A Directory of Heterogeneous Services[1]

Zijie Cong, Alberto Fernández, Carlos A. Soto

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

Abstract.
This paper presents a directory of heterogeneous web services, which addresses the issue of service discovery involving heterogeneous description languages such as OWL-S, SAWSDL, WSDL and plain text. Service descriptions are mapped into a unified description model, which captures various important elements in different service description approaches. Our directory then performs service registration, automatic discovery and manual browsing utilizing these unified models. A preliminary evaluation shows a satisfying result.

Keywords: service directory, service discovery, matchmaking, semantic web services, service oriented architecture.

1 Introduction

In Service-Oriented Architectures, web services can be described in various models, from highly expressive semantic web service description languages such as OWL-S and WSMO to plain text. The possibility and capability of automatic service discovery is limited by the diversity of service description models.

A directory of heterogeneous web services is presented in this paper, which addresses the issue of service discovery involving various service description models. Common approaches use the same description language for both advertisements and requests.

Services description in different description languages are mapped into a unified model, which dedicates to service matchmaking purpose, before registration. This unified model captures many important features of existing description languages, such as the semantic I/Os, category information and syntactic description. It is independent of the original service description language, thus it can be modified and expanded with minimal effort while avoiding the complication of mapping a less expressive description language, such as keywords, to a highly expressive description language with additional information requirement. A matchmaking algorithm is

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

performed over this model, thus providing heterogeneous service discovery capabilities.

The rest of the paper is organized as follows: In section 2, we describe the general structure of the directory, and the mapping from existing service description languages to a unified model. In section 3, the matchmaking process is explained in detail, and the implementation and preliminary evaluation of some components is shown in section 4. The related works and conclusion are then presented in section 5 and 6, respectively.

2 Service Directory Architecture

The architecture of our service directory is depicted in Fig. 1. 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 [16], WSMO [3]), syntactic models (WSDL [4]), hybrid (SAWSDL [6]), as well as other lighter approaches (*keyword*-, *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 [18] 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.

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 their 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.

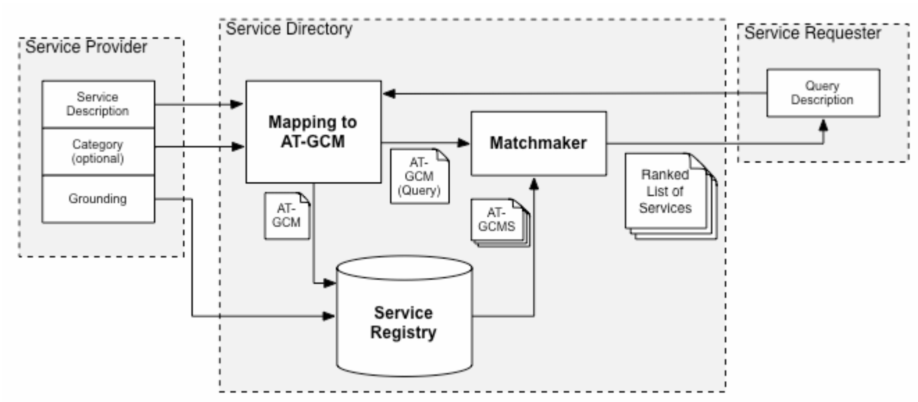

Fig. 1. Service Directory Architecture

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 A unified model for representing service descriptions

Setting out from existing conceptual comparisons between semantic web service descriptions ([11, 12, 20, 22], and considering lighter approaches too, we obtained a *General Common Model* (*AT-GCM*[2]) 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 [2]. 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 $<\mathcal{I}_{GCM}, \mathcal{O}_{GCM}, \mathcal{P}_{GCM}, \mathcal{E}_{GCM}, \mathcal{K}_{GCM}, \mathcal{C}_{GCM}, \mathcal{T}_{GCM}, \mathcal{TC}_{GCM}>$, where:

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

[2] *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 ($C_{sem} \subseteq \mathcal{N}$) (e.g. NAICS or UNSPSC).
- $\mathcal{T}_{GCM}$ is a textual description of the service.
- $\mathcal{TC}_{GCM}$ is a tag cloud. $\mathcal{TC}_{GCM} = \{<t, n>|\ t \in \{\mathrm{a}, \ldots, \mathrm{z}\}^{*}, n \in \mathbb{N}\}$.

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.

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

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.

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

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

Fig. 2 sumarises the characteristics of the *AT-GCM* that can be obtained from each original service description model.

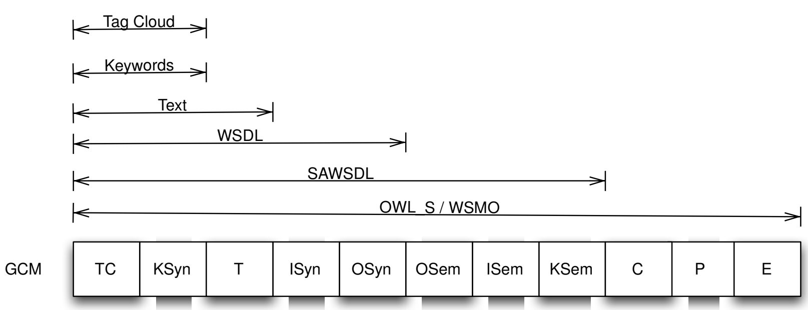

Fig. 2 AT-GCM characteristics covered by service description models

2.2 Model Expansion

Useful information about services may not always be explicitly defined by the providers in their 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 expansion of *AT-GCM* using existing elements and external resources.

A complete schema is shown in Fig. 3.

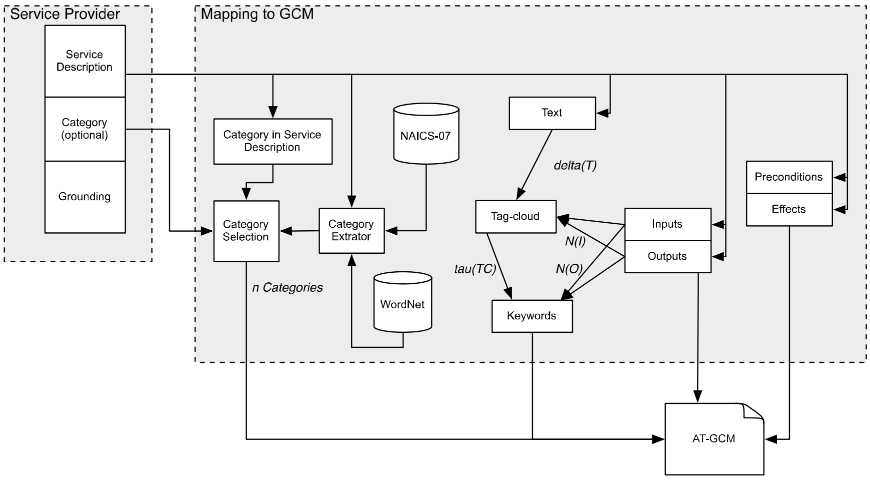

Fig. 3 Mapping to AT-GCM

2.2.1 Extracting tag-clouds and keywords from text

Although, as illustrated in Fig. 2, most service description languages include neither syntactic keywords nor tag-cloud, these two elements can be extracted from other parts of description such as text, inputs and outputs.

Function $\Delta(\mathcal{T})$ (Table 1) extracts the k most relevant keywords from $\mathcal{T}$. The relevance of each word in textual information is their TF-IDF weights [24] calculated using other textual information of services registered in our directory.

Before computing the TF-IDF weight of the word, a set of stop-words is filtered out from the text to accelerate the process. As nouns and verbs are more semantically significant than other parts of speech, words falling into the rest of lexical categories are also filtered out. This process is done using WordNet [17].

WordNet is a lexical database for English language. It 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.

We also use WordNet to lemmatization words. Comparing to other popular stemming algorithms such as Porter's [23] stemming algorithm, WordNet significantly reduces *over-stemming* errors, which could lead to false positive results.

In addition, the set of input concept names $\mathcal{N}(\mathcal{I})$ and output concept names $\mathcal{N}(\mathcal{O})$ in semantic descriptions (OWL-S, WSMO, SAWSDL) are considered for the cloud with non-character symbols removed and converted to lowercase. In 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 words in the cloud (function $\tau(TC)$, being TC a tag-cloud).

The set of input and output concept parameter types ($\mathrm{pt}(\mathcal{I})$ and $\mathrm{pt}(\mathcal{O})$) are also adopted as **semantic keywords**.

2.2.2 Category Discovery

Our directory is organized using service's category information based on the North American Industry Classification System (NAICS). 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 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 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.

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. Only elements existing in both descriptions are considered, the rest are ignored.

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 an ontology.
- The category of a service is often an element in certain classification systems, such elements are usually organized in a hierarchy, which can be considered as an ontology also.

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

Table 2 Categorizing AT-GCM components

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

Fig. 4 illustrates the complete matching schema.

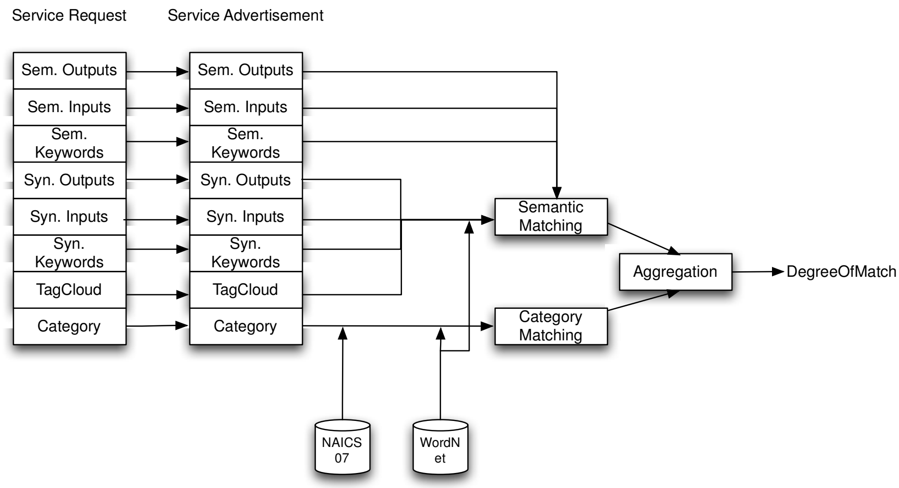

Fig. 4. Service Matchmaking based on AT-GCM

3.1 Semantic Elements Matching

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 [19]: *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, which was introduced by Y. Li et al. in [15]:

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

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.

We combine this function with the four degrees of match commented above into a unique numerical real value between 0 and 1, being *exact = 1, plug-in ∈ (0.5,1), subsumes ∈ (0,0.5)* and *fail* = 0:

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

3.1.1 Semantic Outputs/Inputs

In line with Paolucci's proposal in [19], 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^R_{sem} and O^A_{sem}, the similarity between these two outputs is calculated using function:

$$OSemMatch(O^R_{sem}, O^A_{sem}) = \begin{cases} 1, & \text{if } |O^R_{sem}| = 0 \\ \underset{o^R \in O^R_{sem}, o^A \in O^A_{sem}}{\text{Min Max}} (conceptMatch(o^R, o^A)), & \text{otherwise} \end{cases}$$

In function *OSemMatch* O^R denotes the semantic outputs from service request. Therefore, if the service request requires no outputs ($|O^R_{sem}|=0$), it returns 1, exact match, regardless of the outputs produced by service advertisement O^A_{sem}. 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^{R}_{sem} (R) and from service advertisement, K^{A}_{sem} (A) the degree of match between two sets of semantic keywords is calculated using measure proposed in [5]:

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

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

Alternative semantic similarity measures can be used, such as the measure described by Hau et al. in [7].

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*.

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:

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

where $K^{R}_{synsets}$ and $K^{A}_{synsets}$ denote WordNet synsets associated with keywords in the service request and service advertisement respectively, and δ denotes weight of a keyword, which is always 1 at current stage.

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

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

where δ_r and δ_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.

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

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:

$$CatMatch(C_1, C_2) = sim_{NAICS-07}(C1, C2)$$

3.4 Aggregation Function

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

sim_{Isyn}, sim_{Isem} $sim_{Osem,}$ $sim_{Osyn,}$ sim_{TC} , $sim_{Ksyn,}$ sim_{Ksem}, sim_{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:

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

where n_c denotes the number of elements in component C (for example, number of semantic outputs), and $\overline{N}$ 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

The directory service implementation consists of a web server to perform various operations defined in section 2 (register and search services). The server may be accessible through a web interface implemented on the same server, or through REST operations to receive and respond to customer requests.

We used SQLite[3] database to facilitate the implementation in future distributions of the service directory.

The service directory receives search requests and responds to them through JSON [9] data exchange, including a list of descriptions of the matching services and their corresponding grounding so that they can be invoked if desired.

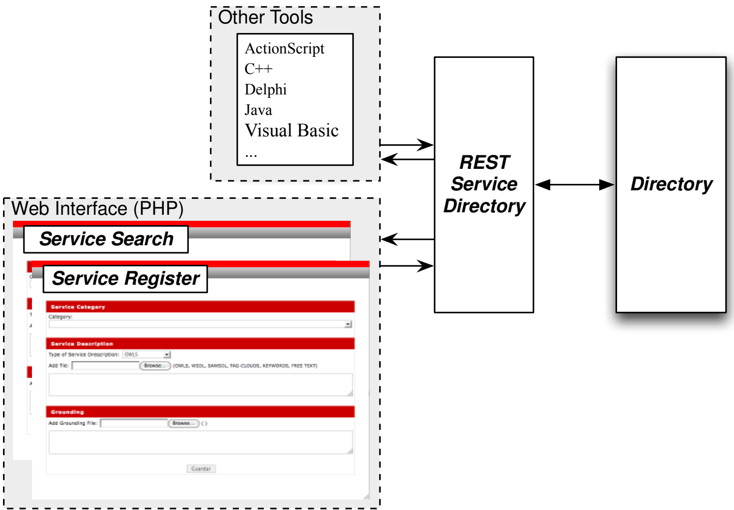

Fig. 5. Service Directory Interaction

The implemented Web Interface also uses REST to interact with the service directory. Fig. 5 shows the interaction of our proposed service directory with the Web Interface and other languages. 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.

4.1 Evaluation

Based on OWLS-TC[4] 4.0, we performed two experiments to evaluate the precision of category extraction and syntactic keywords matching.

---

[3] http://www.sqlite.org/

[4] http://www.semwebcentral.org/projects/owls-tc/

As both experiments involve syntactic matching, the relevance is relatively subjective. Therefore, the precision of the results is calculated against human judgement.

Category Extraction

We selected 78 services from the OWLS-TC, and 5 NAICS-07 categories were extracted using techniques described in section 2.2. Then we manually evaluated how many extracted categories were acceptable (agree with human judgement). The measure is essentially a *precision at 5*:

$$precision = \frac{|C_{extracted} \cap C_{human}|}{|C_{extracted}|}$$

where $|C_{extracted}| = 5$.

In comparison, we also performed an experiment in category extraction without WordNet, i.e, character-wise matching was performed over stemmed keywords from service description and category index.

The results showed an average precision of 0.698 from our approach and 0.2734 from using pure syntactic matching.

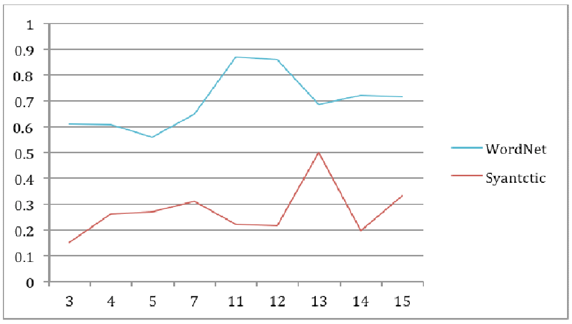

Fig. 6. Precision and number of keywords extracted

In general, the precision of extraction with WordNet is higher than pure syntactic matching. However, Fig. 6 shows that as the number of keywords increases, WordNet approach's precision decrease.

This could be due to the fact that the number of WordNet synsets associated with keywords increases rapidly hence overgeneralized the *domain* of the service An other possible cause could be that the experiment was performed with a relatively small amout of samples, thus noises are very obivious, for example, only one service has 13 keywords extracted and its value cloud be an exceptional extreme value.

Syntactic Keywords Matching

We selected 8 service requests from the OWLS-TC's *Request and Relevance Sets,* and using relevance information provided by OWLS-TC as the benchmark, our syntactic matching algorithm has an average precision of 80.5%.
Again, this results could be not reliable due to the small number of samples used. Therefore, further larger scale experiments will be one of our future works

5 Related Work

Some (not many) other efforts have been made trying to align or compare different service description approaches. As we mentioned in section 2.1, we set out from existing conceptual comparisons between popular semantic web service languages [11, 12, 20, 22] to obtain a general model description of services that facilitates their discovery.

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. [14, 19]). Klusch et. al [10] 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 description.

The directory service using a common model (AT-GCM) in the same direction as iServe [21] 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) [13] 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 [8]. 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 we have dealt with the problem of service discovery in open systems. We proposed an architecture that considers the alignment of service description models, and the transformation of them into a unified common model. We do not only consider explicit information specified in structured service descriptions, but we enrich descriptions with additional information extracted using text processing. Although we provided with an alignment mechanism for a set of service description languages, 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.

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 implementation and a preliminary evaluation showed a satisfying result regarding category and keywords extraction. Further evaluations, such as F-measure and recall of extracted categories as well as precision/recall of service are part of our future plans.

7 References

1. Ambite, J.L., Darbha, S., Goel, A., Knoblock, C.A., Lerman, K., Parundekar, R., Russ, T.: Automatically constructing semantic web services from online sources. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 17–32. Springer, Heidelberg, 2009.
2. Balta, A., and A. Fernandez. Towards Service Matchmaking of Heterogeneous Service Descriptions. Worshop on Service-Oriented Computing: Agents, Semantics, and Engineering (SOCASE@AAMAS2010), Toronto, 2010.
3. Bruijn, J., Bussler, C., Domingue, J., Fensel, D., Hepp, M., Keller, U., Kifer, M., K¨onig-Ries, B., Kopecky, J., Lara, R., Lausen, H., Oren, E., Polleres, A., Roman, D., Scicluna, J., and Stollberg, M. Web Service Modeling Ontology (WSMO). W3C Member Submission, 2005. http://www.w3.org/Submission/WSMO/.
4. Christensen, E., Curbera, F., Meredith, G. and Weerawarana, S. Web Services Description Language (WSDL) 1.1. http://www.w3.org/TR/wsdl, March 2001
5. Ehrig M. Ontology Alignment: Bridging the Semantic Gap. Springer. 2007.
6. Farrell, J. and Lausen, H. Semantic Annotations for WSDL and XML Schema (SAWSDL). W3C Recommendation 28 August 2007. http://www.w3.org/TR/sawsdl/
7. Hau, J., Lee, W., Darlington, J.: A Semantic Similarity Measure for Semantic Web Services. In: Proceedings of the Workshop Towards Dynamic Business Integration co-located with the 14th International World Wide Web Conference (WWW), 2005.
8. Heß, A., Johnston, E., Kushmerick, N.: Machine Learning for Annotating Semantic Web Services. In: Semantic Web Services: Papers from the 2004 AAAI Spring Symposium Series. AAAI Press , 2004.
9. JSON, JavaScript Object Notation. http://www.json.org/, 2011
10. Klusch, M., Fries, B., 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, Volume 7, Issue 2, April 2009, Pages 121-133.

11. Kourtesis, D., and Paraskakis, I. Combining SAWSDL, OWL-DL and UDDI for Semantically Enhanced Web Service Discovery. Springer Berlin / Heidelberg. Pages 614-628. 2008.
12. Lara, R., and Polleres, A. D4.2v0.1 Formal Mapping and Tool to OWL-S, WSMO working draft 17 december 2004. http://www.wsmo.org/2004/d4/d4.2/v0.1/
13. Levy, A.Y.: Logic-based techniques in data integration. In: Minker, J. (ed.) Logic-Based Artificial Intelligence. Kluwer Publishers, Dordrecht, 2000.
14. 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.
15. Li, Y, Bandar, Z.A. and McLean, D. An approach for measuring semantic similarity between words using multiple information sources. IEEE Trans- actions on knowledge and data engineering, pages 871–882, 2003.
16. Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDemott, D., McIlraith, D., Narayanan, D., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., and Sycara, K.. OWL-S: Semantic Markup for Web Services. W3C Member Submission, 2004. Available from http://www.w3.org/Submission/OWL-S/.
17. Miller, G.A. WordNet: a lexical database for English, Communications of the ACM, pages 39-41, 1995. Association for Computational Linguistics.
18. NAICS Association. NAICS code searching. http://www.naics.com/search.htm, 2004.
19. Paolucci, M., Kawamura, T., Payne, T., and Sycara, K. Semantic Matching of Web Service Capabilities. In ISWC, pages 333–347. Springer Verlag, 2002.
20. Paolucci, M., Wagner M., and Martin M. Grounding OWL-S in SAWSDL. Springer Berlin / Heidelberg. Pages 416-421. 2007.
21. Pedrinaci, C., Liu, D., Maleshkova, M., Lambert, D., Kopecky, J. and Domingue, J. iServe: a Linked Services Publishing Platform, Workshop: Ontology Repositories and Editors for the Semantic Web at 7th Extended Semantic Web Conference, 2010.
22. Polleres, A., and 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/.
23. Porter, M. F. An algorithm for suffix stripping, Program, vol. 14, no. 3, 130-137, 1980.
24. Salton, G. Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley, 1989.