

# Enabling Web Service Discovery in Heterogeneous Environments

Zijie Cong and Alberto Fernández Gil

Artificial Intelligence Research Group  
Universidad Rey Juan Carlos  
Calle Tulipán s/n, 28933, Móstoles, Spain  
{zijie.cong, alberto.fernandez}@urjc.es

**Abstract.** In large, open environments, service discovery has to face the challenge of heterogeneity. Service advertisements published by different organizations or individuals may differ in their description models, thus expressiveness levels. Even on the same expressiveness level, organizations may use assorted domain ontologies. In addition, a service discovery tool may not have a global view of all service advertisements in the system. Unfortunately, most contemporary service discovery approaches rely on these key factors. This paper presents a method that addresses the mismatch problem in description models. A neutral model is proposed in this work for aligning different service description models. A matchmaking method that encompasses different semantic, syntactic and hybrid service description languages based on this neutral model is also presented. Implementation and evaluation of the proposed method showed a satisfactory result.

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

## 1 Introduction

Service-Oriented Architecture (SOA) has been widely adopted by the industries due to its discoverability, maintainability, reusability and composability. A SOA is usually implemented using Web Services. While there exist various definitions of the term “web service”, most of these definitions agree on four characteristics of web services: i) A web service is a software unit of business functionality; ii) web services provide Application Programming Interfaces (APIs), which are usually described in a standardized language; iii) web services are accessible through standardized network protocols such as HTTP; iv) web services provide interoperable machine-to-machine interaction. Among these characteristics, ii) and iv) are most promising and important for bringing the aforementioned advantages of SOA to practices.

However, with the rapid rise of interest from academia, industries and public, different web service description approaches have been proposed and are commonly seen in practice. Ranging from simple keywords and free text to highly expressive,

semantic enabled description models such as OWL-S [19], SAWSDL [16] and WSMO [7].

The existence of differences among these service description approaches in open environments reflexes the different needs and the capabilities of their providers, e.g. a human agent without expertise in web services usually describe their services in natural languages instead of a structured service description language based on XML; on the other hand, a software agent will prefer to describe its capabilities in a semantic enabled description languages, such as OWL-S and SAWSDL to facilitate the machine-to-machine interaction.

To provide a clear view of the levels of expressiveness, Figure 1 illustrates three levels used in this paper:

- Machine-understandable
- Machine-processable
- Human-understandable

Scenario of Interaction	Expressiveness	Components	Approach
Machine-to-Machine →	Machine - Understandable	Semantic Inputs/Outputs, Preconditions/Effects	OWL-S, WSMO, SAWSDL
Human-to-Machine →	Machine - Processable	Syntactic Inputs/Outputs	WSDL
Human-to-Human →	Human - Understandable	Keywords, Tag-cloud Text	Keywords Tag-cloud Text

Figure 1 Levels of expressiveness and service description approaches

From this figure, we may notice that approaches that address machine-to-machine level communication take into consideration also the lower level communication needs. This coverage provides possibility for service discovery tools to match service descriptions across expressiveness levels with some information loss. Thus the problem lies on the differences among description languages.

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 process. In addition, the performance of the current approach depends on the size of corpus (registered services) to a large extends due to the employment of classic information retrieval techniques (e.g. TF-IDF), which may lead to scalability problem in decentralized systems.

The contributions of this paper are: i) an architecture for service discovery where semantic alignment mechanisms are integrated into, ii) an alignment technique for

different service description models, and iii) a scalable service matchmaking method that uses the aligned services in decentralized, open environments.

This paper is an extended version of [4], including more detailed explanations, an example and extending implementation and empirical evaluation.

The rest of the paper is organized as: Section 2 presents a service discovery architecture with service alignment and enrichment techniques for decentralized, heterogeneous environment. Section 3 details the matchmaking process based-on proposed model. Section 4 presents the implementation of our service discovery system and evaluation results. Section 5 presents related works and finally, section 6 concludes our work.

## 2 Service Discovery Architecture

The architecture of our service directory is depicted in Figure 2. There are two types of agents that interact with the directory, the one who offers the service (*Service Provider*) and the consumer of services (*Service Requester*). As we will see in Section 4, they can access the directory through a REST service or a human-oriented web interface.

Service providers register services in the directory providing the following information:

- *Service Description*: the service description specified by the provider is essential because it will contain all the information related to the service offered (it can include the service category). In our framework, we allow several service description models. They include semantic models (OWL-S, WSMO), syntactic models (WSDL), hybrid (SAWSDL), as well as other lighter approaches (*keyword*, *tag-cloud*, and *text*-based service descriptions).
- *Grounding*: the service provider must attach the information required to access the service by a client (for example a WSDL file).
- *Category* (optional): the category of the service can be explicitly defined in this section according to the NAICS [2] classification. As we will see later, service category is complemented with information provided in the *service description* section, such as explicit annotation (e.g. in some versions of OWL-S) or extracted from a textual description.

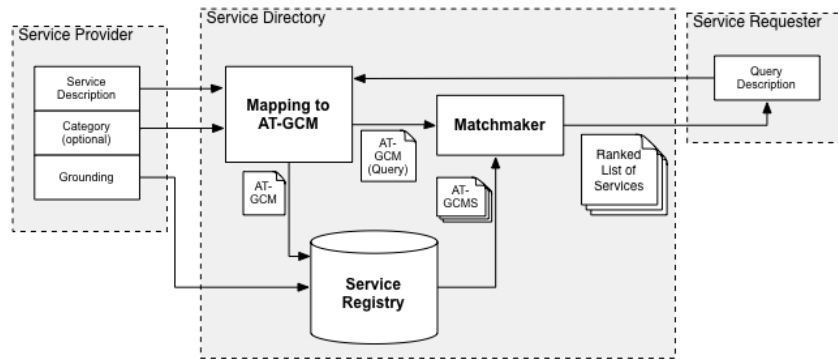


Figure 2 Service Discovery Architecture

Service descriptions and category are combined and converted into a common format (*AT-GCM*) and stored in a *Service Registry*. The common format (section 2.1) comprises the relevant characteristics of the original models, from a service matchmaking point of view. The *Mapping to AT-GCM* module generates the AT-GCM version of the service from the service description and the category.

The *AT-GCM*, the *Grounding*, and the original *Service Description* provided by the Service Provider are stored as an entry in the service registry database.

When client agents (*service requesters*) want to use the service directory for finding a service, they send the necessary information (*Query Description*) to obtain a list of matching services (sorted list by the degree of match with the query). Query descriptions are specified using one of the available description languages. Note that our framework is able to return services described in a different language to the query. For instance, it may return an OWL-S service while the query is specified using WSDL.

When the service directory receives a query description, the query is transformed into the ATM-GCM format (*Mapping to AT-GCM*) and passed to the *Matchmaker*. Then, the matchmaker compares the query against the AT-GCM versions of the services stored in the database and returns a ranked list of services to the client. This process is detailed in section 3.

## 2.1 An unified model for representing service descriptions

Setting out from existing conceptual comparisons between semantic web service descriptions ([26, 18, 23, 17]) and considering lighter approaches too, we obtained a *General Common Model (AT-GCM)*<sup>1</sup> with the following elements: *inputs*, *outputs*, *preconditions*, *effects*, *keywords*, *textual description*, *category* and *tag cloud*.

Detailed description about the model and the mappings from original models to the *AT-GCM* can be found in [9]. Here we summarise that description.

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

- $\mathcal{I}_{GCM} = \langle \mathcal{I}_{syn}, \mathcal{I}_{sem} \rangle$  is the pair of sets of syntactic ( $\mathcal{I}_{syn} \in \{a, \dots, z\}^*$ ) and semantic ( $\mathcal{I}_{sem} \subseteq \mathcal{N}$ ) inputs of the service.
- $\mathcal{O}_{GCM} = \langle \mathcal{O}_{syn}, \mathcal{O}_{sem} \rangle$  is the pair of sets of syntactic ( $\mathcal{O}_{syn} \in \{a, \dots, z\}^*$ ) and semantic ( $\mathcal{O}_{sem} \subseteq \mathcal{N}$ ) outputs.
- $\mathcal{P}_{GCM}$  is the set of preconditions.  $\mathcal{P}_{GCM} \subseteq \mathcal{N}$
- $\mathcal{E}_{GCM}$  is the set of effects.  $\mathcal{E}_{GCM} \subseteq \mathcal{N}$
- $\mathcal{K}_{GCM} = \langle \mathcal{K}_{syn}, \mathcal{K}_{sem} \rangle$  is the pair of sets of syntactic and semantic keywords, where  $\mathcal{K}_{syn} \subseteq \{a, \dots, z\}^*$ ,  $\mathcal{K}_{sem} \subseteq \mathcal{N}$ .

---

<sup>1</sup> *AT* stands for *Agreement Technologies*, meaning agreement among different service description models. It is also the name of one of our funding projects (CSD2007-0022).

- $\mathcal{C}_{GCM}$  is a set of categories of the service, described semantically ( $\mathcal{C}_{sem} \subseteq \mathcal{N}$ ) (e.g. NAICS or UNSPSC).
- $\mathcal{T}_{GCM}$  is a textual description of the service.
- $\mathcal{TC}_{GCM}$  is a tag cloud.  $\mathcal{TC}_{GCM} = \{ \langle t, n \rangle \mid t \in \{a, \dots, z\}^*, n \in \mathbb{N} \}$ .

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

<i>GCM</i>	OWL-S / WSMO	SAWSDL	WSDL	Keyword (tag)	Tag Cloud	Text
$\mathcal{I}_{GCM}$	$\langle \emptyset, \text{pt}(\mathcal{I}) \rangle$	$\langle \mathcal{I}_{syn}, \mathcal{I}_{sem} \rangle$	$\langle \mathcal{I}, \emptyset \rangle$	$\langle \emptyset, \emptyset \rangle$	$\langle \emptyset, \emptyset \rangle$	$\langle \emptyset, \emptyset \rangle$
$\mathcal{O}_{GC}_M$	$\langle \emptyset, \text{pt}(\mathcal{O}) \rangle$	$\langle \mathcal{O}_{syn}, \mathcal{O}_{sem} \rangle$	$\langle \mathcal{O}, \emptyset \rangle$	$\langle \emptyset, \emptyset \rangle$	$\langle \emptyset, \emptyset \rangle$	$\langle \emptyset, \emptyset \rangle$
$\mathcal{P}_{GC}_M$	$\mathcal{P}$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$\mathcal{E}_{GCM}$	$\mathcal{E}$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$\mathcal{C}_{GCM}$	$\mathcal{C}$	$Cat(\mathcal{T})$	$Cat(\mathcal{T})$	$Cat(\mathcal{T})$	$Cat(\mathcal{T})$	$Cat(\mathcal{T})$
$\mathcal{T}_{GCM}$	$\mathcal{T}$	$\mathcal{T}$	$\mathcal{T}$	$\emptyset$	$\emptyset$	$S$
$\mathcal{TC}_{GC}_M$	$\Delta(\mathcal{T}) \cup \mathcal{N}(\mathcal{I}) \cup \mathcal{N}(\mathcal{O})$	$\Delta(\mathcal{T}) \cup \mathcal{I}_{syn} \cup \mathcal{O}_{syn}$	$\Delta(\mathcal{T}) \cup \mathcal{I} \cup \mathcal{O}$	$\{ \langle t, 1 \rangle \mid t \in \mathcal{K}_{syn} \}$	$S$	$\Delta(S)$
$\mathcal{K}_{GC}_M$	$\langle \tau(\Delta(\mathcal{T})) \cup \mathcal{N}(\mathcal{I}) \cup \mathcal{N}(\mathcal{O}), \text{pt}(\mathcal{I}) \cup \text{pt}(\mathcal{O}) \rangle$	$\langle \tau(\Delta(\mathcal{T})) \cup \mathcal{I}_{syn} \cup \mathcal{O}_{syn}, \mathcal{N}(\mathcal{I}_{sem}) \cup \mathcal{N}(\mathcal{O}_{sem}) \rangle$	$\langle \tau(\Delta(\mathcal{T})) \cup \mathcal{I}_{syn} \cup \mathcal{O}_{syn}, \phi \rangle$	$\mathcal{K}$	$\langle \tau(S), \emptyset \rangle$	$\langle \tau(\Delta(S)), \emptyset \rangle$

Table 1 Mapping details for each component in AT-GCM

There are many straightforward mappings that consist of simple associations between parameters in both models. For instance, in OWLS/WSMO  $\mathcal{I}_{GCM} = \langle \emptyset, \text{pt}(\mathcal{I}) \rangle$  because they only provide semantically described inputs  $\mathcal{I}$  ( $\mathcal{I}_{sem}$ ), where  $\text{pt}(\mathcal{I}) = \{ t \mid t = \text{parameterType}(i) \ \forall i \in \mathcal{I} \}$ .

However, some fields (e.g. 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 (section 2.2.1). Syntactic keywords can be easily obtained from tag clouds (either original or calculated with  $\Delta$ ) with their frequency set to 1 (function  $\tau$ ). The set of input and output concept names as well as their parameter types ( $\text{pt}(\mathcal{I})$  and  $\text{pt}(\mathcal{O})$ ) are

also adopted as syntactic and semantic keywords, respectively. Function  $Cat(T)$  discovers an appropriate category based on  $T$  (section 2.2.2). Function  $N(I/O)$  retrieves the syntactic name of an input or output, e.g. URI fragment of an semantic input.

Figure 3 summarises the characteristics of the *AT-GCM* that can be obtained from each original service description model.

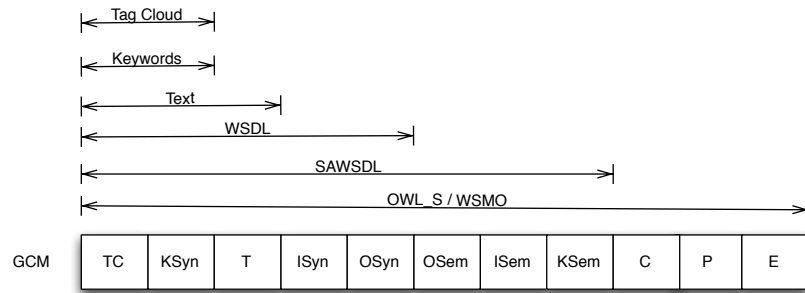


Figure 3 AT-GCM characteristics covered by service description models

## 2.2 Model enrichment

Useful information about services may not always be explicitly defined by the providers in their original service descriptions. Such information could, however, be discovered from other elements in the description and/or by using external resources. In this section, we briefly introduce the enrichment of *AT-GCM* using existing elements and external resources.

A complete schema is shown in Figure 4.

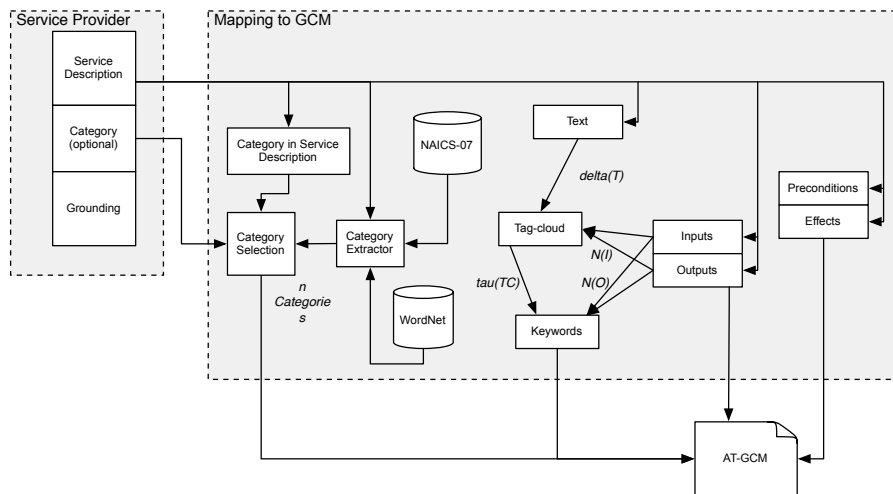


Figure 4 Mapping to GCM

### 2.2.1 Extracting syntactic information

To extract keywords from a textual service description, if user-provided keywords are presented in service descriptions, i.e. training examples exist; supervised-learning algorithms can be employed to find keywords automatically. Most related techniques, for example [25], uses TF-IDF for feature selection and Naïve Bayes for establishing model.

In the current stage, our approach first tokenizes the textual description. Instead of breaking the text using spaces and punctuation marks, word segmentation is performed to tokenize text into logical lexical units. For example, text description:

*“This service returns the size of the Air Force of a given country.”*

will be tokenized into (after filtering):

*{“size”, “Air Force”, “country”}*

Then we filter out *stopwords* that include conjugations, prepositions, articles and some web service related words, e.g. service, information, give, provide etc. We assume that nouns and verbs usually carry more information than adjectives and adverbs. A third party lexical database, *WordNet*, is used to determine the part-of-speech of each word. Adjectives and adverbs are filtered out.

The extracted keywords are then lemmatized using WordNet. Comparing to other popular stemming algorithms such as Porter’s [10], [28] stemming algorithm, WordNet significantly reduces false positive results.

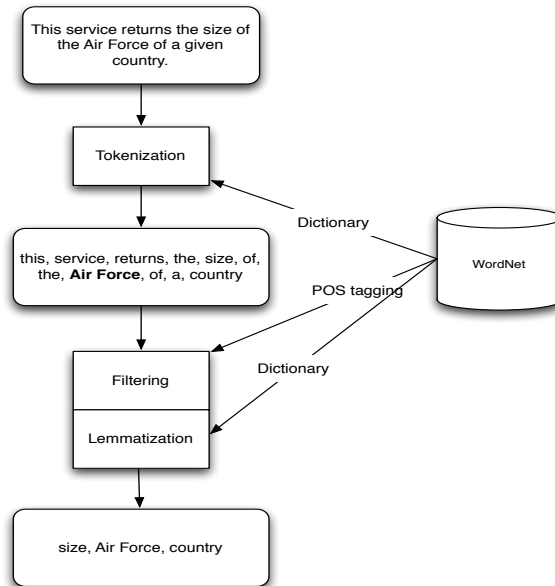


Figure 5 Keywords extraction based on WordNet

Figure 5 demonstrates the whole keywords extraction process.

In structured web service descriptions, such as WSDL, SAWSDL and OWL-S, where inputs and outputs of web services are also specified, the set of input and output concept parameter types (pt(I) and pt(O)) are converted into syntactic IOs by extracting their URIs' *fragments*. For example if a semantic input is an individual of concept: *http://www.ia.urjc.es/ontologies/Transportation.owl#Car*, then the fragment *Car* is extracted as a syntactic input. Then syntactic IOs will also be added into tag-clouds.

### 2.2.2 Category Discovery

Our directory is organized using service's category information based on the North American Industry Classification System (NAICS) [2]. Services need to provide at least one NAICS category to be registered in our directory.

Among all service description languages considered by our directory, only OWL-S provides a mechanism to include NAICS category information in the service description, but also commonly ignored by service providers.

To automatically categorize a web service according to their industrial sectors, various approaches have been proposed. Most existing approaches [3, 5, 11, 13, 21] consider service categorization as a text classification problem.

Classic classifiers such as Support Vector Machine [27] and Naive Bayesian [20] are employed by these works to classify web service using keywords extracted from textual service description, Input/Outputs and other components.

As most of these classifiers perform supervised learning, training examples must be provided a priori, these training examples are usually obtained manually. This requirement may not always be satisfied because, a) standard service categorization taxonomies are usually enormous, e.g. NAICS 2007 has approximated 24,000 categories, it is highly unlikely to provide sufficient training examples that covers all categories; b) multiple taxonomies may exist within a registry, once a new taxonomy is presented, new training examples must be provided to train the classifier.

[5] uses also information from functional components in structured service description such as WSDL. It assumes that services under same category have similar inputs and outputs, which is a strong assumption for industrial-sector-oriented categorization systems.

To associate an appropriate category with the service, we first extract keywords related to each category from NAICS 2007 Index file. During each service registration, if no category information is provided by the service provider nor defined in the service description, *category extractor* calculates the similarity between keywords extracted from service description and keywords of each NAICS 2007 category to find the most suitable categories for the service.



The similarity is measured by mapping each keyword from both NAICS categories and service description to WordNet synsets and domain category relations, and the similarity is defined as:

$$\frac{|K_S \cap k_c|}{|k_c|}$$

where  $K_S$  denotes the keywords extracted from service description  $S$ , and  $k_c$  denotes sets of keywords of each NAICS 2007 category  $c$ .

### 2.3 Example

To demonstrate the proposed mapping process, this section shows our proposal with an example of a university researcher service taken from the OWL-S test collection. This service returns information of research in a given university. We present an OWL-S representation of that service, as well as their mapping to the *GCM*.

A segment of original service definition is presented as following:

```
<profile:textDescription xml:lang="en">
  This service returns researcher of a university
</profile:textDescription>

<profile:hasInput  rdf:resource="#_UNIVERSITY"/>
<profile:hasOutput rdf:resource="#_RESEARCHER"/>

<process:Input  rdf:ID="_UNIVERSITY">
  <process:parameterType
    rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">http://127.0.0.1/ontology/portal.owl#University</process:parameterType>
</process:Input>

<process:Output  rdf:ID="_RESEARCHER">
  <process:parameterType
    rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">http://127.0.0.1/ontology/portal.owl#Researcher</process:parameterType>
</process:Output>
```

We define formally the OWL-S university researcher example service  $S = \langle \mathcal{I}, \mathcal{O}, \mathcal{P}, \mathcal{E}, \mathcal{C}, \mathcal{T} \rangle$  as follows:

```
 $\mathcal{I} = \{\text{npr}^2\text{:\_UNIVERSITY}\}$ 
 $\mathcal{O} = \{\text{npr}\text{:\_RESEARCHER}\}$ 
 $\mathcal{P} = \emptyset$ 
 $\mathcal{E} = \emptyset$ 
 $\mathcal{C} = \emptyset$ 
 $\mathcal{T} = \text{"This service returns researchers in university"}$ 
```

---

<sup>2</sup> npr: [http://\(OWLS-TC URI\)/services/1.1/university\\_researcher-service.owl](http://(OWLS-TC URI)/services/1.1/university_researcher-service.owl)

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/outputs, and the semantic inputs would be the semantic concepts (parameter types) of inputs  $\mathcal{I}$ . 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 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:

$$\begin{aligned}
S_{GCM} &= \langle \mathcal{I}_{GCM}, \mathcal{O}_{GCM}, \mathcal{P}_{GCM}, \mathcal{E}_{GCM}, \mathcal{K}_{GCM}, \mathcal{C}_{GCM}, \mathcal{T}_{GCM}, \mathcal{T}_{GCM} \rangle \text{ where:} \\
\mathcal{I}_{GCM} &= \langle \{\text{university}\}, \{\text{portal}^3:\text{University}\} \rangle \\
\mathcal{O}_{GCM} &= \langle \{\text{researcher}\}, \{\text{portal}:\text{Researcher}\} \rangle \\
\mathcal{P}_{GCM} &= \emptyset \\
\mathcal{E}_{GCM} &= \emptyset \\
\mathcal{K}_{GCM} &= \langle \{\text{university}, \text{researcher}\}, \{\text{portal}:\text{University}, \text{portal}:\text{Researcher}\} \rangle \\
\mathcal{C}_{GCM} &= \emptyset \\
\mathcal{T}_{GCM} &= \text{"This service returns researchers in university"} \\
\mathcal{T}_{GCM} &= \{ \langle \text{university}, 2 \rangle, \langle \text{researcher}, 2 \rangle \}
\end{aligned}$$

### 3 Service Matchmaking

Service matchmaking is an essential part of our service directory. The similarity between two service descriptions (request and advertisement) is based on the similarities of each pair of corresponding elements in their AT-GCMs.

We further classify the elements in AT-CGM into three categories: semantic elements, syntactical elements and category information. Each type of element is associated with an ontology, and a generic ontological similarity algorithm is applied to calculate the similarity between each pair of corresponding elements of service request ( $S_R$ ) and advertisement ( $S_A$ ).

- Semantic elements are associated directly with their original ontologies used in the service description.
- Syntactic information is associated with external lexical databases such as WordNet, which can also be considered as ontology.
- The category of a service is often an element in certain classification systems, such elements usually organized in a hierarchy, which can be considered as ontology also.

Table 2 summarizes the AT-GCM components in each categories and the associated ontology:

---

<sup>3</sup> portal: [http://\(OWL-TC URI\)/ontology/portal.owl](http://(OWL-TC URI)/ontology/portal.owl)

Category	Component	Ontology
Semantic Elements	$I_{sem}, O_{sem}, K_{sem}$	[From service description]
Syntactic Elements	$K_{syn}, I_{syn}, O_{syn}, TC$	WordNet
Category Information	$C$	NAICS-07

Table 2 GCM components and ontologies

Figure 6 illustrates the complete matching schema.

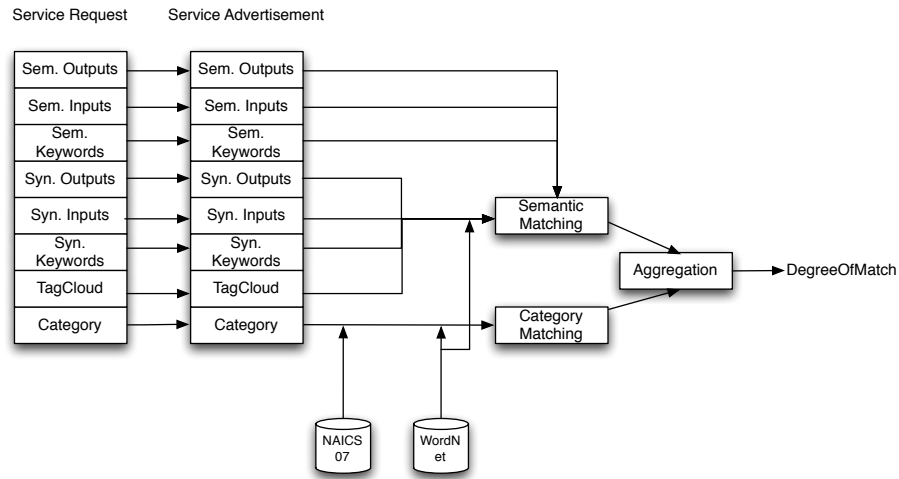


Figure 6 Service Matchmaking Process

### 3.1 Semantic Elements Matchmaking

Semantic elements in AT-GCMs include semantic inputs, semantic outputs and semantic keywords. For instance, in an AT-GCM obtained from an OWL-S description, the semantic elements are  $I_{sem}$ ,  $O_{sem}$  and  $K_{sem}=(I_{sem} \cup O_{sem})$ .

The matching process of semantic concepts in web services takes one concept from service request ( $C_R$ ) and service advertisement ( $C_A$ ) and returns their degree of match.

The degree of match between these semantic concepts is based on their subsumption relation in the ontology. In this paper, we adopt the four degrees of

match proposed by Paolucci et al. in [22]: *exact* ( $C_A = C_R$ ), *plug-in* ( $C_R$  subsumes  $C_A$ ), *subsumes* ( $C_A$  subsumes  $C_R$ ) and *fail* (otherwise).

To obtain a numerical similarity between two concepts, we further calculate the length of the *shortest ancestral path* between these two concepts:

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

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.

With this function, we define the degree of match between concepts,  $C_A$  and  $C_R$  as:

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

### 3.1.1 Semantic Outputs/Inputs

In line with Paolucci's proposal in [22], a semantic 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.

For two sets of semantic outputs,  $O_{sem}^R$  and  $O_{sem}^A$ , the similarity between these two outputs is calculated using function:

$$OSemMatch(O_{sem}^R, O_{sem}^A) = \begin{cases} 1, & \text{if } |O_{sem}^R| = 0 \\ \min_{o^R \in O_{sem}^R} \max_{o^A \in O_{sem}^A} (conceptMatch(o^R, o^A)), & \text{otherwise} \end{cases}$$

In function *OSemMatch*  $O^R$  denotes the semantic outputs from service request. Therefore, if the service request requires no outputs ( $|O_{sem}^R|=0$ ), it returns 1, exact match, regardless of the outputs produced by service advertisement  $O_{sem}^A$ . Otherwise, the semantic match is obtained by taking, for each output in the request, the best match against the ones in the advertisement. The worst case (minimum value) is then chosen to combine the best matches.

For semantic inputs, an analogous approach is followed, but with the order of the request and advertisement reversed.

### 3.1.2 Semantic Keywords

For semantic keywords from service request,  $K_{\text{sem}}^R$  and from service advertisement,  $K_{\text{sem}}^A$  the degree of match between two sets of semantic keywords is calculated using measure proposed in [8]:

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

with  $r = (sim(r, r_1), sim(r, r_2), \dots, sim(r, a_1), sim(r, a_2), \dots)$ , and  $a$  analogously.

## 3.2 Syntactic Elements Matching

Syntactic elements in AT-GCM include syntactic keywords, tag-cloud, syntactic I/Os and text. To achieve uniformity and simplicity, we would like to adopt the similarity measures defined in the last section to suit the syntactic elements too.

However, these elements have no associated ontological concepts explicitly defined in the service description, thus these elements need to be mapped into concepts of a certain lexical database with subsumption relation defined, such as WordNet.

Beyond a simple English dictionary, WordNet groups English words into sets of synonyms called *synsets*, with various semantic relations between these synsets. These semantic relations include *hyponym*, *hypernym*, *domain*, *cause*, *member*, *holonym*, *meronym* similar, *antonym*, *instance* etc. With these semantic relations, WordNet can be considered as an ontology.

There are several benefits of employing a universal lexicon in service discovery. The first and most important reason is that when matching services with no semantic information available, such as WSDL, lexicon may provide extra semantic information base on natural language description, even though limited. Also, when domain ontologies differ, this independent lexicon may perform as a universal ontology minimizes the possibility of mismatch. Finally, similar to domain ontology semantic matching, lexicon-based semantic matching will not be affected by the size of corpus.

WordNet also helps the effectiveness of syntactic matching, e.g. *U.S.* and *United States of America* may be considered as different terms in pure syntactic matching algorithms, but synonyms in WordNet.

### 3.2.1 Syntactic Keywords

Syntactic keywords are first mapped to WordNet synsets, with *hypernym/hyponym* relations defined between synsets, we simply adopt function  $KSemMatch$  defined in the last section:

$$KSynMatch(R_{syn}, A_{syn})_{WordNet} = \frac{\sum_{r \in R} \delta_r \vec{r}}{|\sum_{r \in R} \delta_r \vec{r}|} \cdot \frac{\sum_{a \in A} \delta_a \vec{a}}{|\sum_{a \in A} \delta_a \vec{a}|}$$

where  $K_{\text{synsets}}^R$  and  $K_{\text{synsets}}^A$  denote WordNet synsets associated with keywords in the service request and service advertisement respectively.

Similarity between tag-clouds is calculated in the same way with weights (frequencies):

$$\text{TagMatch}(R_{\text{syn}}, A_{\text{syn}})_{\text{WordNet}} = \frac{\sum_{r \in R} \delta_r \vec{r}}{|\sum_{r \in R} \delta_r \vec{r}|} \cdot \frac{\sum_{a \in A} \delta_a \vec{a}}{|\sum_{a \in A} \delta_a \vec{a}|}$$

where  $\delta_r$  and  $\delta_a$  denotes the frequency of term  $r$  (in  $R$ ) and  $a$  (in  $A$ ) respectively.

### 3.2.2 Syntactic Inputs/Outputs

Degree of match of WordNet synsets mapped from syntactic inputs and outputs are calculated in the same way as their semantic counterparts.

$$\text{OSynMatch}(O_{\text{syn}}^R, O_{\text{syn}}^A)_{\text{WordNet}} = \begin{cases} 1, & \text{if } |O_{\text{syn}}^R| = 0 \\ \text{Min}_{o^R \in O_{\text{syn}}^R} \text{Max}_{o^A \in O_{\text{syn}}^A} (\text{conceptMatch}(o^R, o^A)), & \text{otherwise} \end{cases}$$

### 3.3 Category Matching

As stated in section 2, our directory uses NAICS 07 as services categorization standard. With 2341 categories in total, NAICS 07 standard organizes these categories in a 5-level hierarchy.

Each category is considered as a concept in this category taxonomy, the calculation of the similarity between two categories is done by using:

$$\text{CatMatch}(C_1, C_2) = \text{sim}_{\text{NAICS-07}}(C_1, C_2)$$

### 3.4 Aggregation Function

Finally, service matching must combine the similarity value for each of these fields.

$\text{sim}_{\text{Isyn}}$ ,  $\text{sim}_{\text{Isem}}$ ,  $\text{sim}_{\text{Osem}}$ ,  $\text{sim}_{\text{Osyn}}$ ,  $\text{sim}_{\text{TC}}$ ,  $\text{sim}_{\text{Ksyn}}$ ,  $\text{sim}_{\text{Ksem}}$ ,  $\text{sim}_{\text{C}}$  denote the similarity of syntactic/semantic inputs, syntactic/semantic outputs, tag-cloud, syntactic/semantic keywords and category respectively between a service request and a service advertisement. An aggregation function is a function that combines these similarity values.

For the moment, a general approach is taken: a weighted sum of each similarity, where the weighting parameters are the contribution of the corresponding components of the AT-GCM. The contribution of each component is calculated using a logistic function:

$$w(n_c) = \frac{1}{(1 + e^{\frac{n_c}{0.5\bar{N}}})}$$

where  $n_c$  denotes the number of elements in component C (for example, number of semantic outputs), and  $\bar{N}$  denotes the average number of elements in both service models.

Function  $w$  is a logistic function, which makes the weights of the components with number of elements close to the average increase rapidly. Also, logistic function prevents the over-influence caused by components with excessive number of elements.

## 4 Implementation and Evaluation

In this section we describe the implementation and evaluation of the proposed approach.

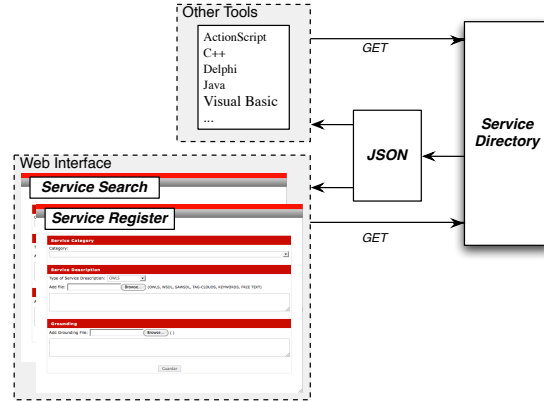


Figure 7 Service Directory Interaction

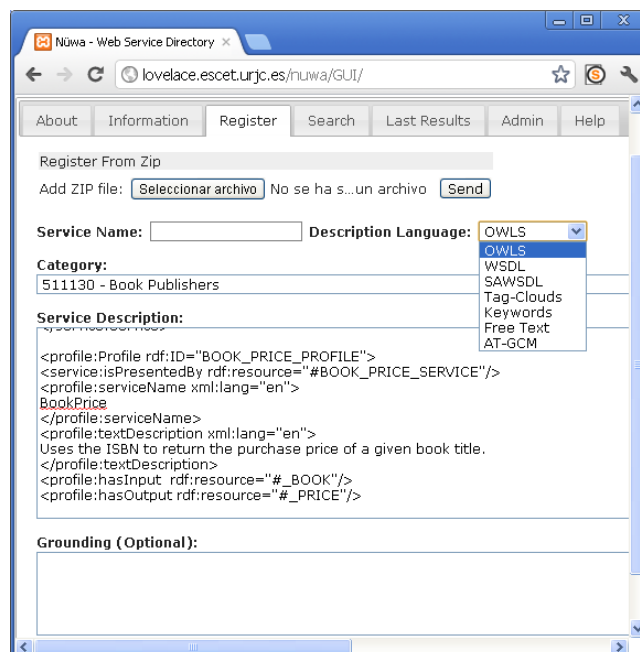
### 4.1 Directory implementation

The proposed framework has been implemented as a web server that performs two main operations: register and search services. Moreover, information about the content of the directory (e.g. number of categories of registered services) and NAICS category list can also be obtained. The information about the content is the number of services registered belonging to each category. That information gives an idea of the specialization of a particular directory. It can be seen as a category-cloud description of the directory, which might be used by an agent to select among a set of candidate directories for registering and/or searching services.

We used SQLite<sup>4</sup> database to facilitate the implementation in future distributions of the service directory.

There are two types of agents that interact with the directory, service providers and the service requesters. They can access the directory through a REST service that can be implemented in different programming languages (Java, PHP, etc.).

The service directory receives search requests and responds to them through JSON [6] data exchange, including a list of descriptions of the matching services and their corresponding grounding so that they can be invoked if desired. When the directory receives a client request (GET) it carries out the operation using the specific parameters included in the request and answers using JSON objects. The client can use the received information to show it or invoke the services.



The screenshot shows a web browser window titled "Nüwa - Web Service Directory" with the URL "lovelace.escet.urjc.es/nüwa/GUI/". The page has a navigation bar with tabs: About, Information, Register, Search, Last Results, Admin, and Help. The "Register" tab is active. The form includes a "Register From Zip" section with an "Add ZIP file:" label, a "Seleccionar archivo" button, a text input field containing "No se ha s...un archivo", and a "Send" button. Below this is the "Service Name:" label with an empty text input field. To the right is the "Description Language:" label with a dropdown menu showing "OWLS" as the selected option. Other options in the dropdown include WSDL, SAWSDL, Tag-Clouds, Keywords, Free Text, and AT-GCM. The "Category:" label is followed by a text input field containing "511130 - Book Publishers". The "Service Description:" label is followed by a text area containing XML code: 

```
<profile:Profile rdf:ID="BOOK_PRICE_PROFILE">
<service:isPresentedBy rdf:resource="#BOOK_PRICE_SERVICE"/>
<profile:serviceName xml:lang="en">
BookPrice
</profile:serviceName>
<profile:textDescription xml:lang="en">
Uses the ISBN to return the purchase price of a given book title.
</profile:textDescription>
<profile:hasInput rdf:resource="#_BOOK"/>
<profile:hasOutput rdf:resource="#_PRICE"/>
</profile:Profile>
```

 At the bottom is the "Grounding (Optional):" label followed by an empty text area.

Figure 8 Service registration

In addition, we have implemented a Web Interface that helps human users to work with the directory. It also uses the REST to interact with the service directory. Figure 7 shows the interaction of our proposed service directory with the Web Interface and other languages.

---

<sup>4</sup> <http://www.sqlite.org/>



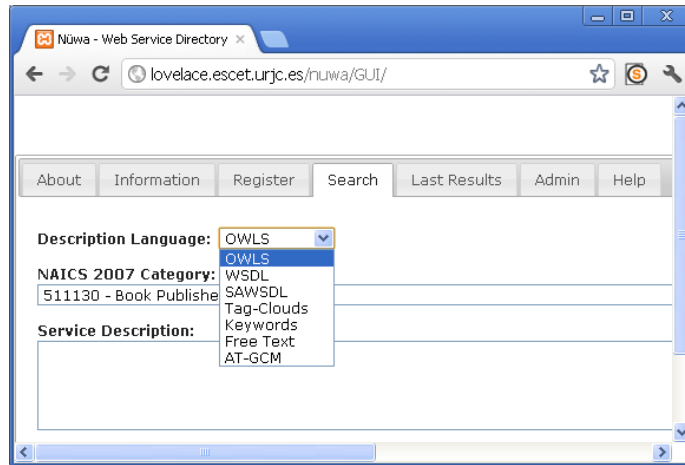


Figure 9 Service search

The Web Interface has several tabs providing different functionality. The information tab shows the number of services registered belonging to each category.

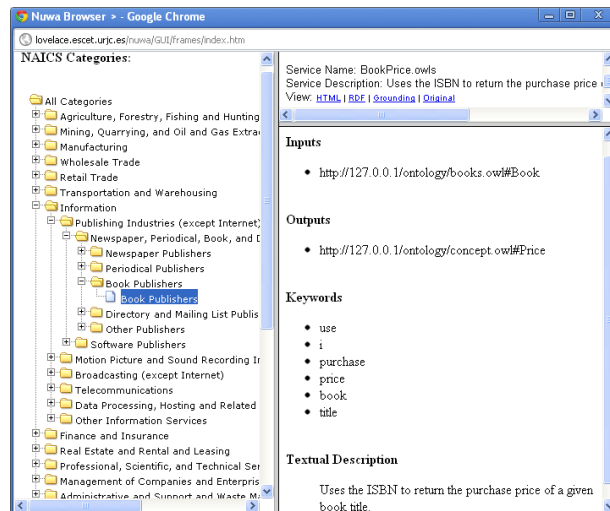


Figure 10 Category tree browsing

Figure 8 shows the service registration tab. Services can be registered individually or zipped for adding a set of services from files. For individual registrations, the user can fill a form where the description language and the service NAICS category can be chosen from a combo boxes. The user must provide a service name and a service description using the language selected. Optionally, the user can also supply a

grounding, which is not processed by internally the directory but only stored for its retrieval.

There are two ways of carrying out a service search (search tab). One is by specifying a service request, which can be done choosing a service category, writing a service query in one of the available languages or both (see Figure 9). An alternative way is by browsing the category tree (See Figure 10).

## 4.2 Evaluation of Matchmaking

To evaluate the performance of our matchmaker, we used a public test collection OWL-S TC 4.0. This test collection consists of 1083 service advertisements and 42 queries, relevance information are provided based on human judgment.

Two performance measures, *precision* and *recall* were collected. Precision and recall are two commonly used measures for measurement of IR effectiveness. Given a query  $q$ , precision  $P$  is the proportion of the relevant services retrieved by the matchmaker to all the retrieved services, and is described as:

$$p = \frac{|relevant\_services \cap retrieved\_services|}{|retrieved\_services|}$$

Recall  $R$  is the proportion of relevant services, which have been retrieved to all the relevant services, and is described as

$$r = \frac{|relevant\_services \cap retrieved\_services|}{|relevant\_services|}$$

Detailed description of test-collection and evaluation measures can be found in the OWLS-TC Manual<sup>5</sup>.

Figure 11 illustrates the precision and recall of 5 different strategies based on AT-GCM:

1. *Semantic I/O Logical* is a classical strategy based on semantic I/Os' subsumption relation defined in domain ontology, widely used by semantic service matchmakers, e.g. OWLS-MX [15], iSeM [14]. This strategy works on service descriptions with semantic inputs and outputs such as OWL-S, WSMO and SAWSDL.
2. *WordNet I/O Logical* strategy uses syntactic I/Os' subsumption relation found in WordNet lexicon. This strategy works on service descriptions with syntactic inputs and outputs, such as OWL-S, WSMO, SAWSDL and WSDL.
3. *WordNet I/O+Tag* strategy, in addition to pure WordNet I/O logical matching, takes into consideration also the tag-cloud of a service. The ranking is based on the Cosine value of the semantic similarity (using path distance) vector of all terms in tag-cloud.

---

<sup>5</sup> <http://projects.semwebcentral.org/projects/owl-s-tc/>

4. *TF-IDF Cosine (WordNet Unfold)* uses unfolded WordNet hierarchies instead of domain ontology. This strategy does not rely on domain ontologies. It will be also affected by the statistical characteristics of the term and test collection.
5. *TF-IDF Cosine (Unfolded Domain Ontology)* strategy is a technique in IR field adopted by many matchmakers e.g. OWLS-MX, iSeM. This strategy unfolds concept hierarchy of the semantic I/O concepts, and then performs TF-IDF ranking function to determine the degree of match of two services. This strategy relies on the existence of semantic information and statistical characteristics of the term.

Among this five strategies, 1, 2 and 3 are *single-matching strategies*, which relies only on two services engaged in matching, with a third-party lexicon (in this case *WordNet*). Strategies 2 and 3 are particular suitable for dealing with heterogeneity in open environments, since the information required are usually available.

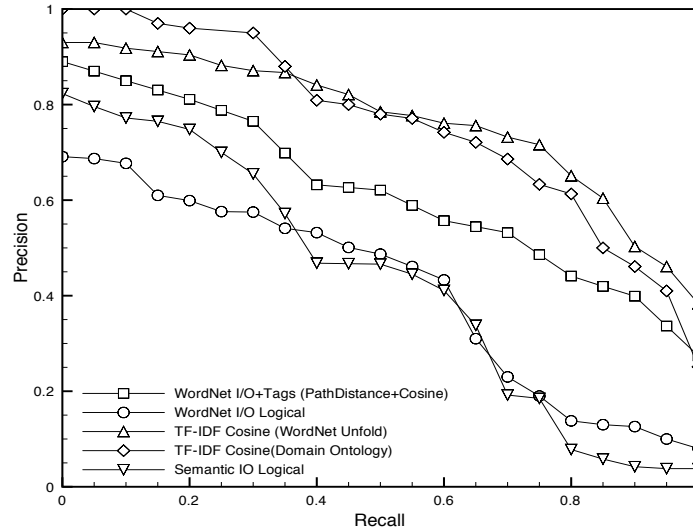


Figure 11 Precision vs. Recall of Matchmakers

Strategy 3, WordNet-based I/O + Tag matching gives an average precision around 0.771 at recall level 0.2, comparable to most contemporary matchmakers' performance based on same test collection, e.g. iSeM, around 0.82, makes this strategy practical in real-world uses.

Since TF-IDF ranking require a global visibility of all service advertisements in the system to produce satisfying results, the last two strategies may not be suitable in decentralized systems. In addition, strategy 5 requires also the domain ontology, which fails for syntactic service descriptions.

Comparing semantic with syntactic information, strategy 2 (WordNet I/O) behaves similar to strategy 1 (Semantic I/O subsumption) at higher recall level. Even

though, at lower recall level, WordNet I/O subsumption approach performs worse than the semantic subsumption approach, it provides extra ability to match service advertisements without explicit semantic annotations. Strategy 4 and 5 perform generally better than other strategies since the *unfolding* exploits more information in the ontology (including WordNet, which can be considered as a global ontology) than shallow I/O concept subsumption matching.

## 5 Related Work

Some other efforts have been made trying to align or compare different service description approaches. As we mentioned in Section 2, we set out from existing conceptual comparisons between popular semantic web service languages [19, 16, 7] to obtain a general model description of services that facilitates their discovery.

Giantsiou et al. [12] 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 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.[15, 14]. Klusch et. al [15] present a hybrid matchmaker that complements logic based reasoning with approximate matching techniques from Information Retrieval. In this sense we propose a hybrid approach, which combines subsumption checking, concepts similarity, and information retrieval. However, we focus on the integration of several different service descriptions.

The directory service using a common model (AT-GCM) in the same direction as iServe [24] uses the minimum service model to address interoperability, the difference is that our board to consider Tag-Cloud, and keywords free text for use in the directory.

Ambite et al introduced a system (DEIMOS) for constructing semantic web service from online sources automatically in [1]. DEIMOS uses an existing semantic web service as a seed, by calculating the syntactic similarity and a brute-force invocation-observation learning process, DEIMOS semantically annotated an external source. Differently to our approach they use only inputs/outputs to characterise services. Also, they use the Local-As-View (LAV) datalog rules to describe the sources. We use RDF instead, although this does not reduce expressivity against LAV, in fact DEIMOS generates an RDF graph from LAV descriptions.

In addition, A. Hess introduced a web service classification approach using machine-learning techniques in [13]. Even though the evaluation showed a remarkable accuracy, no information about computational efficiency was shown. As techniques such as Naïve-Bayes and SVM could be noticeably computationally expensive, this approach might not be entirely suitable for service discovery in a large, open environment.

## 6 Conclusion

In this paper, problems of heterogeneity in open-environment service discovery are identified: 1) the differences in expressiveness levels of service description 2) service description approaches and 3) domain ontology diversity. To address these challenges, an architecture that has semantic alignment as a first citizen component is proposed. In particular, we discussed in detail the alignment of service description models, and the transformation of them into a unified common model. In this paper, we provide alignment mechanism for a set of common service description languages while other languages can be easily integrated into. In fact, if such new model fits into the proposed *AT-GCM* only the adequate mappings have to be specified. Otherwise, new characteristics might be added to the *AT-GCM* to account for those new languages, and considering them empty for the previous models (additionally those legacy models might be completed with the corresponding mappings to the new characteristics). Matchmaking techniques dedicate to specific components in *AT-GCM* are also proposed. In addition to common semantic subsumption matching, our system employs third party lexicon, WordNet, bringing semantic matching ability to syntactic information.

Regarding computational aspects, note that the mapping of service advertisements to the *AT-GCM* can be done at registration time, so we only need to process the service request at run time (as well as the matchmaking algorithm). We also proposed the combination of service matching and concept similarity into an integrated service-matching framework.

The proposed framework has been implemented and both machine and human interfaces are available. We carried out a matchmaking performance evaluation using a well-known semantic test collection (OWLS-TC 4.0). The evaluation showed a satisfying result in open decentralized environment. In particular, the results showed that the proposed approach to enriching syntactic descriptions obtained performances close to semantic ones. This is an important result towards the unified framework integrating descriptions at different level of expressiveness as is the aim of this work. Also, note that experiments have been carried out using a test collection specific for one semantic description model (OWL-S). Our future work includes experimenting with a larger unified test collection including services written in different languages. In such a setting, we expect our framework to perform better than others (which are usually focused in only on type of descriptions).

**Acknowledgment.** Work partially supported by the Spanish Ministry of Science and Innovation through grants TIN2009-13839-C03-02 (cofunded by Plan E) and CSD2007-0022.

## 7 References

1. Ambite, J., Darbha, S., Goel, A., Knoblock, C., Lerman, K., Parundekar, R. and Russ, T. "Automatically constructing semantic web services from online sources." *The Semantic Web-ISWC 2009* (2009): 17-32.
2. Ambler, C. A., and Kristoff, J. E. "Introducing the North American industry classification system." *Government Information Quarterly* 15, no. 3 (1998): 263-273.
3. Bruno, M., Canfora, G., Di Penta, M. and Scognamiglio, R. "An approach to support web service classification and annotation." In *Proceedings of the IEEE International Conference on e-Technology, e-Commerce and e-Service (2005)*, pp. 138-143.
4. Cong, Z., Fernandez, A. and Soto, C. "A Directory of Heterogeneous Services", *Fourth International Workshop on REsource Discovery (RED@ESWC2011)*, CEUR Workshop Proceedings vol. 737, pp. 65-79, (2011)
5. Corella, M. A. and Castells, P.. "Taxonomy-Based Web service categorization using conceptual parameter descriptions." In *Proceedings of the 1st International Workshop on Semantic Matchmaking and Resource Retrieval: Issues and Perspectives (SMR 2006) at the 32nd International Conference on Very Large Data Bases (VLDB)*, (2006).
6. Crockford, D. "The application/json Media Type for JavaScript Object Notation (JSON), July 2006." URL: <http://www.rfc-editor.org/rfc/rfc4627.txt>.
7. De Bruijn, J., Bussler, C., Domingue, J., Fensel, D., Hepp, M., Kifer, M., König-Ries, B. et al. "Web service modeling ontology (WSMO)." *Interface* 5 (2006): 1.
8. Ehrig, M. *Ontology alignment: bridging the semantic gap*. Springer, 2006.
9. Fernandez, A., Cong, Z. and Balta, A. "Bridging the Gap Between Service Description Models in Service Matchmaking", *Multiagent and Grid Systems*, vol. 8, no. 1, pp. 83-103. (2012)
10. Frakes, W. B. "Stemming algorithms." *Information retrieval: Data structures and algorithms* (1992): 131-160.
11. Funk, A. and Bontcheva, K. "Ontology-based categorization of web services with machine learning." In *Proceedings of the seventh international conference on Language Resources and Evaluation (LREC)*, (2010).
12. Giantsiou, L., Loutas, N., Peristeras, V. and Tarabanis K. "Semantic Service Search Engine (S3E): An Approach for Finding Services on the Web." *Visioning and Engineering the Knowledge Society. A Web Science Perspective* (2009): 316-325.
13. Hess, A. and Kushmerick, N. "Automatically attaching semantic metadata to web services." In *Proceedings of Information Integration on the Web (IIWEB) 2003* , pp. 111-116. (2003).
14. Klusch, M. and Kapahnke, P. "iSem: Approximated reasoning for adaptive hybrid selection of semantic services." *The Semantic Web: Research and Applications* (2010): 30-44.
15. Klusch, M., Benedikt F. and Sycara, K. "OWLS-MX: A hybrid Semantic Web service matchmaker for OWL-S services." *Web Semantics: Science, Services and Agents on the World Wide Web* 7, no. 2 (2009): 121-133.
16. Kopecky, J., Vitvar, T., Bournez, C. and Joel Farrell. "SawSDL: Semantic annotations for wsdl and xml schema." *Internet Computing, IEEE* 11, no. 6 (2007): 60-67.
17. Kourtesis, D., and Paraskakis, I. "Combining SAWSDL, OWL-DL and UDDI for semantically enhanced web service discovery." *The Semantic Web: Research and Applications* (2008): 614-628.

18. Lara, R., Roman, D., Polleres, A. and Fensel, D. "A conceptual comparison of WSMO and OWL-S." *Web Services* (2004): 254-269.
19. Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S. et al. "OWL-S: Semantic markup for web services." *W3C Member submission* 22 (2004).
20. McCallum, A. and Nigam, K. A comparison of event models for naive bayes text classification. In *AAAI-98 Workshop on learning for text categorization* (Vol. 752, pp. 41-48). (1998).
21. Mohanty, R., Ravi, V. and Patra, M. R. "Web-services classification using intelligent techniques." *Expert Systems with Applications* 37, no. 7 (2010): 5484-5490.
22. Paolucci, M., Kawamura, T., Payne, T. and Sycara, K. "Semantic matching of web services capabilities." *The Semantic Web—ISWC 2002* (2002): 333-347.
23. Paolucci, M., Wagner, M. and Martin, D. "Grounding OWL-S in SAWSDL." *Service-Oriented Computing-ICSOC* (2007): 416-421.
24. Pedrinaci, C., Liu, D., Maleshkova, M., Lambert, D., Kopecky, J. and Domingue, J. "iServe: a linked services publishing platform." In *Proceedings of the 1st Workshop on Ontology Repositories and Editors for the Semantic Web*. CEUR Workshop Proceedings, vol. 596. (2010).
25. Renz, I., Ficzy, A. and Hitzler, H.. "Keyword Extraction for Text Characterization." In *Proceedings of Applications of Natural Language to Data Bases*. (2003).
26. Scicluna, J., Lara, R., Polleres, A. and Lausen, H. "Formal mapping and tool to owl-s." WSMO Working Draft 17 (2004).
27. Vapnik, V., Golowich, S. E. and Smola, A. "Support vector method for function approximation, regression estimation, and signal processing". *Advances in neural information processing systems*, 281-287. (1997)
28. Willett, P. "The Porter stemming algorithm: then and now." *Program: electronic library and information systems*, 40, no. 3 (2006): 219-223.