible, resulting in a similarity value of 0. Unfortunately, the mapping between HTTP verbs and CRUD operations is non-normative, and its strict application can or will not always be enforced by providers of RESTful services. A common exception in practice is, e.g., the exchangeable use of *POST* and *GET*. Additionally, the coherent use of HTTP verbs is complicated by technical restrictions, e.g., the limited size of GET requests or the filtering of DELETE requests through firewalls. To account for these deficits, we allow for the explicit specification of *verb similarities* between each pair of the four aforementioned verbs. Verb similarities are numerical values ranging from 0 to 1. This allows for the specification of (relative) compatability between operations types. The verb similarities are regarded in the determination of operation similarities. That is, the initially determined similarity value is multiplied with the verb similarity. An example configuration of verb similarity values is depcited in Table 1.

Another potential source of information in an hRESTS document are the resources (also referred to as *HTTP nouns*). These resources are specified using *Uniform Resource Identifiers* (URI), e.g., `http://quotes.provider.ex/stock/XYZ`. Such resource URI may potentially prove useful in the matchmaking process. We refrain from the utilization of resources in XAM4SWS, however, because we assume that the underlying semantics of any resource should be specified using the *model* attribute of the corresponding parameters. This is based on the notion that semantic information (as specified in a *model* attribute) provides more certainty than purely syntactical information (such as a resource URI).

Finally, we would like to discuss two possible lightweight extensions to the hRESTS format that may be valuable in the matchmaking process. As Pautasso et al. state, RESTful services encode their payload in various formats, e.g., Plain XML, *Java Script Object Notation* (JSON), or *Multipurpose Internet Mail Extensions* (MIME). The explicit specification of the payload format would introduce the potential to filter service offers for a specifically desired format. MIME is of additional elevated interest, because it specifies an *Internet Media Type* (also referred to as content type). This content-related information would also allow to filter service offers, given that a specific type of content – such as an image or a Portable Document Format (PDF) document – is requested. Accordingly, we suggest to augment hRESTS descriptions with a payload format and content type attribute for each operation.

4.4 Fallback Strategy and Caching

One potential problem in matchmaking is the lack of semantic annotations or the inability to process them. To address this issue, XAM4SWS implements a basic fallback strategy. It determines the similarity between two components based on their associated concept (or component) names using the WordNet ontology of the English language [21]. For that purpose, names are split into tokens using common separators, such as dash, underscore, and *camelCase*. Tokens that do not correspond to words in WordNet are recursively scanned for meaningful strings, again with the substrings being validated against WordNet. The resulting sets of words from the service request and offer component constitute partitions of a bipartite graph. The edge weights are assigned using the inverse distance of a pair of words in WordNet. The overall similarity of two names, and thus components, is then given by the average edge weight obtained in bipartite graph matching and lies in the range 0 to 1.

Table 1: Examplary Verb Similarity Values

		Request Verb			
		POST	GET	PUT	DELETE
Offer Verb	POST	1	0.8	0.5	0
	GET	0.8	1	0.2	0
	PUT	0.5	0.2	1	0
	DELETE	0	0	0	1

In the implementation, XAM4SWS employs different mechanisms for caching in order to improve the run-time performance. In detail, we use caches for the path length between semantic concepts, aforementioned splitting of names, and WordNet distances between pairs of words. This is motivated by the fact that all three processes consume a considerable amount of performance. The caches can be permanently stored for future reference and may not only be populated at query time, but also at registration time. For that matter, every new service offer is optionally matched against all registered services, thus increasing the number of entries and potential matches in the caches.

5. EVALUATION

5.1 Setup

To date, to the best of our knowledge, there exists no standard test collection for hRESTS. For the evaluation of XAM4SWS, we have selected *SAWSDL-TC*[2] as a basis. It constitutes the de facto standard that is employed both in related work [4, 5], as well as in the annual S3 contest of semantic matchmakers [2]. SAWSDL-TC consists of 894 service offers that cover various domains. It provides 26 service queries and corresponding (binary) relevance sets, encoded as WSDL 1.1 documents. As an important restriction, all services provide only one interface with one operation. Furthermore, semantic annotations are solely included at the parameter level, i.e., in the XSD section of the WSDL files. In line with our assumption in Section 2, all referenced semantic concepts are contained in OWL DL ontologies.

We have mapped all services from SAWSDL-TC to their respective hRESTS counterparts using a self-developed XSLT stylesheet. The stylesheet and the resulting test collection, hRESTS-TC, are available via SemWebCentral[3]. All aforementioned restrictions of SAWSDL-TC also apply to hRESTS-TC. In the mapping process, SAWSDL interface components have been converted into hRESTS service components; both constitute the level of abstraction right above operations. For the parameters, we only include the topmost XSD type that is referenced by a WSDL *part* construct, ignoring the underlying potentially complex structure of individual elements. Finally, all operations in hRESTS are assigned a default HTTP method of *POST*. Aforementioned restric-

[2] `http://projects.semwebcentral.org/projects/sawsdl-tc/`

[3] `http://projects.semwebcentral.org/projects/hrests-tc`

tions should be kept in mind when comparing the results of XAM4SWS for hRESTS to those of competing matchmaking approaches that natively operate on SAWSDL-TC.

We have conducted evaluation runs for different configurations of XAM4SWS. The variation concerns the weighting of abstraction levels and the assignment of numerical DoM equivalents. Version 1 conducts matching based on the service signature, i.e., input and output parameters. Input and output levels are given identical weights of 0.5 each. This version exclusively operates on semantic information, i.e., does not employ the fallback strategy. In Version 2, we assign a weight of 0.1 to the service and operation levels and of 0.4 to the input and output parameter levels respectively. This accounts for the fact that semantic information is only available on the parameter levels and should thus be regarded to a larger extent for similarity assessment. In Version 3, a naive approach is pursued, with all abstraction levels being assigned an identical weight of 0.25. For every version, we evaluate two different Variants, *a* and *b*. In Variant *a*, we follow a suggestion by Syeda-Mahmood et al. [22] and manually set the numerical DoM equivalents to 1 (for an exact DoM), 0.5 (for a sub and super DoM), and 0 (for a fail DoM) respectively. In Variant *b*, the numerical equivalents are determined by means of the OLS estimator. The weights are determined using k-fold cross-validation [23]. In our evaluation, $k = 26$ is given by the number of queries and corresponding relevance from hRESTS-TC that serve as partition from the service set. An overview of configurations is provided in Table 2. Due to the fact that all operations exhibit the *POST* method for invocation, a default verb compatibility of 1 was assumed for all operations.

XAM4SWS has been implemented in Java, using *Pellet 2.0* for logic reasoning and *JWNL 1.4* as the interface to WordNet. hRESTS documents are first converted into an equivalent *Resource Description Framework (RDF)* representation using a stylesheet by Kopecký[4]. For more details on the conversion process, we refer to Kopecký et al. [10]. The RDF document is consecutively read using *JENA 2.6*. To conduct the actual evaluation process, we utilize the Semantic Matchmaker Evaluation Environment (SME2)[5].

5.2 Utilized Metrics

To assess the matchmaking quality of XAM4SWS, we employ several metrics from the domain of IR. SME2 automatically computes $Precision = \frac{|A \cap B|}{|B|}$ and $Recall = \frac{|A \cap B|}{|A|}$, with A denoting the set of all relevant offers for a request and B denoting the set of all retrieved offers for a request.

In order to determine a mean precision for answer sets at standard recall levels, we also make use of macro-averaged precision $Precision_i = \frac{1}{|Q|} \times \sum_{q \in Q} \max\{P_O | R_O \geq Recall_i \wedge (R_O, P_O) \in O_q\}$, with O_q representing the set of recall precision value pairs of the relevant documents for query q. O_q is determined by a stepwise comparison of the sorted result set in descending order with the corresponding relevance set from hRESTS-TC [4]. SME2 uses equidistant steps $\frac{n}{\lambda}, n = 1 \ldots \lambda$ for the individual recall and precision levels. For our evaluation, the default value $\lambda = 20$ is applied. To assess the precision at low recall levels, i.e., for the first k offers in a result set, we employ Precision at k (*P(k)*). For k, we use the common values 5 and 10 (*P(5)* and *P(10)*, in short). Additionally, we provide R-Precision (RP), which corresponds to Precision at k, with k being the number of relevant services for a specific request. For RP, recall and precision is identical, i.e., RP identifies the break-even point of recall and precision. Finally, the Average Precision *(AP)* corresponds to the mean precision rate over all recall levels [24]. All these metrics are macro-averaged over all queries.

5.3 Results

Table 2 provides an overview of evaluation results for all previously described variants of XAM4SWS. Figures 2a and 2b further depict the recall-precision curves for the manually tuned and OLS-based variants respectively.

Results for other matchmaking approaches have not been included for two reasons. First, XAM4SWS is currently the only matchmaker to operate on hRESTS services. Second, as was explained in Section 5.1, the transformation from SAWSDL-TC to hRESTS-TC necessarily leads to a loss of information. This renders the results of XAM4SWS largely incomparable with those of SAWSDL-based matchmakers.

As it is evident from the result overview, the signature-based Version 1 delivers the worst performance with respect to the considered IR metrics. The inclusion of information from the service and operations level, which is employed in Versions 2 and 3, leads to a significant improvement. Figures 2a and 2b indicate that the gain in precision is most significant for low recall levels. This fact is also reflected in the changes in the P(5) and P(10) metrics. It is important to note that Version 2, which emphasizes the semantically annotated parameter level, achieves the best overall results. That means, the increasing consideration of non-semantic information in Version 3 triggers a decline of overall matchmaking precision. The utilization of OLS leads to significant improvements with respect to all IR metrics. A single exception can be observed for Version 1, where the use of OLS results in a reduction in P(5). The highest absolute performance gain is achieved for Version 2, with Variant 2b delivering the best overall results.

5.4 Discussion

As this work demonstrates, the principal mechanisms of matchmaking for WS-* services can be transferred to the RESTful world with limited effort. This is specifically true for the description formats that have been at the focus of this work, namely hRESTS and MicroWSMO. The combination of these two formats exhibits a significant number of common charateristics with the popular WSDL and SAWSDL standards. Most commonly, this concerns the structural organization of services and a comparable model for semantic annotation. As has been shown in our previous work [6], the extension of traditional subsumption reasoning with a numerical mapping may provide very good matchmaking results. The evaluation of XAM4SWS supports these findings in terms of the observed IR metrics. Unfortunately, the specific features of hRESTS could not be utilized due to the lack of a suitable test collection. This problem will hopefully be elevated by the research community in the future through the creation of such test set. Hence, our evaluation should be seen as preliminary.

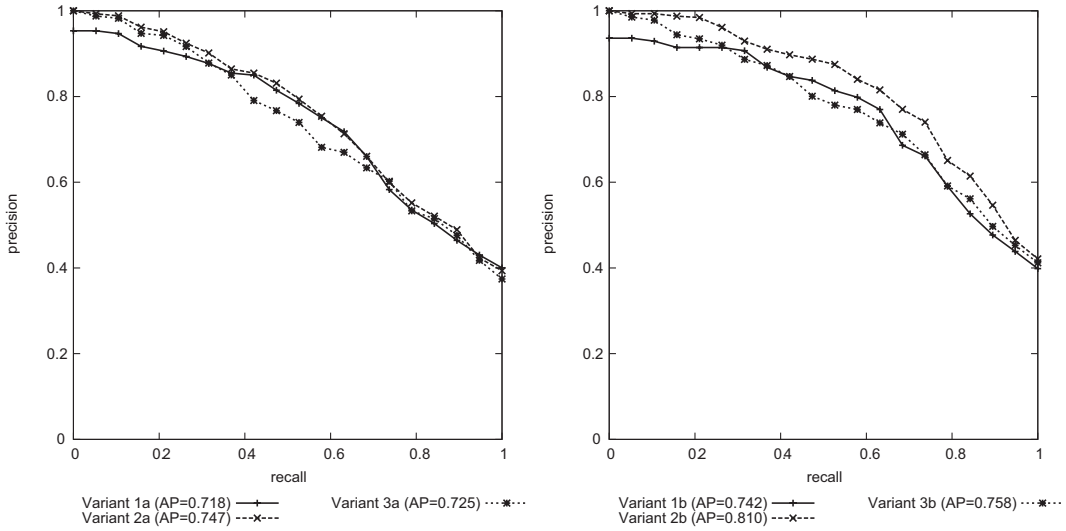
Because XAM4SWS is based on LOG4SWS.KOM, it also exhibits a potential drawback: In traditional subsumption

[4] `http://cms-wg.sti2.org/TR/d12/v0.1/20081202/xslt/hrests.xslt`

[5] `http://projects.semwebcentral.org/projects/sme2/`

Table 2: Configurations and Evaluation Results for XAM4SWS

#	Level Weights	Num. DOM Equivalents	AP	P(5)	P(10)	RP
1a	(0, 0, 0.5, 0.5)	(1, 0.5, 0.5, 0)	0.718	0.869	0.792	0.647
1b	(0, 0, 0.5, 0.5)	From OLS	0.742	0.846	0.827	0.678
2a	(0.1, 0.1, 0.4, 0.4)	(1, 0.5, 0.5, 0)	0.747	0.931	0.842	0.685
2b	(0.1, 0.1, 0.4, 0.4)	From OLS	0.810	0.962	0.885	0.736
3a	(0.25, 0.25, 0.25, 0.25)	(1, 0.5, 0.5, 0)	0.725	0.931	0.842	0.661
3b	(0.25, 0.25, 0.25, 0.25)	From OLS	0.758	0.954	0.835	0.700



(a) Manually Tuned Variants (b) OLS-Based Variants

Figure 2: Recall-Precision Curves

matching, as applied by Paolucci et al. and Klusch et al. [3,4,8], the overall DoM of two services is determined by the worst individual DoM between two matched components. This *global DoM* can be interpreted as a guaranteed lower bound of similarity between a service offer and the initial request. In contrast, XAM4SWS computes the overall similarity between two services based on the average DoM between all matched components. This average value reflects the degree of adaptation which is required to "fit" a service offer to the service request. As discussed in our previous work, this approach also has its advantages. First, if only a small number of services is publicly available, the similarity value provides a good estimate of required manual adaptions effort for the discovered services candidates. Second, it is more tolerant toward single outliers in the DoMs of the matched components.

6. CONCLUSIONS

The SOA paradigm is gaining increasing momentum for the realization of large, distributed IT systems and cross-organizational workflows. Today, WS-* (SOAP-based) Web services constitute the de facto standard for the implementation of SOA within business entities. However, RESTful services are receiving attraction as a lightweight alternative, e.g., in the realization of mashups. This development triggers the need for suitable matchmaking solutions, i.e., algorithms that discover functionally adequate service offers based on a given service request.

In the work at hand, we have introduced XAM4SWS, an adaptive matchmaker for RESTful services, based on the hRESTS description format in conjunction with MicroWSMO annotations. XAM4SWS extends our previously presented matchmaker LOG4SWS.KOM. XAM4SWS maps traditional Degrees of Match from subsumption reasoning into numerical equivalents. It thus allows for the integration of additional numerical similarity measures. As proof of concept, our matchmaker implements a path length-based measure as well as a WordNet-based fallback strategy.

XAM4SWS has been evaluated using the adapted version of a commonly accepted test collection for semantic Web services, SAWSDL-TC. In the evaluation, our matchmaker exhibits promising evaluation results with respect to common IR metrics such as precision and recall.

In our future work, we will address the inclusion of the WADL service description format in XAM4SWS. We will further evaluate additional adaption mechanisms besides the currently utilized OLS estimator. Lastly, we are encouraging the SWS community to participate in the creation of a commonly accepted test collection for RESTful services.

Acknowledgments

This work is partially supported by E-Finance Lab e. V., Frankfurt am Main, Germany (`www.efinancelab.de`). We

would like to thank Martin Pinto and Martin Prodanov for their assistance in the implementation of XAM4SWS.

7. REFERENCES

[1] N. M. Josuttis, *SOA in Practice: The Art of Distributed System Design.* O'Reilly Media, Sebastopol, CA, USA, 2007.

[2] M. Klusch, A. Leger, D. Martin, M. Paolucci, A. Bernstein, and U. Küster, "3rd International Semantic Service Selection Contest – Retrieval Performance Evaluation of Matchmakers for Semantic Web Services (S3 Contest)," 2009, http://www-ags.dfki.uni-sb.de/~klusch/s3/s3-2009-summary.pdf, access at 2010-09-20.

[3] M. Klusch and P. Kapahnke, "OWLS-MX3: An Adaptive Hybrid Semantic Service Matchmaker for OWL-S," in *Third International Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web (SMR2)*, 2009.

[4] M. Klusch, P. Kapahnke, and I. Zinnikus, "Hybrid Adaptive Web Service Selection with SAWSDL-MX and WSDL-Analyzer," in *6th European Semantic Web Conference (ESWC 2009)*, 2009, pp. 550–564.

[5] P. Plebani and B. Pernici, "URBE: Web Service Retrieval Based on Similarity Evaluation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 11, pp. 1629–1642, 2009.

[6] S. Schulte, U. Lampe, J. Eckert, and R. Steinmetz, "LOG4SWS.KOM: Self-Adapting Semantic Web Service Discovery for SAWSDL," in *IEEE 2010 Fourth International Workshop of Software Engineering for Adaptive Service-Oriented Systems (SEASS '10)*, 2010, pp. 511–518.

[7] S. A. McIlraith, T. C. Son, and H. Zeng, "Semantic Web Services," *IEEE Intelligent Systems*, vol. 16, no. 2, pp. 46–53, 2001.

[8] M. Paolucci, T. Kawamura, T. R. Payne, and K. P. Sycara, "Semantic Matching of Web Services Capabilities," in *First International Semantic Web Conference (ISWC 2002)*, 2002, pp. 333–347.

[9] C. Pautasso, O. Zimmermann, and F. Leymann, "RESTful Web Services vs. Big Web services: Making the Right Architectural Decision," in *17th International Conference on World Wide Web*, 2008, pp. 805–814.

[10] J. Kopeckỳ, K. Gomadam, and T. Vitvar, "hRESTS: an HTML Microformat for Describing RESTful Web Services," in *2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 2008, pp. 619–625.

[11] M. Maleshkova, C. Pedrinaci, and J. Domingue, "Supporting the Creation of Semantic RESTful Service Descriptions," in *Third International Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web (SMR2)*, 2009.

[12] R. Chinnici, J.-J. Moreau, A. Ryman, and S. Weerawarana, Eds., *Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language.* W3C Recommendation, June 2007, http://www.w3.org/TR/wsdl20/, access at 2010-09-07.

[13] J. Kopecký, T. Vitvar, C. Bournez, and J. Farrell, "SAWSDL: Semantic Annotations for WSDL and XML Schema," *IEEE Internet Computing*, vol. 11, no. 6, pp. 60–67, 2007.

[14] M. Hadley, "Web Application Description Language (WADL)," *Sun Microsystems, Inc. Technical Reports; Vol. SERIES13103; SMLI TR-2006-153*, 2006.

[15] C. Kiefer and A. Bernstein, "The Creation and Evaluation of iSPAQRL strategies for Matchmaking," in *5th European Semantic Web Conference (ESWC 2008)*, 2008, pp. 463–477.

[16] R. Battle and E. Benson, "Bridging the Semantic Web and Web 2.0 with Representational State Transfer (REST)," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 6, no. 1, pp. 61–69, 2008.

[17] R. Guo, D. Chen, and J. Le, "Matching Semantic Web Services across Heterogeneous Ontologies," in *Fifth International Conference on Computer and Information Technology (CIT 2005)*, 2005, pp. 264–268.

[18] U. Bellur and R. Kulkarni, "Improved Matchmaking Algorithm for Semantic Web Services Based on Bipartite Graph Matching," in *2007 IEEE International Conference on Web Services (ICWS 2007)*, 2007, pp. 86–93.

[19] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[20] K. A. Nedas, *Munkres' (Hungarian) Algorithm*, 2005, http://konstantinosnedas.com/dev/soft/munkres.htm, access at 2010-09-20.

[21] G. A. Miller, "WordNet: a Lexical Database for English," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[22] T. Syeda-Mahmood, G. Shah, R. Akkiraju, A.-A. Ivan, and R. Goodwin, "Searching Service Repositories by Combining Semantic and Ontological Matching," in *2005 IEEE International Conference on Web Services (ICWS 2005)*, 2005, pp. 13–20.

[23] T. M. Mitchell, *Machine Learning.* McGraw-Hill, New York, NY, USA, 1997.

[24] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval.* Cambridge University Press, New York, NY, USA, 2008.