Towards an abstract architecture for service discovery with semantic alignment

Analay Baltá, Alberto Fernández

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

Abstract. In large-scale open environments mechanisms for locating appropriate services have to deal with the additional problem of semantic mismatches among the components. Semantic alignment mechanisms need to be purposefully integrated into a service discovery framework in order to fully exploit its potential. The objective of this paper is to present an ongoing work towards the analysis and design of basic mechanisms able to locate adequate services in open heterogeneous environments. An abstract architecture that addresses the semantic mismatches at service description model level as well as domain ontology is presented. Several open issues are pointed out.

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

1 Introduction

In multiagent systems, agents communicate with the aim of achieving their objectives. Agents are autonomous entities capable of planning the tasks they have to carry out to maximize their utilities. An individual agent may require a service to be performed by another entity. In order to be able to achieve a fruitfully interaction the two agents must understand the semantic of the messages they exchange. This is typically done by sharing the same ontology, although this is not easy to achieve in open systems. Another option is to use ontology bridges, which make use of ontology alignment techniques [1, 2] to transform information from one ontology to another.

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. After an agreement has been reached the service can be called. *Agreement Technologies* [3] like semantic alignment, negotiation, argumentation, virtual organizations, decision making, learning, trust, and so on, will be used to develop such large-scale open systems.

We concentrate on the phase of provider location. Distributed service directories and efficient decentralised matching techniques with powerful description languages are essential for dynamic and scalable service discovery. Furthermore, in such environments the mechanisms for locating appropriate services have to deal with the additional problem of semantic mismatches among the components.

In this paper we present an ongoing work towards the development of a service discovery framework where semantic alignment mechanisms are purposefully integrated into.

The rest of the paper is organized as follows. In the next section we begin by describing the Service Oriented Architecture so as to establish the context in which this work is situated. Section 3 presents an abstract architecture for semantic service discovery, which pays especial attention to semantic alignments of service descriptions. In section 4 we discuss several open issues for further research. Finally, we present some conclusions and future work.

2 Service Oriented Architectures

Fig. 1 shows a general Service Oriented Architecture (SOA) [4]. It involves three or more parties: a *requester,* one or more *providers* and a *directory*, which supports services during the transaction and possibly mediates between the requester and the provider. Roughly speaking, the requester corresponds to the client, and the provider corresponds to the server in the typical client-server architecture.

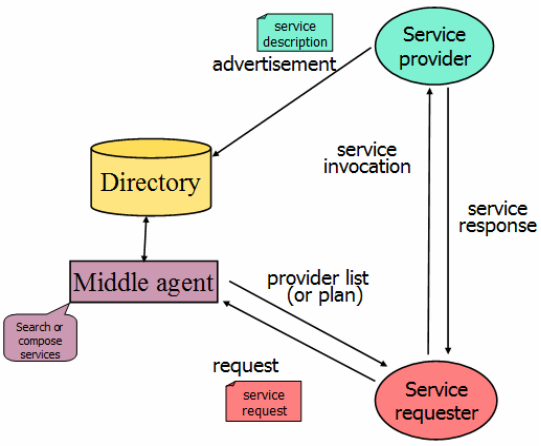

Fig. 1 General Service Oriented Architecture.

Web Services are the reference enabling technology for SOA. Web Services can be seen as a collection of technologies, protocols and standards that build programming solutions for specific application integration problems. As the number of available web services is steadily increasing, companies realize the need for automatically discovering web services and having an automated composition.

SOA combines the service discovery, selection, and engagement, thus adding a new level of functionality on top of the current Web. The addition of semantic information to describe Web Services, in order to enable the automatic location, combination and use of distributed components, is nowadays one of the most relevant research topics due to its potential to achieve dynamic, scalable and cost-effective Enterprise Application Integration and eCommerce.

The process of discovering and interacting with a Semantic Web Service includes the following phases [5]:

(i) Candidate Service Discovery is the distributed search for available services that can accomplish the client's internal goal or objective. It is a process of identifying candidate services by clients. It involves three types of stakeholders: service *providers* that publish service advertisements, service *requesters* that require a service and *matchmakers* that accept descriptions of available service from providers and match them against requirements from requesters.

(ii) Service Engagement includes the process of interpreting candidate service enactment constraints, described by each candidate service published, and then requesting or possibly negotiating with prospective services to reach an agreement. Engagement concludes with both service and client knowing and agreeing to the terms of service provision in an explicit or implicit service contract.

(ii) Service Enactment consists of alternative protocols to initiate service activity, monitor service processes, and confirm service completion. If the service terminates abnormally after a contract has been formed, there may be a final set of protocol interactions to address compensation issues.

Proper methods to enable the automatic location and selection of suitable services in order to solve a given task or user request are an essential ingredient. To this end, 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 [6], WSMO [7]) to syntactical ones (WSDL [8], keywords, tag clouds), with some approaches in between (SAWSDL [9]). Semantic service descriptions are supported on ontologies.

In this context, several description frameworks to semantically match services on the one hand and service requests on the other have been presented in the literature. 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.

3 Abstract Service Discovery Architecture

Fig. 2 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.

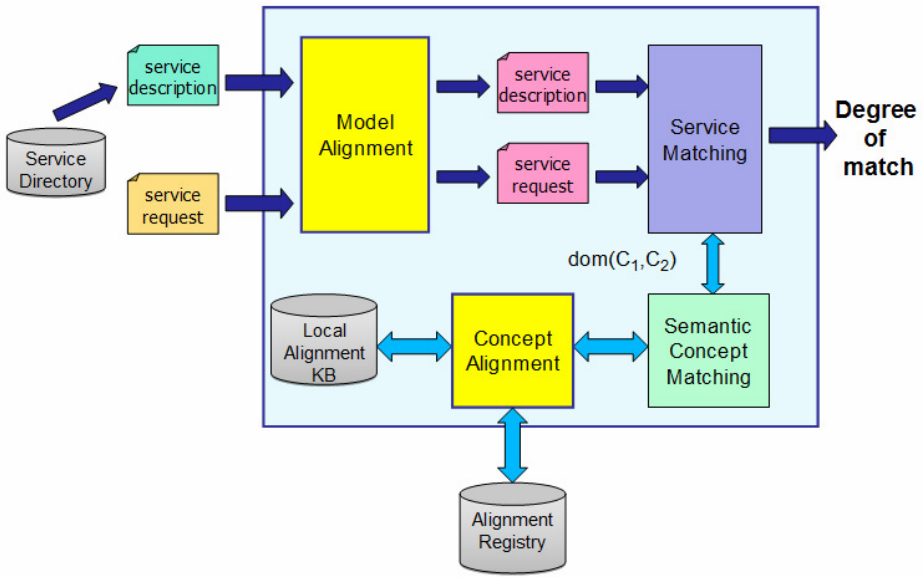

Fig. 2 Service Discovery Architecture

The first step in the matching of two services is the *alignment of the service description models*. 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. Service model alignment consists of mapping both service descriptions (request and advertisement) into a common service model.

Once the adequate model alignment has been applied, the unified service descriptions are prepared to be matched. The *Service Matching* module takes those two descriptions and returns the degree of match between them. We envision here the existence of different matchmakers to deal with different common models. 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 *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 a 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 alignment if there is an alignment published by third parties.

In the next sections we go into details of the building blocks of the proposed architecture.

3.1 Service Model Alignment

As commented above, service model alignment consists of defining a common model for two different ones, and mapping descriptions described in those source models into the common model. This transformation may produce a loss of expressiveness in at least one of the original descriptions, especially if they use models with different expressiveness power. Here, we do not aim at the design of a unified model for any description, which would probably lead us to the definition of a very simple (lightweight) model to account for all the different source models. However, we envision the definition of mappings between pairs of models, thus keeping that particular common model as close to the original ones as possible. Besides, this approach is more modular and flexible to consider new models. Model-to-Model alignments consist of three steps:

1. *Conceptual analysis* of characteristics finding mappings between models. Note that this task can be done focusing on service matchmaking, i.e. only the aspects considered by the matching techniques have to be mapped.
2. Definition of a *common model language* (CML), which might be one of the originals. The use of standard languages is encouraged here.
3. Implementation of a tool for *automatic transformation* of service descriptions from the original models to the CML.

In the next subsection we show an example of the alignment between the two most important Semantic Web Service description languages, OWL-S and WSMO.

3.1.1 WSMO/OWL-S alignment

In order to establish the relationships between the terminologies used in each ontology and propose a mapping, we set out from existing conceptual comparisons between WSMO and OWL-S [10, 111]. As shown in Table 1, Web Service descriptions in WSMO are defined by their *preconditions, posconditions, assumptions* and *effects*, and their equivalent in OWL-S are *inputs, outputs, preconditions* and *results*, respectively.

Table 1. Mapping between OWL-S and WSMO.

Element	OWL-S	WSMO
input	hasInput	precondition
output	hasOutput	postcondition
precondition	hasPrecondition	assumption
effect	hasResult	effect
others		

After the conceptual mapping between models, the next step is the definition of a common description language. We opt for using RDF as the common model language. RDF is a W3C recommendation language to represent resources on the Web. There are a lot of RDF contents and tools to process them. Although RDF is less expressive than OWL-S and WSMO, it is enough (in practice) for representing the information needed for service search. It also allows the use of SPARQL [12] to query the descriptions. The resulting RDF Schema graph describing semantic Web services for this mapping is shown in Fig. 3 and the RDF code is shown in **Fig. 4**.

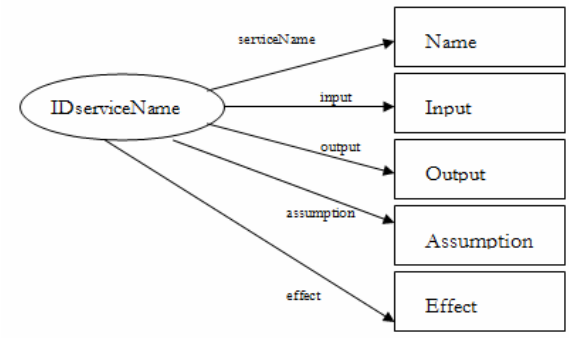

Fig. 3. RDF graph for service discovery

```
<?xml version="1.0"?>
<rdf:RDF>
  < rdf: Description rdf:ID="IDserviceName">
  </rdf:Description>
  <rdf:Description rdf:ID="serviceName">
  <rdfs:domain rdf:resource="#IDserviceName"/>
  <rdfs:range rdf:resource="#Name"/>
  </rdf:Description>
  <rdf:Description rdf:ID="input">
  <rdfs:domain rdf:resource="#IDserviceName"/>
  <rdfs:range rdf:resource="#Input"/>
  </rdf:Description>
  <rdf:Description rdf:ID="output">
  <rdfs:domain rdf:resource="#IDserviceName"/>
  <rdfs:range rdf:resource="#Output"/>
  </rdf:Description>
  <rdf:Description rdf:ID="assumptions">
  <rdfs:domain rdf:resource="#IDserviceName"/>
  <rdfs:range rdf:resource="#Assumptions"/>
  </rdf:Description>
  <rdf:Description rdf:ID="effect">
  <rdfs:domain rdf:resource="#IDserviceName"/>
  <rdfs:range rdf:resource="#Effect"/>
  </rdf:Description>
</rdf:RDF>
```

Fig. 4. RDF/XML representation of the CML for the OWL-S/WSMO alignment

3.2 Service Matching

Many of the current approaches to Semantic Web Services matching, particularly those based on OWL-S, started from the work of Paolucci et al. [13]. This approach proposes a matching algorithm that takes into account inputs and outputs of advertised (A) and requested (R) services. A match between two output concepts (O_A, O_R) is the (subsumption) relation between the two concepts in the ontology. They differentiate among four degrees of match: *exact* ($O_A = O_R$), *plug-in* (O_R subsumes O_A), *subsumes* (O_A subsumes O_R) and *fail* (otherwise). An output matches if and only if for each output of the request there is a matching output in the service description. If there are several outputs with different degree of match, the minimum degree is used. The same algorithm is used to compute the matching between inputs, but with the order of request and advertisement reversed. Finally, the set of service advertisements is sorted by comparing output matches first, if equally scored, considering the input matches. Several authors extend or propose variations to that proposal (e.g.[14,15]).

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 ([17, 188]). Other authors do not base concept similarity on the distance between the concepts ([19, 20, 21]).

We propose a combination of service matching and concept similarity. In [22] 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 (e.g. exact, plugin, ...), and then the level is refined with the (numerical) similarity value. 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.

3.2.1 SPARQL as service query language

If we envision the representation of services in RDF (like the proposed common model for OWL-S and WSMO), then SPARQL [12] might be used as a query language for services. Note that the SPARQL query might be created either from the scratch or by transforming a service request. In this section we analyse how those requests look like, and the potential and limitations of SPARQL as service request language. We use the RDF service descriptions proposed in section 3.1.1.

This is an example of a first attempt of defining a flight service request in SPARQL:

```
SELECT ?Name
WHERE {?x serviceName ?Name .
       ?x input #DepartureAirport .
       ?x input #ArrivalAirport .
       ?x input #OutboundDate .
       ?x input #InboundDate .
       ?x output #FlightsFound .
       ?x output #PreferredFlightItinerary .
       ?x effect #HaveSeatResult}
```

This query will return all service names that have at least the four inputs, the two outputs and the effect specified in the query (conditions in the *WHERE* section are interpreted as conjunctions).

Remember that a service advertisement (*A*) match a request (*R*) if and only if all the inputs of *A* are provided by *R* and all the outputs of *R* are provided by *A*.

The first problem we identify is that services that only need a subset of the specified inputs would not be returned. However, services providing more outputs than needed by the requester cause no problem. This problem can be solved by decomposing the query in two: (i) querying the advertisements using the outputs and (ii) using the results to query the original request about the inputs.

The next problem identified is that querying RDF graphs only returns *exact* matches. The use of an inference engine to classify the ontology (compute *subclass* relations) or to interpret the query would provide subsumption reasoning, thus supporting the matchmaking using levels of match (*exact, plugin, subsumes, ...*).

However, if we need more refined complex matching functions, e.g. taking into account the distance between concepts in a taxonomy, there is no straight way to do it. Instead, new SPARQL inference engines adapted for service matchmaking should be developed.

Finally, we have to devise how to include semantic alignments into this approach. A solution could go in the line of query transformation by including fragments to consulting RDF alignment specifications (see section 3.3).

3.3 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 [23]:

$$<e, e', n, R>$$

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 alignment 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 Innitiative[1].

4 Open Issues

In addition to the aforementioned aspects, we point out here some additional open issues that will need to be dealt with.

The abstract architecture proposed in this paper describes the process of obtaining the degree of match between a service advertisement and a service request. Typically, this process is carried out by a middle agent or matchmaker (either separated or integrated in the directory of services). Roughly, that process takes as inputs two service descriptions and returns a matching degree (a level of match or a numerical value).

However, there might be situations in which this task should be done in a different way. Firstly, it is arguable that the matching process can be effectively done in a one step process. As Lara point out [24], the semantic description capability of services and of customer goals can be exploited by the service discovery based on a two-phased service discovery model. The first phase identifies services that can provide results required by the customer or specifies semantic matchmaking on the goal template level. In the second phase, the input values required by relevant services are considered and only services for which the customer can provide appropriate input, and for which this input can lead to the expected results are selected. The input values will partially determine the results of the service. Then there is a combination of query of selected service and answer of customer. In the best case the customer can

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

only guess, when defining his goal, what input values will be required by services satisfying such goal. In addition, the customer might have a considerable volume of information from which input values to services can be obtained.

Another usual convention is that service matchmaking is completely carried out in the middle agent (or directory) side. This is possible under the assumption that the middle agent has all the necessary information, i.e. it has complete service descriptions. However, under some circumstances, the provider or the requester might not be interested in revealing some private data, i.e. a credit card number. In those contexts, some mechanisms are needed to deal with such sensitive information. One possible option is to carry out part of the matching on the requester and/or the provider side. In these cases, efficiency issues have to be considered.

In the development of methods, techniques and tools for open large scale environments, scalability issues are fundamental. In the context of service discovery and the architecture presented in Fig. 2, there are two main points that must be considered. They are the decentralised service directories and the discovery of ontology alignments. In both cases, the recent trends in querying distributed data on the semantic web by means of SPARQL end points will be considered in future research.

As described previously, services can be described in different languages at several levels of expressiveness. Depending on the context in which the searched service is going to be used, the set of candidate services can be larger or smaller. For instance, if the service is expected to be used for automatic invocation or for composition, only semantic service descriptions (OWL-S, WSMO, SAWSDL, ...) would be considered. However, if the service is expected to be used by a human user then also more lightweight descriptions (e.g. keyword-based) would be considered.

5 Conclusion

In this paper we have dealt with the problem of service discovery in open systems. We proposed an abstract architecture that has semantic alignment as a first citizen component. We provided preliminary ideas and developments towards the construction of a service discovery framework in which semantic alignment mechanisms are purposefully integrated into.

In particular, we discussed in detail the alignment of OWL-S and WSMO, their differences and the transformation of both into a RDF common model. We also proposed the combination of service matching and concept similarity into an integrated service matching framework. We analysed the use of SPARQL as a service query language and identified the pros and cons, and RDF as ontology alignment representation format.

The definition of other service description model alignments as well as the implementation of the proposed framework are part of our ongoing work. In the future we also plan to investigate the issues pointed out in section 4 (two-phase service discovery, distributed service directories and context).

Acknowledgment: This work has been partially supported by the Spanish Ministry of Science and Innovation through grants TIN2006-14630-C03-02, and CSD2007-0022 (CONSOLIDER, INGENIO 2010)

6 References

1. Jérôme Euzenat and Pavel Shvaiko. Ontology matching. Springer, Heidelberg (DE), 2007
2. Marc Ehrig: Ontology Alignment: Bridging the Semantic Gap. Springer. 2007.
3. Agreement Technologies. www.agreement-technologies.org
4. Papazoglou, M., Van den Heuvel, W. Service oriented architectures: approaches, technologies and research issues. Springer-Verlag 2007.
5. 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.
6. 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/.
7. 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/.
8. E. Christensen, F. Curbera, G. Meredith, and S.Weerawarana. Web Services Description Language (WSDL) 1.1. http://www.w3.org/TR/wsdl, March 2001
9. Joel Farrell and Holger Lausen. Semantic Annotations for WSDL and XML Schema (SAWSDL). W3C Recommendation 28 August 2007. http://www.w3.org/TR/sawsdl/
10. 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/.
11. Lara, R.., 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/
12. W3C World Wide Web Consortium. SPARQL Query Language for RDF. W3C Recommendation 15 January 2008. http://www.w3.org/TR/rdf-sparql-query/.
13. Paolucci, M., Kawamura, T., Payne, T., and Sycara, K. Semantic Matching of Web Service Capabilities. In ISWC, pages 333–347. Springer Verlag, 2002.
14. Klusch, M., Fries, B., and Sycara, K. Automated semantic web service discovery with owls-mx. In AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, pages 915–922, New York, NY, USA, 2006. ACM Press.
15. 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.
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, 19(1):17–30, 1989.
17. 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.
18. 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.

19. Resnik, P. Using information content to evaluate semantic similarity in a taxonomy. In IJCAI, pages 448–453, 1995.
20. Borgida, A., Walsh, T., and Hirsh, H. Towards measuring similarity in description logics. In Description Logics, 2005.
21. Noia, T., Di Sciascio, E., Donini, F., and Mongiello, M. Semantic matchmaking in a p-2-p electronic marketplace. In SAC, pages 582–586. ACM, 2003.
22. 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
23. 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.
24. Lara, R. Two-phased web service discovery. In AI-Driven Technologies for Services-Oriented Computing workshop at the Twenty First National Conference on Artificial Intelligence (AAAI-06), Boston, USA, 07/2006 2006.