Towards Service Matchmaking of Heterogeneous Service Descriptions[1]

Analay Baltá, Alberto Fernández

CETINIA, University Rey Juan Carlos, Móstoles, Spain
analay@ia.urjc.es, alberto.fernandez@urjc.es

Abstract. In service oriented architectures (SOA) the process of service discovery involves the matchmaking between service advertisements and service requests. Most approaches assume that both service advertisements and service requests are expressed using the same description language as well as domain ontologies. However, in large-scale open environments that assumption does not hold. In this paper we present an abstract architecture that addresses the semantic mismatches at service description level. In particular, we focus on the alignment of service description models and service matchmaking on the unified request and advertisements.

Keywords: service discovery, matchmaking, semantic web services, service oriented architecture, agreement technologies, semantic alignment.

1 Introduction

In multiagent systems, agents communicate with the aim of achieving their objectives. An individual agent may require a service to be performed by another entity. There are several stages since an agent identifies a given need until the service that provides it is eventually executed. First, the agent identifies some functionality that it is not able to perform or that might be executed more efficiently by an external entity. Then, candidate service providers must be located. Once a set of potential providers are known, the agent must choose one among them. This decision can be made based on several factors such as quality of service, price, reputation, etc. After the selection is made the two agents might engage in a negotiation about the conditions under which the service is going to be performed.

In this paper we concentrate on the phase of provider location. Distributed service directories and efficient decentralised matching techniques are essential for dynamic and scalable Web service discovery.

[1] Work partially supported by the Spanish Ministry of Science and Innovation through grants TIN2009-13839-C03-02 and CSD2007-0022 (CONSOLIDER-INGENIO 2010)

The process of discovering and interacting with a Semantic Web Service includes [1] *candidate service discovery* (match advertised service descriptions against specifications from requesters), *service engagement*, and *service enactment*.

Several description frameworks to annotate provided services on the one hand and express service requests on the other have been proposed. They range from logic-based complex and expressive semantic service descriptions (e.g. OWLS[2], WSMO[3]) to syntactical ones (WSDL[4], keywords, tag clouds), with some approaches in between (SAWSDL[5]). In this context, several frameworks to semantically match service advertisements and requests have been presented in the literature [4, 6, 9, 12].

In such open environments the mechanisms for locating appropriate services have to struggle with the additional problem of semantic mismatches among the *service description models or languages* as well as *domain ontologies* used to specify the concepts in the descriptions. Note that most approaches assume the use of the same language for both service advertisements and requests. Therefore, semantic alignment mechanisms need to be purposefully integrated into the service discovery mechanism.

In this paper we describe an architecture for service discovery where semantic alignment mechanisms are integrated into. The rest of the paper is organized as follows. In the next section we present the architecture. In section 3, we propose a service description model alignment including an integrated model which contains all the important characteristics of the existing models. We propose an integrated service matching framework in section 4. Finally, we review related works in section 5 and present some conclusions and future work.

2 Service Discovery Architecture

Figure 1 illustrates the proposed architecture that defines the service discovery functionality. The architecture comprises the building components to match a service request against a service advertisement. In particular, this proposal pays special attention to the problem of semantic mismatches between descriptions. Semantic mismatches are considered at two different levels:

- *Service description models.* Services (advertisements and requests) might be described using different languages or models (e.g. OWL-S, WSMO, SAWSDL, ...).
- *Domain ontology concepts.* Since semantic service descriptions rely on the use of domain ontologies, the second type of mismatch is due to the use of different domain ontologies to specify the concepts used in the descriptions.

Note that both options can be combined. For instance, two services might share the same service model (e.g. OWL-S) but use different domain ontologies, or they might use the same domain ontology but different service models.

---

[2] http://www.w3.org/Submission/OWL-S/
[3] http://www.w3.org/Submission/WSMO/
[4] http://www.w3.org/TR/wsdl
[5] http://www.w3.org/TR/sawsdl/

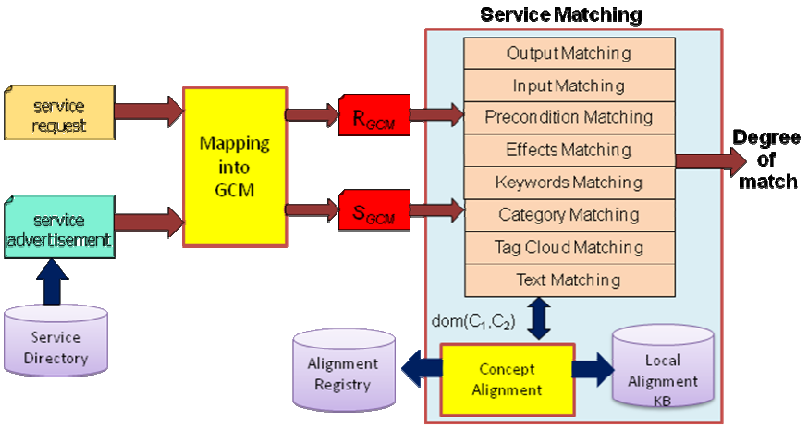

Fig. 1 Service Discovery Architecture

In the first step, we describe the mapping from the original models into a general common model (*GCM*) for service discovery purposes that integrates the relevant characteristics of service description languages. Therefore, service model alignment consists of mapping both descriptions (request and advertisement) into the GCM. Note that service description approaches not only differ in the language in which they are written. They are classified at different levels of expressiveness, ranging from complex, formal, logic-based semantic descriptions to lightweight syntactical ones.

The *Service Matching* module takes those two descriptions and returns the degree of match between them. As we will see in section 4, service matchmaking may include syntactic and semantic matching of description fragments (depending on the source descriptions). Semantic matchmaking algorithms usually include a semantic concept matching process to analyze the (similarity) relation between concepts used in advertisements and requests. As it was pointed out above, those concepts might belong to different ontologies so a *Semantic Concept Alignment* is carried out to solve this problem. In this case, we keep a local knowledge base of alignments and assume the existence of an external registry of alignments. The *local knowledge base* acts as a cache of alignments used in previous matchings. The *external alignment registry* is consulted to avoid carrying out a process of ontology matching if there is an alignment published by third parties.

In the next sections we detail the building blocks of the proposed architecture.

3 Service Description Model Alignment

As commented above, service model alignment consists of defining a common model for two different ones (the request and the candidate services), and mapping descriptions expressed in those source models into the common model.

In the next subsection we describe the characteristics of current service description approaches, so as to propose an integrated model which integrates the different common models. Afterwards, we present a unified common model for matchmaking.

3.1 Service description approaches

In this section we describe different approaches to service description. We include semantic models (OWL-S, WSMO), syntactic models (WSDL), hybrid (SAWSDL), as well as other lighter approaches. For each approach, we concentrate on the important characteristics from a service matchmaking point of view.

OWL-S. Web Ontology Language for Services (OWL-S) *service profile* is used to describe what the service does and it is crucial in the web service discovery process since it describes the capabilities of web services. We select the profile's relevant fields for service matchmaking so as to define a service by $\mathcal{S} = <\mathcal{I}, \mathcal{O}, \mathcal{P}, \mathcal{E}, \mathcal{C}, \mathcal{T}>$, where $\mathcal{I}$ represents a set of inputs (property *hasInput*), $\mathcal{O}$ is a set of outputs (property *hasOutput*), $\mathcal{P}$ is a set of preconditions (property *hasPrecondition*), $\mathcal{E}$ a set of effects (property *hasResult*), $\mathcal{C}$ is a set of categories (property *hasCategory*), and $\mathcal{T}$ is plain text description of the Web service (property *textDescription*).

WSMO. Web Service Modeling Ontology (WSMO) offers four key components to model different aspects of Semantic Web Services: *ontologies*, *goals*, *services*, and *mediators*. Web Service descriptions are defined into WSMO capability by their *precondition, postcondition, assumption, effect*, and their *nonFunctionalProperties* (*title, subject, description*). For our service discovery approach, WSMO and OWL-S share the same formalization, although these components are obtained in WSMO from different fields: $\mathcal{I}$ (*precondition*), $\mathcal{O}$ (*postcondition*), $\mathcal{P}$ (*assumption*), $\mathcal{E}$ *(effect*), $\mathcal{C}$ (*subject*), $\mathcal{T}$ (*description*).

WSDL. Web Service Description Language (WSDL) is an XML-based language for syntactically describing the service, including the service name, functions, and input and output parameters of abstract definitions. WSDL descriptions are limited in their search by their inability to extend beyond the keyword-based matches. We define a WSDL service $\mathcal{S} = <\mathcal{I}, \mathcal{O}, \mathcal{T}>$ where $\mathcal{I}$ is a set of inputs (strings extracted from the *wsdl:input* element defined in the *messages* of operations), $\mathcal{O}$ is the set of outputs (*wsdl:output*) and $\mathcal{T}$ is a text (*wsdl:documentation*).

SAWSDL. Semantic Annotations for WSDL and XML Schema (SAWSDL) introduces three new attributes: *modelReference* for specifying associations between WSDL components and semantic concepts, *liftingSchemaMapping* and *loweringSchemaMapping* for specifying mappings between semantic data and XML. Some of the elements might not be semantically annotated. For us, SAWSDL service is described by $\mathcal{S} = <\mathcal{I}, \mathcal{O}, \mathcal{T}>$ where $\mathcal{I}=<\mathcal{I}_{syn}, \mathcal{I}_{sem}>$ is a pair of sets containing the inputs not semantically annotated ($\mathcal{I}_{syn}$) and those annotated with concepts from ontologies ($\mathcal{I}_{sem}$), analogously for $\mathcal{O}=<\mathcal{O}_{syn}, \mathcal{O}_{sem}>$, and $\mathcal{T}$ is the description.

Keyword-based descriptions. A typical way of describing resources is by annotating them using keywords (or tags). It can also be applied to Service descriptions. E.g., *seekda!*[6] web service search engine includes free text and tag descriptions provided by service users, as well as WSDL files. We admit two kinds of keyword-based descriptions ($\mathcal{K} = <\mathcal{K}_{syn}, \mathcal{K}_{sem}>$): syntactic (free text) and semantic (ontology-based).

[6] http://webservices.seekda.com/

Tag Cloud based descriptions. The advantage of using tags in the context of service discovery is that they supply a user-defined vocabulary based on a consensus of how the service is perceived or used in the world [3, 14]. A tag-cloud-based service description is a set of pairs $\mathcal{TC} = \{<t, n>\}$, where t is a free text tag and n is the frequency of the tag t in the cloud.

Textual descriptions. Services can be defined by free text human-readable information. For these kinds of descriptions service discovery is based on information retrieval techniques (IR).

3.2 Integrated Model for service discovery

From the analysis of the different approaches to service descriptions (section 3.1) we obtain an integrated *general common model (GCM)*.

For each approach, we present a brief description and a formal specification of the selected characteristics from a discovery point of view. Semantic service descriptions include concepts defined in some domain ontologies. In the following, we consider such domain ontology $O = <\mathcal{N}, \prec>$, where:

- $\mathcal{N}$ represents the set of concept names, and
- $\prec$ is a partial order representing the *subclass of* relation between $\mathcal{N}$ concepts.

O can be considered as the union of all domain ontologies used by the different service descriptions. Note that, for service matchmaking purposes, we only require ontology fragments including the *subclass of* relation.

Definition 1. Let $\mathcal{N}$ be a set of concepts of domain ontologies, a *general common model* (*GCM*) for service discovery is a tuple $<\mathcal{I}_{GCM}, \mathcal{O}_{GCM}, \mathcal{P}_{GCM}, \mathcal{E}_{GCM}, \mathcal{K}_{GCM}, \mathcal{C}_{GCM}, \mathcal{T}_{GCM}, \mathcal{TC}_{GCM}>$, where:

- $\mathcal{I}_{GCM} = <I_{syn}, I_{sem}>$ is the set of syntactic ($I_{syn} \in \{a, .., z\}^*$) and semantic ($I_{sem} \subseteq N$) inputs of the service.
- $\mathcal{O}_{GCM} = <O_{syn}, O_{sem}>$ is the set of syntactic ($O_{syn} \subseteq \{a, ..., z\}^*$) and semantic ($O_{sem} \subseteq N$) outputs.
- $\mathcal{P}_{GCM}$ is the set of preconditions. $\mathcal{P}_{GCM} \subseteq \mathcal{N}$
- $\mathcal{E}_{GCM}$ is the set of effects. $\mathcal{E}_{GCM} \subseteq \mathcal{N}$
- $\mathcal{K}_{GCM} = <\mathcal{K}_{syn}, \mathcal{K}_{sem}>$ is the set of syntactic and semantic keywords, where $\mathcal{K}_{syn} \subseteq \{a, \ldots, z\}^*$, $\mathcal{K}_{sem} \subseteq \mathcal{N}$.
- $\mathcal{C}_{GCM}$ is a set of categories of the service, described semantically ($C_{sem} \subseteq \mathcal{N}$) (e.g. NAICS or UNSPSC).
- $\mathcal{T}_{GCM}$ is a textual description of the service.
- $\mathcal{TC}_{GCM}$ is a tag cloud. $\mathcal{TC}_{GCM} = \{<t, n> \mid t \subseteq \{a, \ldots, z\}^*, n \in \mathbb{N}\}$.

We opt for using RDF as the representation language for the *GCM*. RDF is a W3C recommendation for representing data on the Web. A lot of RDF contents and tools to process them are available. Although RDF is less expressive than OWL-S, WSMO and SAWSDL, it is enough for representing the information needed for semantic service search. Furthermore, the use of RDF will also allow the exploitation of SPARQL [19] to query service descriptions.

3.3 Mapping service descriptions into the GCM

Table 1 shows how the different elements of the *GCM* can be obtained from each source service description model. The first column specifies the element of the *GCM*, while each cell contains the value mapped from the model shown in the first row.

There are many straightforward mappings that consist of simple associations between parameters in both models. For instance, in OWLS/WSMO $\mathcal{I}_{GCM}$=<ϕ, pt($\mathcal{I}$)> because they only provide semantically described inputs $\mathcal{I}$ ($\mathcal{I}_{sem}$), where pt($\mathcal{I}$)={ t | t=parameterType(i) $\forall i \in \mathcal{I}$}. The contrary applies to WSDL, where only the syntactic values are filled ($\mathcal{I}_{GCM}$=<$\mathcal{I}$, ϕ>). However, SAWSDL may contain both syntactic and semantic descriptions, thus $\mathcal{I}_{GCM}$=$\mathcal{I}$ since $\mathcal{I}$ = <I_{syn}, I_{sem}>. The same is applied to the outputs. Trivial mappings apply to preconditions ($\mathcal{P}_{GCM}$), effects ($\mathcal{E}_{GCM}$), categories ($\mathcal{C}_{GCM}$) and textual descriptions ($\mathcal{T}_{GCM}$).

However, some fields (tag-clouds, keywords) may not be explicitly described by a given model but they can be obtained from the rest of the description. **Tag-clouds** can be calculated from textual descriptions by means of a function $\Delta(\mathcal{T})$, which returns the *k most relevant* words from the text $\mathcal{T}$ as well as their frequency. We adopt information retrieval (IR) techniques to obtain that information through a process of (i) word extraction, (ii) stemming, and (iii) filtering out non relevant terms (chosen heuristically). The case of keyword-based service descriptions (where no text is included), a plain cloud is created with frequency 1 for every keyword in the description.

Syntactic **keywords** can be easily obtained from tag clouds (either original or calculated with Δ), by simply adopting the *k* most relevant words (function $\tau(TC)$, being *TC* a tag-cloud). The input concept name $\mathcal{N}(\mathcal{I})$ and output concept name $\mathcal{N}(\mathcal{O})$ in semantic descriptions (OWL-S, WSMO, SAWSDL) are adopted as semantic keywords.

Table 1. Service(S)-to-*GCM* mapping

GCM	**OWL-S / WSMO**	**SAWSDL**	**WSDL**	**Keyword (tag)**	**Tag Cloud**	**Text**
$\mathcal{I}_{GCM}$	<ϕ,pt($\mathcal{I}$)>	<$\mathcal{I}_{syn}$,$\mathcal{I}_{sem}$ >	<$\mathcal{I}$, ϕ>	<ϕ, ϕ>	<ϕ, ϕ>	<ϕ, ϕ>
$\mathcal{O}_{GCM}$	<ϕ pt($\mathcal{O}$)>	<$\mathcal{O}_{syn}$,$\mathcal{O}_{sem}$>	<$\mathcal{O}$, ϕ>	<ϕ, ϕ>	<ϕ, ϕ>	<ϕ, ϕ>
$\mathcal{P}_{GCM}$	$\mathcal{P}$	ϕ	ϕ	ϕ	ϕ	ϕ
$\mathcal{E}_{GCM}$	$\mathcal{E}$	ϕ	ϕ	ϕ	ϕ	ϕ
$\mathcal{K}_{GCM}$	<$\tau(\Delta(\mathcal{T}))\cup$ $\mathcal{N}(\mathcal{I})\cup\mathcal{N}(\mathcal{O})$, pt($\mathcal{I}$)$\cup$pt($\mathcal{O}$)>	<$\tau(\Delta(T)) \cup \mathcal{I}_{syn}$ $\cup\ \mathcal{O}_{syn}$, $\mathcal{N}(\mathcal{I}_{sem})\cup\mathcal{N}(\mathcal{O}_{sem})$ >	<$\tau(\Delta(\mathcal{T})$ $\cup\ \mathcal{I}_{syn}\cup$ $\mathcal{O}_{syn}$, ϕ>	$\mathcal{K}$	<$\tau(S)$, ϕ>	<$\tau(\Delta(S)$, ϕ>
$\mathcal{C}_{GCM}$	$\mathcal{C}$	ϕ	ϕ	ϕ	ϕ	ϕ
$\mathcal{T}_{GCM}$	$\mathcal{T}$	$\mathcal{T}$	$\mathcal{T}$	ϕ	ϕ	S
$\mathcal{TC}_{GCM}$	$\Delta(\mathcal{T})$	$\Delta(\mathcal{T})$	$\Delta(\mathcal{T})$	{<t, 1>| $t\in$ $\mathcal{K}_{syn}\cup\mathcal{K}_{sem}$}	S	$\Delta(S)$

3.4 Examples

In this section we show our proposal with an example of an airline service (the Bravo Air example[7]), a fictitious service that provides flight reservations based on the specification on a flight request. We present an OWL-S and a SAWSDL representation of that service, as well as their mapping to the *GCM*.

3.4.1 OWL-S

We define the OWL-S BravoAir example service $S = <\mathcal{I}, \mathcal{O}, \mathcal{P}, \mathcal{E}, \mathcal{C}, \mathcal{T}>$ as follows:

$\mathcal{I}$ = {ba[8]:DepartureAirport, ba:ArrivalAirport, ba:OutboundDate, ba:InboundDate,ba:RoundTrip,ba:AcctName,ba:Password,ba:Confirm}

$\mathcal{O}$ = {ba:FlightsFound, ba:PreferredFlightItinerary, ba:ReservationID}

$\mathcal{P} = \phi$

$\mathcal{E}$ = {ba:HaveSeatResult}

$\mathcal{C}$ = {unspsc[9]:Travel_agents}

$\mathcal{T}$ = "*This service provides flight reservations based on the specification on a flight request. This typically involves a departure airport, an arrival airport, a departure date, and if a return trip is required, a return date. If the desired flight is available, an itinerary and reservation number will be returned.*"

Figure 2 shows a partial view of the OWL-S profile for that service. In particular, it shows the definition of the input *DepartureAirport*. There, we can see that the semantic input concepts are obtained from the value of the *parameterType* property (in this case *concepts:Airport*) of OWL-S input *DepartureAirport* in the BravoAir service. The same applies for other inputs, outputs, preconditions and effects.

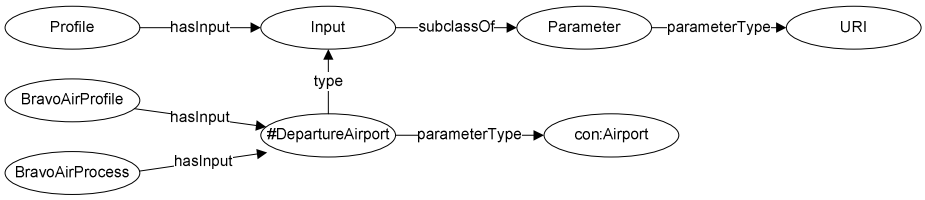

Fig. 2 OWL-S mapping

From the OWL-S service description we can obtain the service mapping described in the *GCM* according to Table 1. There are no syntactic inputs, and the semantic inputs would be the semantic concepts (parameter types) of inputs $\mathcal{I}$. The same applies to the outputs, preconditions and effects. The tag cloud includes the most relevant words of the text $\mathcal{T}$ as well as their frequency. The syntactic keywords are obtained using tag clouds with frequency 1 as well as the inputs and outputs concept

[7] http://www.ai.sri.com/daml/services/owl-s/1.0/examples.html

[8] ba = "http://www.daml.org/services/owl-s/1.1/BravoAirProcess.owl#"

[9] unspsc = "http://www.cs.vu.nl/~mcaklein/unspsc/unspsc84-title.rdfs#"

names. As for the semantic keywords the semantic concepts of inputs and outputs (parameter types of inputs $\mathcal{I}$ and outputs $\mathcal{O}$) are taken. Therefore:

S_{GCM} = <$\mathcal{I}_{GCM}$, $\mathcal{O}_{GCM}$, $\mathcal{P}_{GCM}$, $\mathcal{E}_{GCM}$, $\mathcal{K}_{GCM}$, $\mathcal{C}_{GCM}$, $\mathcal{T}_{GCM}$, $\mathcal{TC}_{GCM}$ > where:

$\mathcal{I}_{GCM}$= <ϕ,{con[10]:Airport, con:Airport, con:FlightDate, con:FlightDate, con:RoundTrip, con:AcctName, con:Password, con:Confirmation}>

$\mathcal{O}_{GCM}$= <ϕ,{con:FlightList,con:FlightItinerary, con:ReservationNumber}>

$\mathcal{P}_{GCM}$ = ϕ

$\mathcal{E}_{GCM}$ = ba:HaveSeatResult

$\mathcal{K}_{GCM}$ = <{flight, airport, reservation, departure, DepartureAirport, ArrivalAirport,OutboundDate,InboundDate,RoundTrip,AcctName, Password, Confirm, FlightsFound, PreferredFlightItinerary, ReservationID}, {con:Airport, con:Airport, con:FlightDate, con:FlightDate, con:RoundTrip, con:AcctName, con:Password, con:Confirmation, con:FlightList, con:FlightItinerary, con:ReservationNumber}>

$\mathcal{C}_{GCM}$ = {unspsc:Travel_agents}

$\mathcal{T}_{GCM}$ = "*This service provides flight …*"

$\mathcal{TC}_{GCM}$ = {<flight,3>, <airport,2>, <reservation,2>, <departure,2>}.

3.4.2 SAWSDL

We define the Bravo Air service in SAWSDL as S=<$\mathcal{I}$,$\mathcal{O}$,$\mathcal{T}$>, where:

$\mathcal{I}$ = <{DepartureAirport, ArrivalAirport, OutboundDate, InboundDate, RoundTrip, AcctName, Password, Confirm}, {ai[11]:AirportCodes, ai:AirportCodes, date[12]:Date, date:Date, con:RoundTrip, con:AcctName, con:Password, con:Confirmation}>

$\mathcal{O}$ = <{FlightsFound, PreferredFlightItinerary, ReservationID}, {con:FlightList, con:FlightItinerary, con:ReservationNumber}>

$\mathcal{T}$= *This service provides flight …*"

Note that we consider semantic annotation for every input and output. In this case, we use *ai:AirportCodes* and *ai:Date* to annotate inputs and outputs related to airports and dates, respectively.

From the SAWSDL service description we obtain the following service mapping described in the *GCM*.

S_{GCM} = <$\mathcal{I}_{GCM}$, $\mathcal{O}_{GCM}$, $\mathcal{P}_{GCM}$, $\mathcal{E}_{GCM}$, $\mathcal{K}_{GCM}$, $\mathcal{C}_{GCM}$, $\mathcal{T}_{GCM}$, $\mathcal{TC}_{GCM}$ > where:

$\mathcal{I}$ = <{DepartureAirport, ArrivalAirport, OutboundDate, InboundDate, RoundTrip, AcctName, Password, Confirm}, {ai:AirportCodes, ai:AirportCodes, date:Date, date:Date, con:RoundTrip, con:AcctName, con:Password, con:Confirmation}>

$\mathcal{O}$ = <{FlightsFound, PreferredFlightItinerary, ReservationID}, {con:FlightList, con:FlightItinerary, con:ReservationNumber}>

$\mathcal{P}_{GCM}$ = ϕ

$\mathcal{E}_{GCM}$ = ϕ

[10] con = http://www.daml.org/services/owl-s/1.0/Concepts.owl

[11] ai = http://www.daml.ri.cmu.edu/ont/AirportCodes.daml

[12] date = http://www.ai.sri.com/daml/ontologies/sri-basic/1-0/Date.daml

$\mathcal{K}_{GCM}$ = <{flight, airport, reservation, departure, DepartureAirport, ArrivalAirport, OutboundDate, InboundDate, RoundTrip, AcctName, Password, Confirm, FlightsFound, PreferredFlightItinerary, ReservationID}, {ai:AirportCodes, ai:AirportCodes, date:Date, date:Date, con:RoundTrip, con:AcctName, con:Password, con:Confirmation, con:FlightList, con:FlightItinerary, con:ReservationNumber}>

$\mathcal{C}_{GCM} = \phi$

$\mathcal{T}_{GCM}$ = "*This service provides flight …*"

$\mathcal{TC}_{GCM}$ = {<flight,3>, <airport,2>, <reservation,2>, <departure,2>}.

4 Service Matchmaking

4.1 Semantic Concept Matching

Since service descriptions are composed by several elements or fields, each of them defined by concept descriptions, the process of semantic matchmaking includes a concept matching task. This function takes two concepts, C_A (advertisement) and C_R (request), and returns the degree of match between them.

Many of the current proposals for defining the degree of match between service advertisements and requests are based on subsumption checking of concepts present in inputs and outputs of service descriptions. We adopt the four degrees of match proposed by Paolucci et al. [12]: *exact* (C_A = C_R), *plug-in* (C_R subsumes C_A), *subsumes* (C_A subsumes C_R) and *fail* (otherwise).

Several similarity (or distance) measures for concept matching have been proposed in the literature, although their application to the concrete domain of service matching is very limited. One of the most well known distance measures between concepts is the length of the shortest path between them in the taxonomy, proposed by Rada et al [16]. Other proposals further refine that approach ([8, 10, 20]).

We propose a combination of service matching and concept similarity. In [4] we describe how both approaches can be combined into a unified service selection framework which returns a numeric value that can be used for ranking services. The ranking function compares the level of match first (exact, plugin, subsumes, fail), and then the level is refined with the (numerical) similarity value. The degree of match between two concepts is defined as follows.

Definition 2. The degree of match between two concepts C_A (advertisement) and C_R (request) is given by:

$$conceptMatch(C_R, C_A) = \begin{cases} 1 & if \quad C_A = C_R \\ \frac{1}{2} + \frac{1}{2} sim(C_A, C_R) & if\ C_R\ subsumes\ C_A \\ \frac{1}{2} sim(C_A, C_R) & if\ C_A\ subsumes\ C_R \\ 0 & otherwise \end{cases}$$

Where concept similarity (*sim*) is calculated as [10]:

$$\mathrm{sim}(C_1, C_2) = \begin{cases} e^{-\alpha l} \cdot \dfrac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} & if \quad C_1 \neq C_2 \\ 1 & otherwise \end{cases}$$

where $\alpha \geq 0$ and $\beta \geq 0$ are parameters scaling the contribution of the shortest path length (l) between the two concepts and the depth (h) of the least common subsumer in the concept hierarchy, respectively.

4.2 Service Matching

Since service descriptions consist of several components (inputs, outputs, ...), the similarity between services must be defined based on its individual elements (e.g. each of its inputs) and aggregation operators.

For service matching, given the *GCM* representation of the service advertisement (GCM_A) and the service request (GCM_R), we analyze the common elements defined in them. Only those non-empty fields belonging to the intersection of both ($GCM_A \cap GCM_R$) are considered here, as they define the common model to A and R (CM_{AR}).

For each pair of the common elements we propose a matching algorithm taking into account the semantic concept matching, being boolean (0, 1) in the case of syntactic values. Finally an aggregation function is applied to combine the results.

Definition 3. Given a service request GCM_R and a service advertisement GCM_A where, according to Definition 1, $GCM_R = <\mathcal{I}_R, \mathcal{O}_R, \mathcal{P}_R, \mathcal{E}_R, \mathcal{K}_R, \mathcal{C}_R, \mathcal{T}_R, \mathcal{TC}_R>$ and $GCM_A = <\mathcal{I}_A, \mathcal{O}_A, \mathcal{P}_A, \mathcal{E}_A, \mathcal{K}_A, \mathcal{C}_A, \mathcal{T}_A, \mathcal{TC}_A>$. Then, the *degree of match* between GCM_R and GCM_A is calculated as:

$dom(GCM_R, GCM_A) = f(IM(\mathcal{I}_R, \mathcal{I}_A), OM(\mathcal{O}_R, \mathcal{O}_A), PM(\mathcal{P}_R, \mathcal{P}_A), EM(\mathcal{E}_R, \mathcal{E}_A),$
$KM(\mathcal{K}_R, \mathcal{K}_A), CM(\mathcal{C}_R, \mathcal{C}_A), TM(\mathcal{T}_R, \mathcal{T}_A), TCM(\mathcal{TC}_R, \mathcal{TC}_A))$

IM, OM, PM, EM, KM, CM, TM and *TCM* are defined below.

As a service matching example, we will use the services described in sections 1.1.1 and 3.4.2 (S^1_{GCM} and S^2_{GCM}), acting the OWL-S example as the service advertisement ($GCM_A = S^1_{GCM}$) and the SAWSDL as the service request ($GCM_R = S^2_{GCM}$).

Output matching (*OM*).

In line with Paolucci's proposal [12], an **output** matches if and only if for each output of the request there is a matching output in the service description, i.e. the service provides all the outputs required.

Given $\mathcal{O}_R = <O^R_{syn}, O^R_{sem}>$ and $\mathcal{O}_A = <O^A_{syn}, O^A_{sem}>$, $OM(\mathcal{O}_R, \mathcal{O}_A)$ is calculated as a combination of the semantic match (*OSemM*) and the syntactic output match (*OSynM*). If both services include semantic and syntactic outputs, then a linear combination of those matches is calculated. However, it may occur that some of the sets are empty. In that case, we apply only the semantic match if both semantic descriptions (O^R_{sem} and O^A_{sem}) are available. Otherwise, only the syntactic match is calculated, completing each set of syntactic outputs with the names of the semantic

concepts ($\mathcal{N}(O^R_{sem})$ or $\mathcal{N}(O^A_{sem})$) if any of them is specified. The next equation gives the details.

$$\mathrm{OM}(\mathrm{O_R}, \mathrm{O_A}) = \begin{cases} \mathrm{OSemM}(\mathrm{O^R_{sem}}, \mathrm{O^A_{sem}}) & \text{if } \mathrm{O^R_{sem}} \neq \phi \wedge \mathrm{O^A_{sem}} \neq \phi \wedge (\mathrm{O^R_{syn}} = \phi \vee \mathrm{O^A_{syn}} = \phi) \\ \mathrm{OSynM}(\mathrm{O^R_{syn}}, \mathrm{O^A_{syn}}) & \text{if } \mathrm{O^R_{sem}} = \phi \wedge \mathrm{O^A_{sem}} = \phi \wedge \mathrm{O^R_{syn}} \neq \phi \wedge \mathrm{O^A_{syn}} \neq \phi \\ \mathrm{OSynM}(\mathrm{N}(\mathrm{O^R_{sem}}), \mathrm{O^A_{syn}}) & \text{if } \mathrm{O^R_{sem}} \neq \phi \wedge \mathrm{O^A_{sem}} = \phi \wedge \mathrm{O^R_{syn}} = \phi \wedge \mathrm{O^A_{syn}} \neq \phi \\ \mathrm{OSynM}(\mathrm{O^R_{syn}}, \mathrm{N}(\mathrm{O^A_{sem}})) & \text{if } \mathrm{O^R_{sem}} = \phi \wedge \mathrm{O^A_{sem}} \neq \phi \wedge \mathrm{O^R_{syn}} \neq \phi \wedge \mathrm{O^A_{syn}} = \phi \\ \mathrm{OSynM}(\mathrm{O^R_{syn}} \cup \mathrm{N}(\mathrm{O^R_{sem}}), \mathrm{O^A_{syn}}) & \text{if } \mathrm{O^R_{sem}} \neq \phi \wedge \mathrm{O^A_{sem}} = \phi \wedge \mathrm{O^R_{syn}} \neq \phi \wedge \mathrm{O^A_{syn}} \neq \phi \\ \mathrm{OSynM}(\mathrm{O^R_{syn}}, \mathrm{O^A_{syn}} \cup \mathrm{N}(\mathrm{O^A_{sem}})) & \text{if } \mathrm{O^R_{sem}} = \phi \wedge \mathrm{O^A_{sem}} \neq \phi \wedge \mathrm{O^R_{syn}} \neq \phi \wedge \mathrm{O^A_{syn}} \neq \phi \\ \alpha * \mathrm{OSemM} + \beta * \mathrm{OSynM} \mid \alpha + \beta = 1 & \text{if } \mathrm{O^R_{sem}} \neq \phi \wedge \mathrm{O^A_{sem}} \neq \phi \wedge \mathrm{O^R_{syn}} \neq \phi \wedge \mathrm{O^A_{syn}} \neq \phi \\ 0 & \textit{otherwise} \end{cases}$$

The semantic match is obtained by taking, for each output in the request, the best match (section 4.1) against the ones in the advertisement. The worst case (minimum value) is then chosen to combine the best matches, i.e.:

$$\mathrm{OSemM}(\mathrm{O^R_{sem}}, \mathrm{O^A_{sem}}) = \underset{\mathrm{o^R} \in \mathrm{O^R_{sem}}}{\mathrm{Min}} \; \underset{\mathrm{o^A} \in \mathrm{O^A_{sem}}}{\mathrm{Max}} \left\{ \mathrm{conceptMatch}(\mathrm{o^R}, \mathrm{o^A}) \right\}$$

At the syntactic level, we check whether all the required outputs are provided by the service:

$$\mathrm{OSynM}(\mathrm{O^R_{syn}}, \mathrm{O^A_{syn}}) = \begin{cases} 1 & \text{if } \mathrm{O^R_{sym}} \subseteq \mathrm{O^A_{syn}} \\ 0 & \textit{otherwise} \end{cases}$$

In our example, $\mathrm{OM}(\mathcal{O}_R,\mathcal{O}_A)$ = $\mathrm{OSemM}(\mathcal{O}^R_{sem}, \mathcal{O}^A_{sem})$ because only the service request includes syntactic outputs. Note that exactly the same semantic concepts are used in the output annotations in both services. Thus:

$\mathrm{OM}(\mathcal{O}^2_{GCM,}\mathcal{O}^1_{GCM})$ = $\mathrm{OSemM}(\mathcal{O}^2_{GCM,}\, \mathcal{O}^1_{GCM})$ = Min Max $\{\mathrm{conceptMatch}(\mathcal{O}^1_{GCM,}\, \mathcal{O}^2_{GCM})\}$
= Min(Max(1,0,0), Max(0,1,0), Max(0,0,1)) = 1

Input matching (*IM*).

For input matching (*IM*), an analogous approach is followed, but with the order of request and advertisement reversed.

In this case, there are *exact* matches for all the inputs except for the concepts *ai:AirportCodes* and *ai:Date* (request) vs *con:Airport* and *con:FlightDate* (advertisement). Analyzing the ontologies in which those concepts are specified, we can see that *con:Airport* is a subclass of *ai:AirportCode* and *con:FlightDate* is a subclass of *Date*.

In both cases, the *ConceptMatch* is 0.73, (taking $\alpha, \beta = 0.5$, l= 1, h= 1).

As occurred with the output matching, only the semantic match is calculated:

$\mathrm{IM}(\mathcal{I}^2_{GCM,}\mathcal{I}^1_{GCM})$ = $\mathrm{ISemM}(\mathcal{I}^2_{GCM,}\, \mathcal{I}^1_{GCM})$ = Min Max $\{\mathrm{conceptMatch}(\mathcal{I}^1_{GCM,}\, \mathcal{I}^2_{GCM})\}$ = Min(Max(0.73, 073, 0, 0, 0, 0, 0, 0), Max(0, 0, 0.73, 0.73, 0, 0, 0, 0), Max(0, 0, 0, 0, 1, 0, 0, 0) , Max(0, 0, 0, 0, 0, 1, 0, 0), Max(0, 0, 0, 0, 0, 0, 1, 0) , Max(0, 0, 0, 0, 0, 0, 0, 1)) = 0.73

Precondition/Effect matching (*PM/EM*).

Since preconditions and effects can usually not be expressed in a taxonomy or ontological hierarchy, rather more complex formalisms (e.g. SWRL rules) are proposed to describe these. We are not aware of service matching approaches which practically exploit precondition/effect matching at the moment. It seems to be not entirely clear, how precondition/effect matching can be done in open service environments, which might also be a reason why they have not been considered e.g. in SAWSDL. For these reason we consider preconditions and effects specified by concept definitions, and keep them as they were specified in the original model.

This way, it is up to the matching process to deal with those descriptions. For the moment we take a simple approach (concept matching), although it is a matter of future research how to improve that aspect.

$$\mathrm{PM}(\mathrm{P_R}, \mathrm{P_A}) = \underset{\mathrm{p^R} \in \mathrm{P_R}}{\mathrm{Min}} \underset{\mathrm{p^A} \in \mathrm{P_A}}{\mathrm{Max}} \left\{ \mathrm{conceptMatch}(\mathrm{p^R}, \mathrm{p^A}) \right\}$$

$$\mathrm{EM}(\mathrm{E_R}, \mathrm{E_A}) = \underset{\mathrm{e^R} \in \mathrm{E_R}}{\mathrm{Min}} \underset{\mathrm{e^A} \in \mathrm{E_A}}{\mathrm{Max}} \left\{ \mathrm{conceptMatch}(\mathrm{e^R}, \mathrm{e^A}) \right\}$$

Keyword matching (*KM*)

Given $\mathcal{K}_R = <K^R_{syn}, K^R_{sem}>$ and $\mathcal{A} = \mathcal{K}_A = <K^A_{syn}, K^A_{sem}>$, keyword matching is based on the syntactic and semantic keywords:

$\mathrm{KM}(\mathrm{K_R}, \mathrm{K_A}) = \alpha * \mathrm{KSemMatch}(\mathrm{K^R_{sem}}, \mathrm{K^A_{sem}}) + \beta * \mathrm{KSynMatch}(\mathrm{K^R_{syn}}, \mathrm{K^A_{syn}})$, with $\alpha, \beta = 1$.

For syntactic **keywords** the degree of match is determined by the ratio of coincident keywords of the candidate service (A) and the service requested (R).

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

For semantic concept keyword, we adopt Ehrig [2] measure to compare sets of concepts:

$$\mathrm{KSemMatch}(\mathrm{R}, \mathrm{A}) = \frac{\sum_{r \in R} \vec{r}}{\left|\sum_{r \in R} \vec{r}\right|} \cdot \frac{\sum_{a \in A} \vec{a}}{\left|\sum_{a \in A} \vec{a}\right|}$$

with $\vec{r} = (sim(r, r_1), sim(r, r_2), ..., sim(r, a_1), sim(r, a_2)...)$, $\vec{a}$ analogously.

Tag-cloud matching (*TCM*)

In the context of service matching, **tag-clouds** have a natural correspondence to the typical vector space model used in standard information retrieval. Using the vector-space model, each tag-cloud is represented as a vector in term space and each term in the vector is weighted according to the standard *tf-idf* weighting scheme [17]. In the vector-space model, similarity is calculated using the cosine measure. To prevent large clouds having undue influence during similarity matching, each vector is normalized so that it is of unit length on the hypersphere.

Text matching (*TM*)

In the case of **text** element, it also has syntactic information (or other syntactic element) so we apply IR techniques to determine its matching degree.

Category matching (*CM*)

For the **category** field, which may be described by a concept from an ontology, a 0/1 value or the result of the semantic concept similarity function is taken, respectively.

We require that for each category in the request *R* at least there is one matching category in *A*. Therefore, similar to output matching:

$$CM(\mathcal{C}_R, \mathcal{C}_A) = \min_{c^R \in C_R} \max_{c^A \in C_A} \{\text{conceptMatch}(c^R, c^A)\}$$

Finally, service matching must combine the similarity value for each of these fields, Definition 3 ($dom(GCM_R, GCM_A)$). Different options can be considered for *f* (f($IM(\mathcal{I}_R,\mathcal{I}_a)$, $OM(\mathcal{O}_R,\mathcal{O}_A)$, $PM(\mathcal{P}_R,\mathcal{P}_A)$, $EM(\mathcal{E}_R,\mathcal{E}_A)$, $KM(\mathcal{K}_R,\mathcal{K}_A)$, $CM(\mathcal{C}_R,\mathcal{C}_A)$, $TM(\mathcal{T}_R,\mathcal{T}_A)$, $TCM(\mathcal{TC}_R,\mathcal{TC}_A)$)). If we consider those fields as a conjunctive set (i.e. all are expected to be matched) then a triangular norm (e.g. the *minimum*) can be used.

However, for the moment, a more general approach is taken: a weighted sum of each similarity, where the weighting parameters are equally distributed (i.e. the mean).

4.3 Concept Alignment

In the previous section we saw that current service matchmaking algorithms are based on checking the relations between the concepts that appear in the different fields of semantic service descriptions. If the concepts being compared are defined in different ontologies then semantic alignments must be considered instead of obtaining a *fail* match.

An alignment (or mapping) between two ontologies *O* and *O'* can be described as a quadruple [11]: *<e, e', n, R>*

where:

- *e* and *e'* are the entities between which a relation is asserted by the mapping (e.g., formulas, terms, classes, individuals)
- *n* is a degree of trust (confidence) in that mapping
- *R* is the relation associated to a mapping, where *R* identifies the relation holding between *e* and *e'*.

In this work we are not concerned about ontology matching techniques, but on the use of alignments. Thus, we are interested in representing and querying mappings between ontologies. We propose using RDF as the language for expressing

alignments, so that they can be published on the web and queried using SPARQL. In particular, we use the format of the Ontology Alignment Evaluation Initiative[13].

5 Related Work

Some (not many) other efforts have been made trying to align or compare different service description approaches. We set out from existing conceptual comparisons between OWL-S and WSMO [15], OWL-S, SAWSDL and WSDL [7, 13] to obtain a general model description of services that facilitates their discovery.

Giantsiou et al. [5] propose a service meta-model in which found services are transformed and represented in RDF. Their meta-model and discovery approach is influenced by light weight approaches (SAREST [18] and SAWSDL). Differently, we focus on both lightweight and semantic techniques, allowing other description models.

Most of the current approaches to Semantic Web Services matching, particularly those based on OWL-S, are based on subsumption reasoning on concepts included in the descriptions (e.g. [9, 12]). Klusch et. al [6] present a hybrid matchmaker that complements logic based reasoning with approximate matching techniques from Information Retrieval. In this sense we also propose a hybrid approach, which combines subsumption checking, concepts similarity and information retrieval. However, we focus on several different service description models and consider the alignment of domain concepts.

6 Conclusion

In this paper we have dealt with the problem of service discovery in open systems. We proposed an architecture that has semantic alignment as a first citizen component. In particular, we discussed in detail the alignment of service description models, and the transformation of them into an RDF common model. We also proposed the combination of service matching and concept similarity into an integrated service matching framework.

Domain ontologies can be specified using different languages. We address that aspect at the semantic concept matching implementation level, where we consider (for the moment) ontologies written in RDF, OWL and WSML. At the service model mapping level we only need to copy the concept without any reasoning with the ontology.

The implementation and evaluation of the proposed framework is part of our ongoing work. We plan to adapt, transform and extend existing test collection (e.g. the OWL-S TC, SAWSDL TC)[14] as a starting benchmark for our experiments.

[13] http://oaei.ontologymatching.org/

[14] http://www.semwebcentral.org/projects/owls-tc/ or /sawsdl-tc/

7 References

1. Burstein, M., Bussler, C., Zaremba, M., Finin, T., Huhns, M. N., Paolucci, M., Sheth, A. P., and Williams, S. A Semantic Web Services Architecture. *IEEE Internet Computing* 9, 5, 72-81. 2005.
2. Ehrig M. Ontology Alignment: Bridging the Semantic Gap. Springer. 2007.
3. Fernandez, A. et al. Closing the service discovery gap by collaborative tagging and clustering techniques. In 2nd International Joint Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web (SMR2), pages 115–128, Karlsruhe, 10/2008.
4. Fernandez, A., Polleres, A., and Ossowski, S. Towards Fine-grained Service Matchmaking by Using Concept Similarity, ISWC-2007 Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web (SMR2). 2007
5. Giantsiou, L., Loutas, N., Peristeras, V. and Tarabanis, K. Semantic Service Search Engine (S3E): An Approach for Finding Services on the Web. In LNCS, Volume 5736/2009, pages 316-325. Springer, 2009.
6. Klusch, M., Fries, B., and Sycara, K. Automated semantic web service discovery with owls-mx. AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, pages 915–922, NY, 2006. ACM Press.
7. Kourtesis, D., Paraskakis, I. Combining SAWSDL, OWL-DL and UDDI for Semantically Enhanced Web Service Discovery. Springer Berlin / Heidelberg. Pages 614-628. 2008.
8. Leacock, C. and Chodorow, M. Combining local context andWord- Net similarity for word sense identification. In Christiane Fellbaum, editor, WordNet: An Electronic Lexical Database, pages 265–283. MIT Press, 1998.
9. Li, L and Horrocks, I. A software framework for matchmaking based on semantic web technology. Int. J. of Electronic Commerce, 8(4):39–60, 2004.
10. Li, Y., Bandar, Z. and McLean, D. An approach for measuring semantic similarity between words using multiple information sources. IEEE Trans. Knowl. Data Eng., 15(4):871–882, 2003.
11. Paolo Bouquet, Jérôme Euzenat, Enrico Franconi, Luciano Serafini, Giorgos Stamou, and Sergio Tessaris. Specification of a common framework for characterizing alignment. Deliverable D2.2.1, Knowledge web NoE, 2004.
12. Paolucci, M., Kawamura, T., Payne, T., and Sycara, K. Semantic Matching of Web Service Capabilities. In ISWC, pages 333–347. Springer Verlag, 2002.
13. Paolucci, M., Wagner M., Martin M. Grounding OWL-S in SAWSDL. Springer Berlin / Heidelberg. Pages 416-421. 2007.
14. Papathanasiou, M., Loutas, N., Peristeras, V., Tarampanis, K. Combining Service Models, Semantic and Web 2.0 Technologies to Create a Rich Citizen Experience. M.D. Lytras et al. (Eds.): WSKS 2009, LNAI 5736, pp. 296–305, 2009
15. Polleres, A., Lara, R. A Conceptual Comparison between WSMO and OWL-S, WSMO Working Group working draft, 2005. http://www.wsmo.org/2004/d4/d4.1/v0.1/.
16. Rada, R., Mili, H., Bicknell, E., and Blettner, M. Development and application of a metric on semantic nets. IEEE Transactions on Systems, Man and Cybernetics, 17–30, 1989.
17. Salton, G. 1989. Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley.
18. Sheth, A., Gomadam, K., Lathem, J.: SA-REST: Semantically Interoperable and Easier-to-Use Services and Mashups. IEEE Internet Computing 11(6), 91–94 (2007).
19. W3C World Wide Web Consortium. SPARQL Query Language for RDF. W3C Recommendation 15 January 2008. http://www.w3.org/TR/rdf-sparql-query/.
20. Wu, Z. and Palmer, M. Verbs semantics and lexical selection. In Proceedings of the 32nd annual meeting on Association for Computational Linguistics, pages 133–138, Morristown, NJ, USA, 1994. Association for Computational Linguistics.