

# Lightweight Ontology-based Service Discovery in Mobile Environments\*

Devis Bianchini, Valeria De Antonellis, Michele Melchiori and Denise Salvi  
Università di Brescia  
Dip. di Elettronica per l'Automazione  
Via Branze, 38  
25123 Brescia - Italy  
{bianchin|deantone|melchior|salvi}@ing.unibs.it

## Abstract

*The current ever-growing evolution of mobile technologies suggests the possibility of accessing services, in an itinerant and ubiquitous way, through many kinds of mobile devices (laptops, palmtops, cellular phones and so on). On the other hand, the highly dynamic and context-dependent requirements of services in distributed environments motivate and recommend the use of ontology-based techniques and tools to automatically locate services that fulfill a given user request.*

*Our aim in this work is to propose a lightweight ontology-based approach to allow the service discovery on mobile terminals, taking into account limited user interactions supported by this kind of devices. In the proposed approach, the user can specify the requested service in terms of expected capabilities. A semantic-enriched framework to describe services and an ontology-based discovery approach where such framework is exploited are applied to find services fulfilling the user needs, combining together different kinds of comparison strategies that provide a flexible and efficient matchmaking between service descriptions.*

## 1 Introduction

The current ever-growing evolution of mobile technologies allows users to access services using many kinds of mobile devices (laptops, palmtops, cellular phones and so on). On the other hand, Semantic Web technologies, in particular ontology-based techniques and tools, are exploited to add semantics to service descriptions and to automatically locate services that fulfill a given user request in highly dynamic environments.

Current approaches to automatic service discovery aim at

---

\*This work has been partially supported by the ESTEEM PRIN Project funded by the Italian Ministry of Education, University and Research, and by NoE INTEROP IST Project n. 508011 - 6th EU Framework Program.

defining what kind of service description is best suitable to express both static and dynamic aspects and perform matchmaking on these descriptions by applying different techniques ranging from logic-based to similarity-based strategies [1, 6, 7, 8, 9]. These techniques exploit some semantic description frameworks for services, among which OWL-S [5] and WSMO [10] are the main ones. However, the proposed approaches request that rather complex semantic descriptions are manually added to the service; in our approach we rely on WSDL service descriptions since WSDL has been accepted as industry standard for web service and most of available web services support WSDL standards. In [12] a simple service discovery approach based on user's preferences is proposed for tiny mobile devices. They consider a simple service ontology, where available services are classified by associating them to the concepts of a domain ontology that can be browsed on mobile devices. Moreover, service discovery on mobile devices has been addressed also in industry [11].

Our aim is to propose a lightweight semantic-enriched framework to describe services and an ontology-based discovery approach where such framework is exploited and to adapt it to the service discovery on mobile terminals, taking into account limited user interactions supported by this kind of devices. In the proposed approach, the user can easily specify the requested service in terms of expected capabilities and requested services fulfilling the user needs are found, combining together different kinds of comparison strategies that provide a flexible and efficient matchmaking between service descriptions.

The paper is organized as follows: Section 2 describes the lightweight ontological framework exploited for service discovery, as explained in Section 3; Section 4 concludes the paper and underlines possible future work.

## 2 Lightweight semantic service description

A comprehensive ontological framework to support an efficient and effective service discovery has already been

presented in [2]. In this work we describe how we adapted it to take into account mobile environments requirements by developing a so called *lightweight* ontological framework, to indicate that the adaptation involves a simplified approach to service discovery to cope with the limited capabilities of mobile devices. In fact, the presented approach focuses on specifying few elements (the requested service operations and outputs) in the request formulation. This is the basis to define simplified user interfaces and interactions that are suitable to the limited interface capabilities of mobile devices.

According to the WSMO approach [10] a service is the provision of value for the requester in some domain. The value provisioning in some given domain has priority on how the requester and the provider interact. A requester is therefore interested *in primis* in what a service is able to provide him, then how the service goal is achieved or what is needed to enable service execution. As a consequence, a service request should mainly express what are the expected capabilities of the service.

In the WSDL document, that describes the interface of a service, the expected capabilities of a service can be described by the operations and the output parameters. Representing these features with a Description Logic expression, a service can be described as a conjunction of:

- a concept in the form  $\exists\text{hasOperation}.OP$ , where  $OP$  is an atomic concept representing an operation performed by the service;
- one or more concepts in the form  $\exists\text{hasOutput}.OUT$ , where  $OUT$  is a concept representing an output of the service;  $OUT$  can be defined as an atomic concept, an enumeration  $\{o_1, o_2, \dots, o_n\}$  of nominals or a complex concept obtained by applying the *intersection* operator ( $\sqcap$ ), the *union* operator ( $\sqcup$ ) and the *negation* operator ( $\neg$ ).

**Example 2.1** *We explain our approach by means of a running example throughout the paper. We consider the problem of fireman squads that want to optimize their work in emergency situations. In particular, the focus is on the following scenario: fireman squads reacting to a toxic cloud spreading from a fire in a chemical industry. The components of squads are equipped with mobile devices (PDAs or smart phones). Firemen should be able to locate emergency area and features (hospitals, rescue areas or other elements of interest) on mobile devices. Each mobile device communicates with a emergency service centre, requesting services that provide useful information to manage the emergency. In particular, are typically requesting maps with aerial photos, streets, gas pipes, water pipes and so on. Emergency squads are divided in two teams: the first team is in charge of putting out the fire (and needs information about gas pipes, water resources and streets), while the second one in evacuating people (requesting information about transport*

*facilities and points of interest such as hospitals or rescue areas). The emergency service centre looks for requested services and it is also able to locate (through the GPS system) and identify the kind of device from which the request is sent.*

*To perform the evacuation task it is important to know what are the transport facilities and the rescue areas where people could be transferred. Through a mobile device with a suitable service discovery interface, the following request could be sent to the emergency service center:*

```
Request1 ≡ ∃hasOperation.displayMap ⊓
           ∃hasOutput.streets ⊓
           ∃hasOutput.rescueArea
```

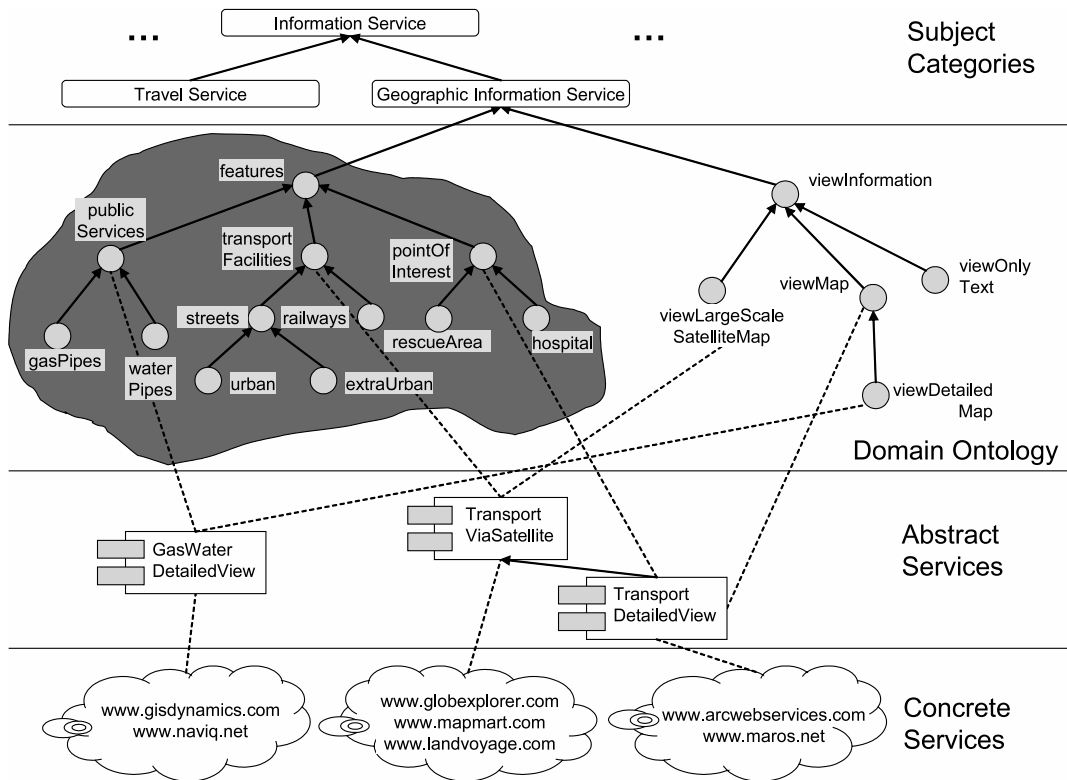
*On the other hand, firemen involved in putting out the fire should have information about gas pipes, water resources and streets, sending a request like the following:*

```
Request2 ≡ ∃hasOperation.displayMap ⊓
           ∃hasOutput.streets ⊓
           ∃hasOutput.gasPipes ⊓
           ∃hasOutput.waterPipes
```

□

To improve the efficacy and efficiency of service retrieval, considering in particular dynamic mobile environments, however, additional semantics should be associated to service description. To do this, service descriptions are embedded in a domain-dependent ontological framework composed of a Domain Ontology, a Thesaurus and a Service Ontology. In this section, we discuss how these three components are related and in Section 3 we will show how they are exploited for service discovery and selection. Figure 1 shows a portion of the ontological framework used for the domain of our reference scenario.

The upper layer of the framework presents a taxonomy of Subject categories, giving a topic-driven access to the underlying layers. Selecting a Subject category it is possible to restrict the search of desired services to a particular domain, referring to a particular Domain Ontology that is related to the chosen category. Subject categories are obtained from standard available classification just used in traditional UDDI Registries. The Domain Ontology is used to represent the domain knowledge through the conceptualizations of the elements used in service descriptions (operation names and output parameters), organized into generalization hierarchies. The Thesaurus (automatically derived from available lexical systems such as WordNet) is used to relate names of concepts of the Domain Ontology to other terms by means of terminological relationships (e.g., *synonymy*, *hypernymy*, etc.) in order to extend matching possibilities between the concept names used in the service request and the names used in the descriptions of provided services. The Service Ontology is used to organize services on different levels of abstraction, distinguishing between *Abstract* and *Concrete* services. Abstract services are connected to the outputs they provide and to the capabilities they perform by means of links to the concepts of



**Figure 1. A portion of the ontological framework exploited in the domain of geographic information services.**

the Domain Ontology. They have been introduced to represent sets of similar Concrete services, differentiated by quality parameters such as the kind of device or the location for which they have been implemented. Similarity among Concrete services is established by applying the same coefficients used for similarity-based matchmaking strategy introduced in Section 3.1. Concrete services, positioned in the lower level of the Service Ontology, are the directly invocable services implementing the functionalities represented by Abstract ones. Abstract services can also be organized into generalization hierarchies: we say that an Abstract service is a *specialization* of another one if it provides at least the same outputs and performs the same capabilities or provides/performs more specific outputs/capabilities.

**Example 2.2** Figure 1 shows a portion of the ontological framework used in the domain of geographic information services that can be used by the emergency service centre. Abstract services are related to ontological concepts representing their capabilities and information they provide. For example, the *TransportViaSatellite* Abstract service provides satellite maps overlapped with the network of transport facilities (streets and railways). Abstract services can be related by means of generalization relation-

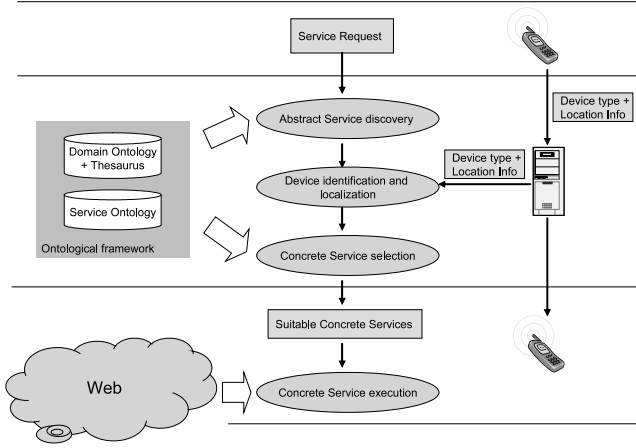
ships. The *TransportDetailedView* Abstract service provides all the features of the first one, but additionally it can render more detailed vectorial maps where urban streets are visualized and points of interest such as hospitals or rescue centers can be localized.

□

### 3 Service discovery and selection

After identifying the domain of interest by means of the taxonomy of Subject categories, the discovery and selection of services that are suitable for user preferences is obtained in two phases: (i) a hybrid matchmaking approach based on the ontological framework is firstly applied on the set of Abstract services to find services that match the user needs from the functional viewpoint; (ii) once the suitable Abstract service has been identified, corresponding Concrete services are ranked and further refined taking into account the user position and the kind of device. Figure 2 shows the overall procedure. The proposed ontological framework is built upon traditional UDDI Registries: the Service and Domain Ontologies are expressed using the OWL-DL language and contain references to the descriptions of Con-

crete services in the underlying UDDI Registry, where their WSDL and URLs to their implementation on the net are stored. The results of the overall discovery procedure is a list of links to these implementations on the Web, where the Concrete service execution takes place.



**Figure 2. Ontology-based service discovery and selection in mobile environments.**

The hybrid approach combines two kinds of strategies: a *deductive strategy*, based on Description Logics, to assess the type of match among services, and a *similarity-based strategy*, exploiting properly defined metrics [4] to measure the degree of match among services and to rank selected services. The hybrid approach has been described in more details in [2] and in this work will be adapted to the mobile context.

### 3.1 Similarity-based strategy

The Thesaurus is exploited to compute the *Affinity* coefficient between names of output parameters and operations by measuring their terminological relationships [4].

The similarity between a service request  $\mathcal{R}$  and an offer  $\mathcal{S}$  is computed through two properly defined coefficients: an *entity-based similarity* coefficient ( $ESim$ ), that measures how much the compared service descriptions provide the same outputs, and a *functionality-based similarity* coefficient ( $FSim$ ), that aims at measuring how much the two services provides the same capabilities. These coefficients are based on the Dice's formula and are tailored to compare service representations presented in Section 2. They are normalized into the range  $[0,1]$  and combined together to give an overall evaluation of service similarity:

$$GSim(\mathcal{R}, \mathcal{S}) = w_1 \cdot NormESim(\mathcal{R}, \mathcal{S}) + w_2 \cdot NormFSim(\mathcal{R}, \mathcal{S}) \in [0, 1] \quad (1)$$

where weights  $w_1$  and  $w_2$ , with  $w_1, w_2 \in [0, 1]$  and  $w_1 + w_2 = 1$ , are introduced to assess the relevance of each kind of similarity in computing the global similarity coefficient.

### 3.2 Deductive strategy

The deductive strategy relies on a non standard test (named  $C \sqsubseteq_{\mathcal{TH}} D$ ) that we proposed in [3] and that is based on the semantic relationships in the Domain Ontology and the affinity according to the Thesaurus.

#### Definition 3.1 (Affinity-based subsumption checking)

Given the Domain Ontology  $DomONT$ , the thesaurus  $\mathcal{TH}$  and a pair of concepts  $C$  and  $D$  with names  $c_n$  and  $d_n$ , respectively,  $C$  is subsumed by  $D$  with respect to  $\mathcal{TH}$ , denoted by  $C \sqsubseteq_{\mathcal{TH}} D$ , if and only if one of the following conditions holds:

- $C, D \in DomONT$  and  $(C \sqsubseteq D)$  is satisfied in  $DomONT$ ;
- only  $C \in DomONT$  and  $GC(C) \cap D_{\mathcal{TH}} \neq \emptyset$ , where  $GC(C) = \{X \in DomONT \mid C \sqsubseteq X\}$  and  $D_{\mathcal{TH}}$  is the set of terms that have name affinity with  $d_n$ , that is,  $D_{\mathcal{TH}} = \{y \in \mathcal{TH} \mid (d_n \sim y)\}$ ;
- only  $D \in DomONT$  and  $SC(D) \cap C_{\mathcal{TH}} \neq \emptyset$ , where  $SC(D) = \{X \in DomONT \mid X \sqsubseteq D\}$  and  $C_{\mathcal{TH}}$  is the set of terms that have name affinity with  $c_n$ , that is,  $C_{\mathcal{TH}} = \{y \in \mathcal{TH} \mid (c_n \sim y)\}$ .

Note that we pose  $C \equiv_{\mathcal{TH}} D$  if both  $C \sqsubseteq_{\mathcal{TH}} D$  and  $D \sqsubseteq_{\mathcal{TH}} C$  hold.  $\square$

This kind of subsumption checking between concepts is exploited to verify the kind of match between the Description Logic expressions that describe an advertisement  $\mathcal{S}$  and a request  $\mathcal{R}$ . Following general guidelines in the current literature, we define five kinds of match, that can be intuitively described as follows:

- *exact match*, when the request and the offer present the same functionalities (this is a strong condition);
- *plug-in match*, when the offer provides at least the requested functionalities and possibly adds new ones;
- *subsume match*, when the functionalities provided by the offer are less than the requested ones (it is like *plug-in match*, but with the roles of  $\mathcal{R}$  and  $\mathcal{S}$  exchanged);
- *intersection match*, when the request and the offer present some common functionalities;
- *mismatch*, when no common functionalities exist between the request and the offer.

These five kinds of match are formally defined in Table 3.2.

Match	Formal definition
Exact	$\forall op_{\mathcal{R}}^i \exists op_{\mathcal{S}}^j$ such that $(Name\_op_{\mathcal{R}}^i \equiv_{\mathcal{TH}} Name\_op_{\mathcal{S}}^j)$ and $\forall out_{\mathcal{R}}^h \exists out_{\mathcal{S}}^k$ such that $(out_{\mathcal{R}}^h \equiv_{\mathcal{TH}} out_{\mathcal{S}}^k)$
Plug-in	$\forall op_{\mathcal{R}}^i \exists op_{\mathcal{S}}^j$ such that $(Name\_op_{\mathcal{R}}^i \sqsubseteq_{\mathcal{TH}} Name\_op_{\mathcal{S}}^j)$ and $\forall out_{\mathcal{R}}^h \exists out_{\mathcal{S}}^k$ such that $(out_{\mathcal{R}}^h \sqsubseteq_{\mathcal{TH}} out_{\mathcal{S}}^k)$
Subsume	$\forall op_{\mathcal{S}}^j \exists op_{\mathcal{R}}^i$ such that $(Name\_op_{\mathcal{S}}^j \sqsubseteq_{\mathcal{TH}} Name\_op_{\mathcal{R}}^i)$ and $\forall out_{\mathcal{S}}^k \exists out_{\mathcal{R}}^h$ such that $(out_{\mathcal{S}}^k \sqsubseteq_{\mathcal{TH}} out_{\mathcal{R}}^h)$
Intersection	$\exists op_{\mathcal{R}}^i, op_{\mathcal{S}}^j$ such that $(Name\_op_{\mathcal{R}}^i \sqsubseteq_{\mathcal{TH}} Name\_op_{\mathcal{S}}^j$ or $Name\_op_{\mathcal{S}}^j \sqsubseteq_{\mathcal{TH}} Name\_op_{\mathcal{R}}^i)$ or $\exists out_{\mathcal{R}}^{ih}, out_{\mathcal{S}}^{jk}$ such that $(out_{\mathcal{R}}^{ih} \sqsubseteq_{\mathcal{TH}} out_{\mathcal{S}}^{jk}$ or $out_{\mathcal{S}}^{jk} \sqsubseteq_{\mathcal{TH}} out_{\mathcal{R}}^{ih})$
Mismatch	Otherwise

**Table 1. Formal definitions of the five kind of match between an advertisement  $\mathcal{S}$  and a request  $\mathcal{R}$ .**

### 3.3 Hybrid matchmaker

The deductive and similarity-based approaches are combined to enhance precision and flexibility of the matching process [2]. Firstly, the deductive strategy is applied to classify the kind of match between the request  $\mathcal{R}$  and each available service  $\mathcal{S}$ , then similarity evaluation is performed according to the following rules:

- if *exact* or *plug-in match* occurs, from the request viewpoint the offer provides completely the requested functionalities, so  $GSim(\mathcal{R}, \mathcal{S})$  is directly set to 1 (full similarity) without computing the similarity coefficients;
- if *mismatch* occurs,  $GSim(\mathcal{R}, \mathcal{S})$  is directly set to zero;
- if *subsume* or *intersection match* occurs, the offer fulfils the request only partially and similarity coefficients are computed to quantify how much the offer satisfies the request; in this case,  $GSim(\mathcal{R}, \mathcal{S}) \in (0, 1)$ .

Only available services for which the  $GSim(\mathcal{R}, \mathcal{S})$  value is equal or greater than a given threshold  $\gamma$  are proposed among the searching results. The application of this hybrid approach ensures flexibility and decreases false negatives, since not only *exact* or *plug-in match* are considered, but also partial matches are taken into consideration by evaluating the similarity degree. Moreover, generalization hierarchy of Abstract services is exploited to make more efficient the service discovery procedure, according to the following

intuition: if an Abstract service  $\mathcal{S}_a$  matches with a given service request  $\mathcal{R}$ , then also Abstract services that provide at least the same capabilities of  $\mathcal{S}_a$  (that is, are specializations of  $\mathcal{S}_a$ ) match with  $\mathcal{R}$ . Finally, note that the discovery process is applied at the level of Abstract services. Once the desired Abstract service is found, its corresponding Concrete services are proposed among the searching results without any further application of the discovery algorithm.

**Example 3.1** *The Request1 presents a plug-in match with the TransportDetailedView Abstract service, since*

$Name\_op_{\mathcal{R}}$	$\equiv_{\mathcal{TH}}$	$Name\_op_{\mathcal{S}}$
displayMap		viewMap
$out_{\mathcal{R}}^h$		$out_{\mathcal{S}}^k$
streets	$\sqsubseteq_{\mathcal{TH}}$	transportFacilities
rescueArea	$\sqsubseteq_{\mathcal{TH}}$	pointOfInterest

*In this case, only the deductive strategy is applied and the similarity between these two services is directly set to 1. The Concrete services {www.arcwebservices.com, www.maros.com} are proposed to the selection phase with a degree of similarity equal to 1.*

*On the other hand, Request2 and the GasWaterDetailedView Abstract service present an intersection match, since*

$Name\_op_{\mathcal{R}}$	$\sqsupseteq_{\mathcal{TH}}$	$Name\_op_{\mathcal{S}}$
displayMap		viewDetailedMap
$out_{\mathcal{R}}^h$		$out_{\mathcal{S}}^k$
gasPipes	$\sqsubseteq_{\mathcal{TH}}$	publicServices
waterPipes	$\sqsubseteq_{\mathcal{TH}}$	publicServices

*In this case the similarity strategy must be applied to quantify how much the advertisement is suitable for the request.*

□

### 3.4 Context-aware service selection

Once the candidate Concrete services have been obtained from the discovery phase, they are filtered with respect to the type and the position of the requesting device. We assume that each device is equipped with a GPS system or another positioning system and that the central unit is able to locate it with a satisfying precision. We model the location as shown in [4], where it can be identified by zero or more *Geo Positions*, i.e., latitudes and longitudes, *Districts*, e.g., special-interest areas, *Towns* and *Countries*. Denoting with  $Location(\mathcal{R})$  and  $Location(\mathcal{S})$  the information about the position of the requesting device and the information about the location for which the service  $\mathcal{S}$  is provided, respectively, we filter out the Concrete service  $\mathcal{S}_i$  from the results if the condition  $Location(\mathcal{R}) \subseteq Location(\mathcal{S}_i)$  is not satisfied (see [4]). We assume that such a comparison is performed in the basis of information stored in the central unit, together with a table stating the maximum and minimum definition admitted for each kind of device: this information is exploited to filter out services whose outputs cannot be provided on the requesting device.

## 4 Summary and future work

In this paper we have lightweighted the ontological framework introduced in [2] to cope with dynamic mobile environments. In this version of the framework we considered only generalization/specialization hierarchies between concepts and services (that, is, the most intuitive kind of semantic relationship for the user) and we have modeled in a more compact way the Domain and the Service ontologies.

The proposed lightweight framework is suitable to work within a mobile environment not only because of the simplified way to express the service request, but also because it adapts itself to the dynamicity of the mobile environments. We distinguish among two kinds of dynamicity: the functional one and the location of user. In the first case, the user changes his functional needs, requesting, for example, additional or different outputs or capabilities. In this case, starting from the current Abstract service, semantic relationships can be followed to visit first the Abstract services that are semantically related to the current one. On the other hand, the user can change his position on the territory and the system keeps into account this aspect in proposing the services. In this case, the Abstract service is always the same and suitable Concrete services for the new user's requirements are searched for inside the set of Concrete services associated to the current Abstract one, without re-applying the whole matchmaking algorithm. This enhances performances of our system even if we work in an environment where user's requirements change very often.

Future work will investigate in more detail these aspects to better manage dynamicity of mobile environments and will study other application scenarios where the approach proposed in this work can be applied. The experimentation will be conducted in the context of ESTEEM Project platform (<http://www.dis.uniroma1.it/esteem/>).

## References

- [1] B. Benatallah, M-S. Hacid, C. Rey, and F. Toumani. Request Rewriting-based Web Service Discovery. In *Proc. of the Int. Semantic Web Conference (ISWC 2003)*, volume 2870, pages 242–257, Sanibel Island, FL, USA, October 2003.
- [2] D. Bianchini, V. De Antonellis, and M. Melchiori. Capability Matching and Similarity Reasoning in Service Discovery. In *CAiSE Int. Workshop on Enterprise Modeling and Ontologies for Interoperability, EMOI 2005*, Porto, Portugal, June 2005.
- [3] D. Bianchini, V. De Antonellis, M. Melchiori, and D. Salvi. Semantic-enriched Service Discovery. In *IEEE ICDE Int. Workshop on Challenges in Web Information Retrieval and Integration, WIRI 2006*, Atlanta, Georgia, USA, 2006.
- [4] D. Bianchini, V. De Antonellis, B. Pernici, and P. Plebani. Ontology-based Methodology for e-Service discovery. *Journal of Information Systems, Special Issue on Semantic Web and Web Services*, 31(4-5):361–380, June-July 2006.
- [5] The OWL Service Coalition. *OWL-S 1.1 release*, November 2004. <http://www.daml.org/services/owl-s/>.
- [6] X. Dong, A. Y. Halevy, J. Madhavan, E. Nemes, and J. Zhang. Similarity Search for Web Services. In *Proc. of the 30th Int. Conference on Very Large Data Bases (VLDB2004)*, pages 372–383, Toronto, Canada, 2004.
- [7] F. Donini, T. Di Noia, and E. Di Sciascio. Extending Semantic-Based Matchmaking via Concept Abduction and Contraction. In *Proc. of the 14th Int. Conf. on Engineering Knowledge in the Age of the Semantic Web (EKAW 2004)*, pages 307–320, Whittlebury Hall, UK, October 2004.
- [8] B. Fries, M. Khalid, M. Klusch, and K. Sycara. OWLS-MX: Hybrid OWL-S Service Matchmaking. In *Proc. of First International AAAI Symposium on Agents and Semantic Web*, Arlington, VA, USA, November 2005.
- [9] U. Keller, R. Lara, H. Lausen, A. Polleres, and D. Fensel. Automatic Location of Services. In *In Proceedings of European Semantic Web Conference (ESWC), Lecture Notes in Computer Science*, volume 3532, pages 1–16, 2005.
- [10] U. Keller, H. Lausen, and D. Roman. *Web Service Modeling Ontology (WSMO)*. WSMO Working Draft, March 2004. <http://www.wsmo.org/2004/d2/v02/>.
- [11] Nokia Web Services Framework for Devices A Service-oriented Architecture. <http://www.projectliberty.org/resources/whitepapers/Web.Services.Nokia.pdf>.
- [12] M. Wagner, T. Liebig, O. Noppens, S. Balzer, and W. Kellerer. Towards Semantic-based Service Discovery on Tiny Mobile Devices. In *ISWC04 Workshop on Semantic Web Technology for Mobile and Ubiquitous Applications*, Hiroshima, Japan, November 2004.