An Agent Oriented Approach for Modeling Web Services in Mobile Environments

HAMIDA Souraya[1], KAZAR Okba[1], AMGHAR Youssef[2]
[1]Department of Computer Science
Mohamed Khaider University
Biskra, Algeria
[2]National Institute of Applied Science of Lyon
LIRIS, Lyon University, France
{hamidasouraya, kazarokba}@yahoo.fr, youssef.amghar@liris.cnrs.fr

ABSTRACT: *Additively to the emergence of wireless networking, high technology and reliability of mobile devices have enabled the widespread use of mobile computing. Currently, this last is in vogue and booming. Today, the combination of mobile networks with Web Services called "Mobile Web Service", opening a promising horizon for the economy. They offer (Mobile Web Service) new personalized services to consumers on their mobile devices. In this paper, we propose the integration of two modern service technologies: Web Services and Mobile Agents. This integration enables wireless users to access and invoke semantically enriched Web services without the need for simultaneous and online presence of the service requestor. Furthermore, in order to improve the capabilities of Service registries, we exploit the advantages offered by the Semantic Web framework. Specifically, we use enhanced registries enriched with semantic information that provide semantic matching to service queries and published service descriptions. In addition, we will consider the information defining the context of the user and the Web service in order to best satisfy the user request.*

Keywords: Mobile Agent, Mobile Web Service, Semantic Web, Ontology, Context-Aware

Received: 13 November 2012, Revised 29 December 2012, Accepted 02, January 2013

1. Introduction

Nowadays the Internet has become a vehicle of service, rather than a static repository of information. Currently, several companies offer access to the web services.

The emergence of new wireless network as well as high technology and reliability of mobile devices put in apogee the use of the mobile computing. Now, this industry (wireless network) has become ubiquitous. The majority of consumers can't be separated from their phones. The use of the mobile device is in vogue and booming. Recently, the combination of mobile network and Web services called mobile Web services, open a promising horizon in economic area. Mobiles Web service offer new personalized services to consumers on their mobile devices.

Most applications on the Internet require the interaction of different entities across the network to exchange data and share tasks. Today, the model "*client* / *server*", where the trade is by remote calls through the network, is the most popular model. This

model has the disadvantage of increasing network traffic and requires a permanent connection between the client and the server which isn't the case of mobile devices that are exposed to the loss of the connection.

In this paper we introduce a novel framework for dynamic discovery and integration of semantically enriched Web services with Mobile Agents. In addition, we will consider the information defining the context of the user and the Web service in order to best satisfy the user request.

The remainder of this paper is organized as follows. In Section 2 we provide some background knowledge, whereas section 3 we discuss relevant prior work. In Section 4, we present in detail our proposed architecture. Section 5 provide details description of the entities used in our architecture. Then we provide an overview of ontologies (section 6). In section 7, development tools and we provide an illustrative example (section 8). Finally, Section 9 concludes the paper.

2. Background knowledge

2.1 Web services

The definition of Web Services from W3C is that «A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards» [1].

Compare with traditional Client/Server system, Web services is working not only within enterprise but between enterprises. Besides, Web services has a property such as program language independent, message driven communication, easily bound to different transports, and loosely-coupled.

2.2 Semantic Web services

The absence of a semantic in basic technologies of Web services poses a handicaps with WSDL description of a Web service; what doesn't make it possible to in no case the automation of the various tasks related to the Web services, such as: publication, invocation, discovery,… etc. To solve this deficiency, we have to amend the descriptions of Web services by other information, reusable, shareable and understandable by machines. This information is data and metadata, which make it possible to interpret the descriptions of Web services. In other words, it is the semantics of descriptions of Web services. So the semantic Web services are at the convergence of two important fields of research which relate to Internet technologies: "*the semantic Web and Web services*". Semantic Web services are a necessary evolution of Web services. They are based totally on the language of the semantic Web, particularly on the ontology.

In the area of Web services, ontology is used to annotate the descriptions of the functionalities of semantic Web services. Thus for a software agent can be aware of the semantics of a description of a Web service, it simply access to a domain ontology.

In literature, various approaches have been proposed to allow the realization of the semantic Web services. Among these works we have: WSDL-S [2], OWL-S [3], IRS-II [4] and WSMF [5].

2.3 Mobile Web services

Mobile Web services (m-services) are the application of Web services technology to the mobile environment [6]. A common and broad definition of m-services is the act of providing and consuming services through wireless handheld devices such as cellular phones, personal digital assistants (PDA), laptops or any other wireless enabled device. Usually expected to be the next wave of electronic services, m-services enable users to access networked services anywhere and anytime.

2.3.1 Classification of mobile Web services

According to Gehlen [7] mobile Web services can be classified into three classes. The classification takes place by means of the Web Service requestor and provider deployment.

The simplest case of a Mobile Web Service deployment is a mobile requestor (client) and a fixed provider (server), (see (1) in Figure 1). The main objective of this scenario, entitled mobile Web service Access, is to use already existing Web Services provided through the Internet on mobile devices.

Exchanging the requestor and provider roles of the fixed and mobile node, the mobile terminal now provides a Web Service and

a fixed node is requesting this service (see (2) in Figure 1).

In the last case (see (3) in Figure 1) mobiles are taking at the same time the roles of requestor and provider and communicate to each other in a Peer to Peer manner. This scenario strongly relates to the class of services that MSN, skype and googletalk has made so successful.

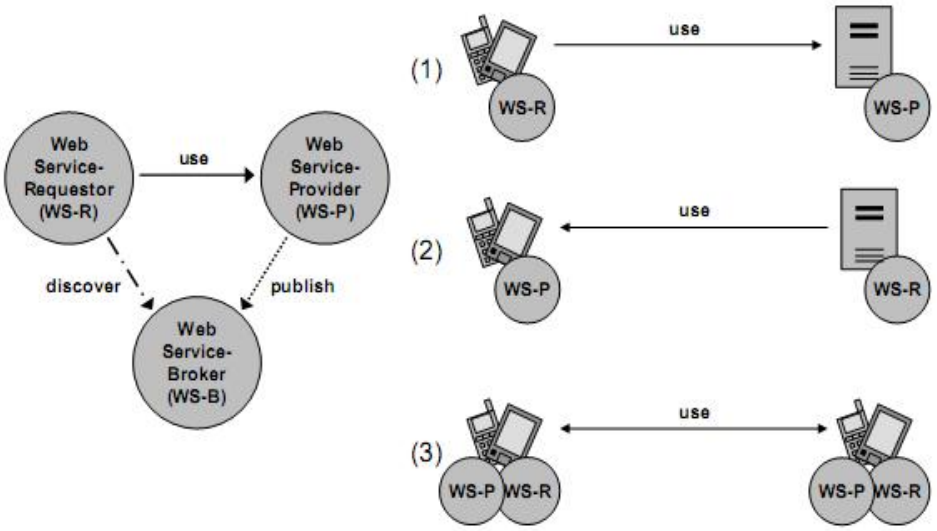

Figure 1. Classification des Services Web Mobiles

2.4 Mobile agent

In the literature, there are several types of agents (reactive agent, cognitive agent and hybrid agent), All these agents can be stationary or mobile. Stationary agents perform their tasks at the site that created them. The mobile agents can move from one site to another during their execution to access data or resources [8]. It is software entity that can move in a heterogeneous network to carry out various tasks on behalf of its user. It carries its actions in an autonomous, active, and reactive manner. Mobile agents have several advantages for mobile services [9]. Mobile agents run autonomously and asynchronously; they can adapt dynamically according to the execution environment; they can reduce network traffic; and they can be packaged with the specific tasks and sent to the destination host. When they arrive at their destination, they can carry out the tasks locally.

3. Related work

In this section, we describe the relevant contributions to the field of mobile Web service and multi-agent systems. Moreover, this section provides an overview of existing approaches that use context information to assist in the service discovery

In [10] and [11], BPEL (Business Process Execution Language) is used to form simple rules to describe mobile agent physical behaviors (migration and cloning). Such simple rules are separated from the integration logic, allowing for addition or change of physical behaviors without modification of the BPEL description. This separation is considered helpful in dealing with the dynamic environment of Web service, however, the discussed framework supports actions only in case of predefined events. The implemented rules don't consider dynamic events that might be generated during Web service invocation and mobile agent roaming. Moreover, and importantly, directory services and multicast protocols are assumed pre-existing and not discussed. The discussed framework refers only to interactions occurring among mobile agent and Web service without considering the interactions of the mobile agent and service registries that have equal importance in such a system. Finally, the system description doesn't include any implementation, hence benchmarking isn't considered.

Moreover, in [12] and [13], the authors propose a policy based framework for flexible management and dynamic configurability of agent mobility behavior in order to reduce code mobility concerns and support rapid mobile code-based service provisioning. Policies specify when, where, how and the parts of the agent that will perform a given task (migrating to a host and invoke a service). However, these models don't provide for the semantic description of the Web service involved in their systems. Other proposed frameworks adopt DAML-S for describing the Web service, thus, allowing for service capability search and matching

[14], [15]. However, the proposed systems are intended for fixed networks, and problems related to the wireless environments aren't considered.

However, mobile devices still have restricted capabilities (processing, storage space, energy consumption, stable connectivity, bandwidth availability). In order to address these shortcomings, a potential solution is context-awareness (by context we refer to the implicit information related both to the requesting user and service provider that can affect the usefulness of the returned results). Context plays the role of a filtering mechanism, allowing only transmission of relevant data and services back to the device, thus saving bandwidth and reducing processing costs. In [16] the authors propose a distributed context-aware service discovery system that is built on top of global service discovery architecture, GloServ that classifies services according to their business domain and associates a context-aware agent to each class. When an agent receives a query, it searches the appropriate classes for it, and forwards the query to the agents of those classes. One of those agents will then ask the user for his/her context and search the appropriate service. Doulkeridis and al [17] propose a system architecture for service discovery, based on a context-aware service directory. In [17], centralized service discovery architecture is proposed. It uses a context-aware service registry (i.e. enriched with context descriptions). To discover a service, two queries are used Qusr and Qcxt. Qusr extracts services using keywords specified by a service consumer. Those services will be transmitted to the context-aware service registry. Qcxt is then used to choose the services matching the consumers context. In those works, a Web service description language wasn't specified.

4. Proposed approach

This portion of study allows us to identify and highlight a systematic and progressive development of the presentation of the architectural study for the discovery of semantic Web services, using the Mobile Agent.

When a user is looking for a Web service, he uses first the UDDI directory. He becomes acquainted with the services available to UDDI defined in WSDL, diffused by the service providers. Once the user chooses a Web service defined, a link is established between these two entities via SOAP. In the process of discovery of Web services, we don't take into account the user context and the context of Web services. The goal is to allow the user to achieve the result of a Web service answering its context.

In our architecture, the web services are described semantically via the class ServiceProfile of the ontology OWL-S, which has been enriched by the parameters for taking into account the context of Web service.

In this context (proposed approach), the discovery of semantic Web services is accomplished in two steps. The first is limited to the search for semantic descriptions of Web services to satisfy the user request in terms of functional parameters (inputs, outputs). The second step is to select only Web services from the research phase, according to the parameters of the context (location, type of device of the user).

4.1 Step 1: semantic search

At this step of research, the Researcher Agent initializes the system of reasoning with ontology of the domain which will be used to calculate the degree of correspondence between the semantic concepts referred to input and output parameters specified in the query and those Web services.

Mobile Agent delivers to the Researcher Agent its query (search). This request is treated by application of the algorithm Matchmaking [18]. The algorithm allows the calculation of the degree of correspondence between the functional parameters of Web services and those requested by the user. The result is a set of Web services, satisfying the user query in terms of inputs and outputs.

4.2 Step 2: selection according to the context

The first step concerns the determination of all the Web services satisfying the request of Researcher Agent in functional parameters. The second step is devoted to filter this set while selecting Web services whose parameters of the context (location and type of device) are adequate with those determined at the request of the user. Selector Agent is responsible for this step. Contextual selection is accomplished in two phases.

4.2.1 Selection based on location of the user

The use of mobile devices and the restriction of use geographical web services, the location of the user has become an important

parameter to consider when selecting Web services. To this end, assume that the device of the user have service that determines the location of the user periodically through the GPS system. When user interacts with the system of discovery, Query Analyzer Agent determines the location of the user automatically.

The location information is then transmitted by Query Analyzer Agent to Selector Agent via Mobile Agent and Interface Agent Directory. Based on the ontology of location relative to a country or region, the Selector Agent can infer other information about the location of the user, such as region, city, etc.

```
Algorithm location
 Input: user.loc, ontology.loc,
        SWS ← {SWS1,SWS2,.., SWSn};
 Output: LSWS;
BEGIN
 LSWS ← ∅;
 For (i ← 1 to N) do
 {
  If (user.loc = SWSi.loc) then
    LSWS ← LSWS ∪ {SWS_i};
  Else
  {
   If(user.loc = getChildrenClass (SWSi.loc, ontology.loc) then
         LSWS ← LSWS ∪ {SWS_i};
  }
 }
 End.
```

4.2.2 Selection according to the type of device

The second phase is the selection of Web services adapted to the type of mobile device of the user. Each Web service description includes the types of devices supported by this last. The description of the type of device is concretized according to ontology of the device.

From the information of the type of device of the user, Selector Agent selects the Web services able to adapt the results transmitted to the characteristics of the device used

In the end, there are a set of Web services classified according to the semantic degree of correspondence, whose context (location, type of device) is adequate with that of the user.

5. Scenario of Proposed Architecture

According to the scenario of our architecture (Figure 2), the user of mobile device writes a request in Graphic User Interface GUI (step 1). It acts as an intermediary between the user of mobile device and other entities. Query Analyzer Agent interacts with mobile user via GUI. The role of this agent is to highlight this query the functional parameters (input and output). It makes it possible to supplement the request of the user by other information about its context (such as the device used and its location) in an automatic way (step 2). The latter are sent to the Mobile Agent (step 3). Then, the system creates a Mobile Agent (MA) that migrates to Web Service registry (step 4) to find the Web Services that meets the user requirements perfectly (step 5,6,7,8 and 9) taking into account the context of the user. The Web Service registry enables better research because it is enriched with semantic information. Mobile Agent, after acquiring the appropriate details (step 9), migrates to the service provider, invokes the Web Service, collects the results (step 10) and returns to the service requestor to deliver the results mobile user (step 11).

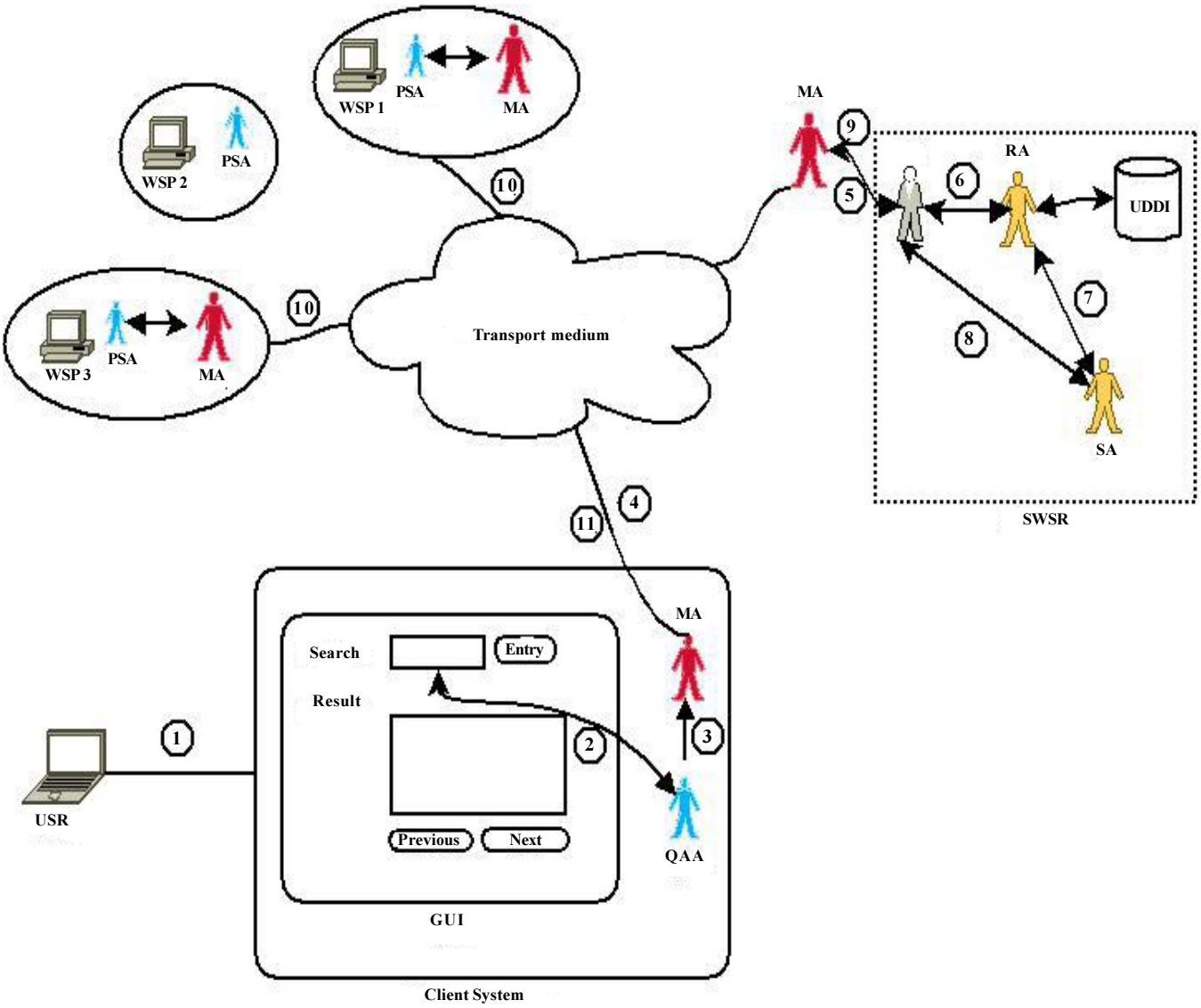

Figure 2. Proposed mobile agent architecture

In the presented scenario, just two Semantic Web Services corresponding to the service request. However, the Mobile Agent migrates and invokes these services (step 10). If the service request matched more than two services, the Mobile Agent migrates to all these matched Web services. The advantages of this scenario is that the Mobile Agent has an intelligence to invoke only the best matched service (s) and unnecessary services invocations are avoided leading to better network utilization. Mobile Agent can send its clones to Web Service provider, invoking services in parallel (and in this way we minimize the time of the invocation), instead of migrating in series to each of them. Connecting the mobile device user isn't required and can get the results in the future.

6. Detailed description of the entities used in the architecture

Our architecture includes the following components:

6.1 User Service Requestor

This is a user of mobile device that invokes a Web Service. It connects to the client system, and submits a request to text field of the GUI, then a Query Analyzer Agent traits it to exit the functional parameters (input and output). It also allows supplementation of the request user by other information about its context as: the device used and its location.

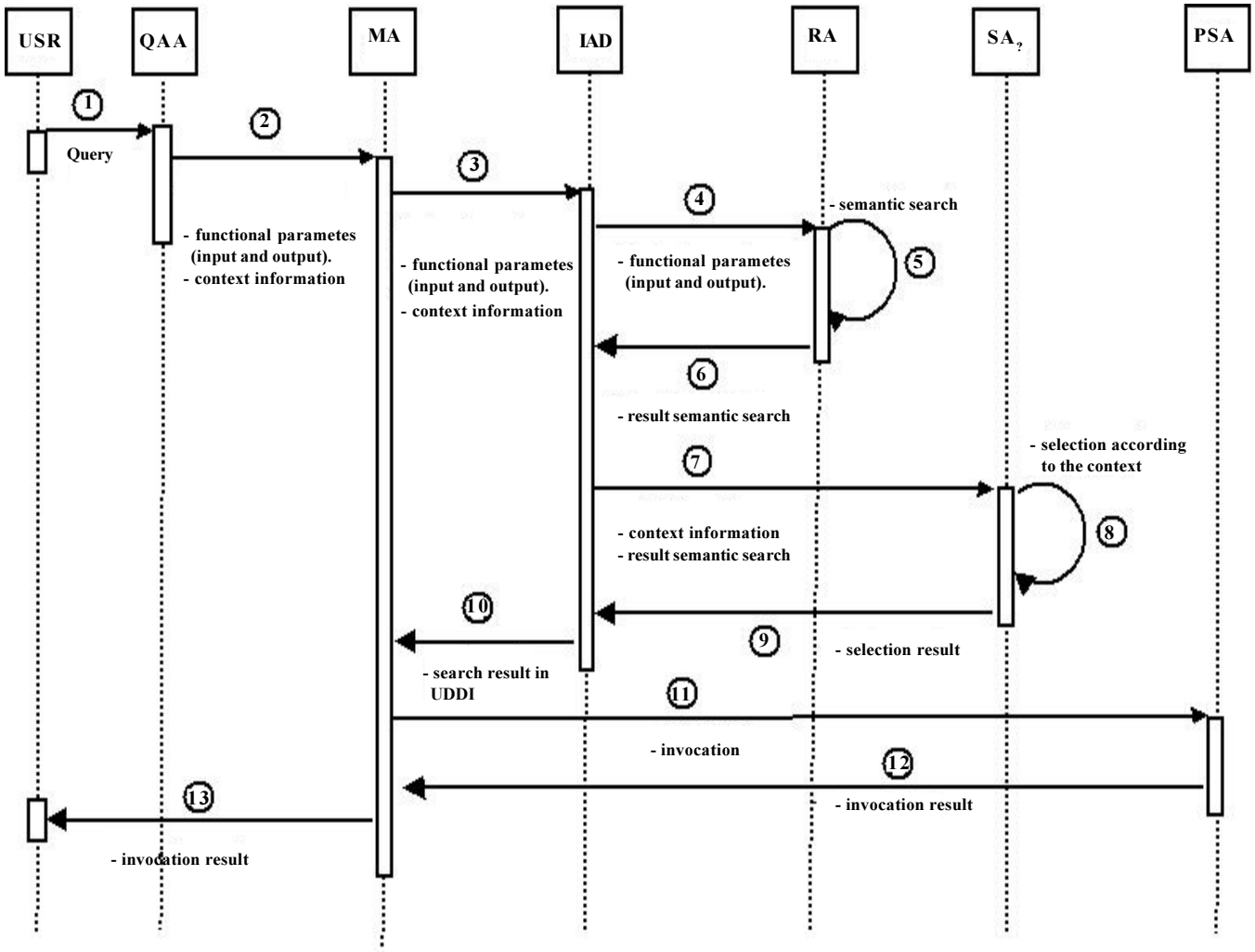

Figure 3. Diagram of discovery phase

6.2 Client System

It allows mobile users to interact with the system. It acts as an intermediary between the User service requestor and other entities in our architecture. It has three main components namely: Graphical User Interface, Query Analyzer Agent and Mobile Agent.

6.2.1 Graphical User Interface

Graphical User Interface (GUI) allows mobile users to interact with the system. The GUI plays the role of a interface between the end-user and the agents. It contains:

• **Text field:** for users to submit their query.

• **Menu:** to display the search result of the Mobile Agent. It can return multiple services in response to a user request. Two buttons are located at the bottom of the menu allows the user to go through all services

6.2.2 Query Analyzer Agent

It recovers the query lodged by the user in terms of functional parameters (input and output). It allows the supplementation of the user's query by other information relative to its context such as: the device used and its location. This information is automatically retrieved from the device of the user. These are sent to the Mobile Agent. The internal architecture of the Query Analyzer Agent consists of three modules shown in Figure 4.

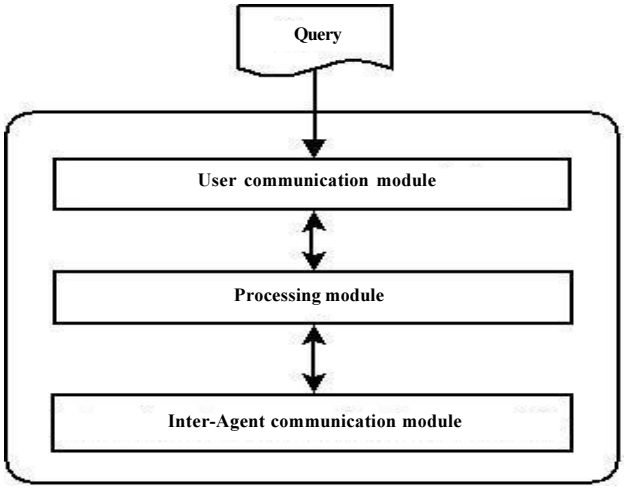

Figure 4. Architecture of QAA

6.2.3 Mobile Agent

The Mobile Agent (MA) is the representative of the user in the fixed network. It is capable of roaming, finding and executing Web services and delivering results to the user. Mobile Agent may also spawn clones that execute the selected Web services in parallel to minimize the total processing time. Clones can migrate and invoke Web services simultaneously and return results to the service requestor.

In our architecture, the physical behaviors of Mobile Agents (migration and cloning) are separated from the logic of integration. This makes it possible to add or change the physical behavior depending on environmental conditions, without changing the architecture. That is to say, the physical behaviors of the Mobile Agent are portable to any Mobile Agent platform (Jade or Grasshopper for example).

Mobile Agent has the following components (Figure 5):

• **Data state (DATA):** The data component contains information collected by the Mobile Agent from the Semantic Web Services invocations.

• **Code:** Mobile Agent needs a code to run.

• **Cloning and migration policies:** The migration and cloning policies component specifies the autonomous behavior Mobile Agent. It should be noted that the social behavior of Mobile Agent (migration, cloning) is separated from the integration logic and code implementation.

• **Policy Management component:** is responsible for the Mobile Agent external communication.

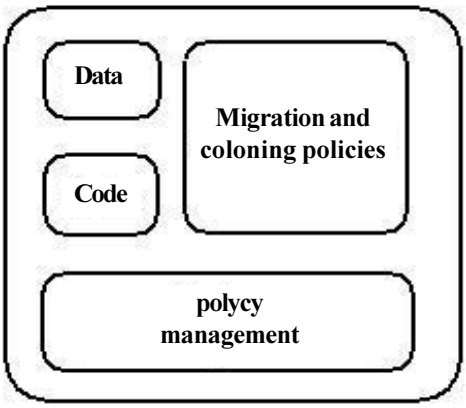

Figure 5. Architecture of MA

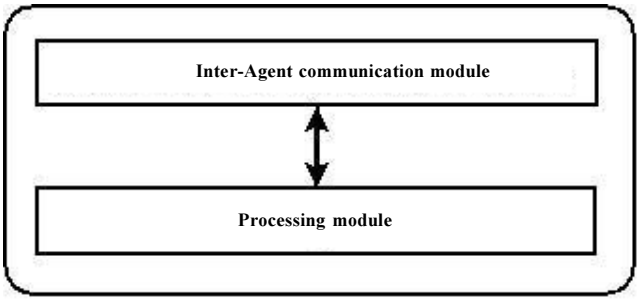

Figure 6. Architecture of IAD

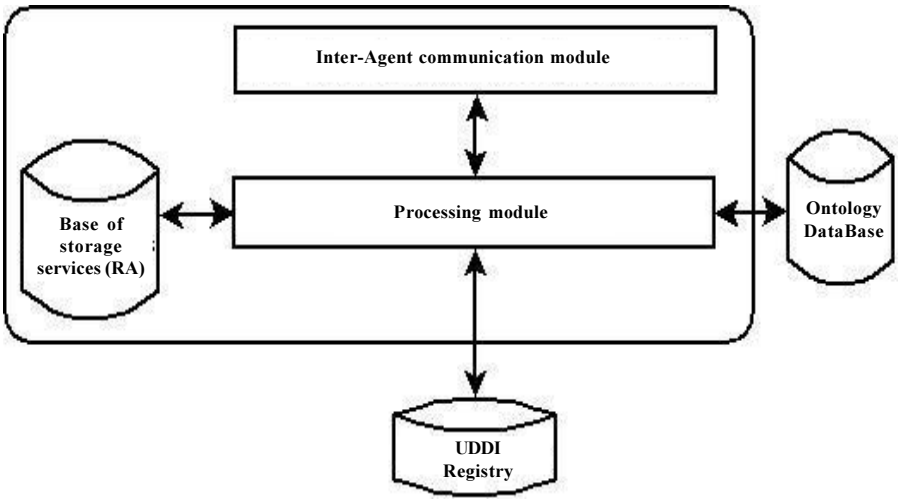

Figure 7. Architecture of RA

6.3 Semantic Web Service Registry
It consists of an Interface Agent Directory, Researcher Agent, Selector Agent and UDDI registry.

6.3.1 Interface Agent Directory
It is stationary agent that acts as intermediary between the Mobile Agent and other agents included in the Semantic Web Service Registry. Moreover, it acts as a supervisor, controlling interactions within the Semantic Web Service Registry.

6.3.2 Researcher Agent
Researcher Agent is a stationary agent. Its role is the discovery of descriptions of Web services satisfying the user's query semantically. The internal architecture of Researcher Agent is composed of:

• **Communication Module Inter-Agent:** to establish communication with other agents,

• **Processing module:** for implementing the matchmaking algorithm for mapping between the query and the descriptions of web services.

• **Base of storage Services (CA):** is used to store the semantic descriptions of Web services satisfying the user query.

6.3.3 Selector Agent
Selector Agent is a stationary agent. Its main role is to filter the set of Web services by selecting whose parameters of the context (location and type of device) are adequate with those determined at the request of the user (the processing module is responsible for this task). Base of storage Services (SA): is used to store the set of Web services satisfying the user query.

6.4 Web Service provider
Web Service provider provides a Web service which interested clients. The invocation of the service is realized by the Mobile

Agent on the semantic description of the service. In our architecture, the invocation of Web service (by Mobile Agent) is performed by the Provider Stationary Agent.

6.5 Provider Stationary Agent

Provider Stationary Agent is a stationary agent that resides in the host offering a certain Web Service. Its purpose is to wrap the functionality of the Web service. Provider Stationary Agent is created and maintained by the Web service provider. When the Mobile Agent migrates to the Web service provider, he gets results through Provider Stationary Agent.

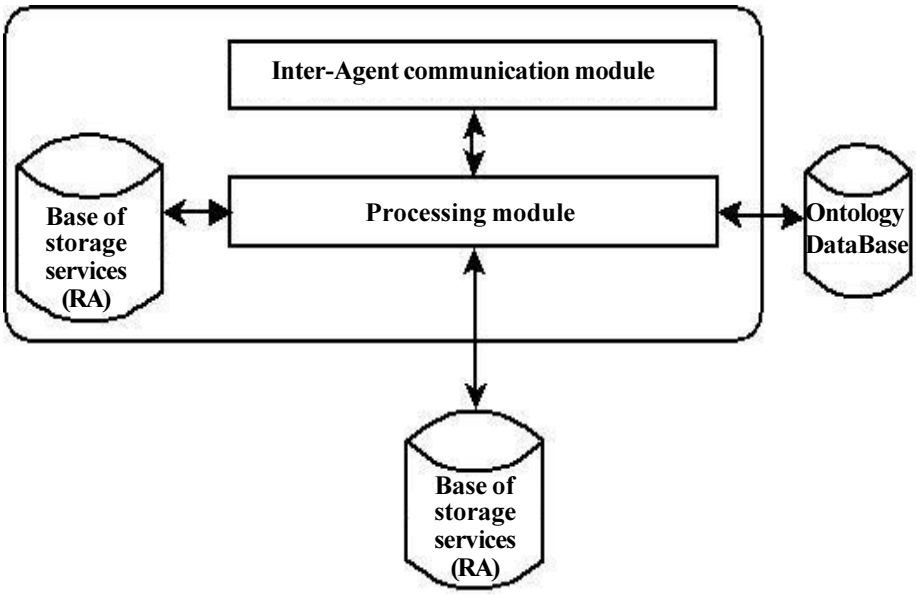

Figure 8. Architecture of SA

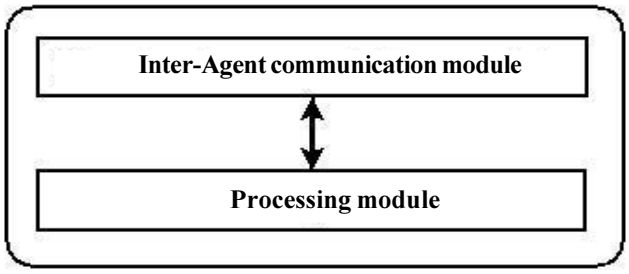

Figure 9. Architecture of PSA

7. Description ontologies of context information

Various models of representation of the context have been proposed. But we opt for the ontologies in the representation of the context. Our choice is justified by the fact that ontologies are regarded as being the most expressive model of representation. They provide a semantic representation between the various parameters that form the context and relationships. They are based on the languages of the semantic Web which allows sharing and reuse.

7.1 Context ontology of the user

The ontology of which we propose makes it possible to model the context of the user. This context is formed by a set of parameters (identity of the user (name, first name), his function (company, role, etc) and the type of device used (PDA, laptop, mobile phone, etc…)). These parameters are classified as follows:

7.1.1 Static parameters

They represent permanent dimensions in the personal life of the user. In our case two static aspects were taken into account: the name and the first name.

• **Name:** this name can be the usual name of the user or a pseudonym.

• **First name:** this aspect makes it possible to supplement the identity of the user where necessary.

7.1.2 Dynamic parameters

These parameters represent dimensions which evolve during the life of the user. The dynamic aspects taken into account are: location and the function of the user.

• **Location:** the use of mobile devices makes the location of the user useful and indispensable. In order to know this location in an automatic way GPS enters in plays to help us. Through the GPS we can know the following information: country, city, etc.

• **Function:** This parameter includes several aspects, namely: the company affiliation of the user, role, etc. Indeed when a service is accessed by a working group hierarchy, it is important that the user provides information on its role, because each member of a group doesn't have the same rights or can't perform certain operations service.

7.1.3 Parameters of the context of the user related to the device

This category of parameters represents the hardware and software constraints of the device with which the user accesses the service. Indeed, these parameters have become important with increasing use of mobile devices. The device is defined as a physical description via the following parameters: size of screen, types of connections, multimedia data accepted (image, text, video etc.). A second description of the capabilities is defined by software (platform and browser type).

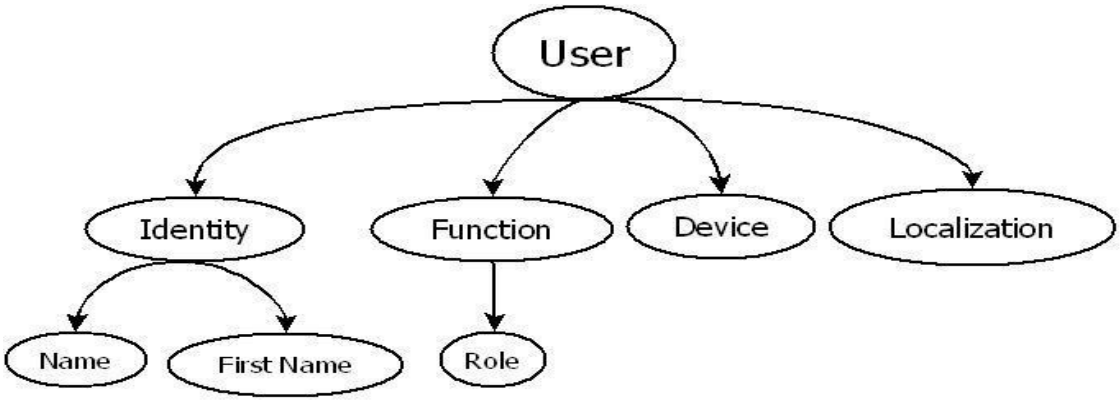

Figure 10. Graphic representation of user ontology

7.2 Context of a Web service

Our orientation towards the ontology OWL-S as ontology of semantic representation of Web services was the choice for our study. This one is justified by the fact that OWL-S is a standard OWL ontology of the semantic Web, giving it a future prospect to be a standard for the semantic representation of the Web services.

The discovery of Semantic Web Services (OWL-S) is based on the functional parameters of services. These parameters are specified at the class profile subclass of ServiceProfile of OWL-S ontology.

However, other non-functional parameters are determined at the level ServiceParameter class (subclasses of Profile) and which aren't considered in the discovery of Web services. These parameters form the context of the Web service.

To have efficient Web services (with adequate context with that of the user), we established a subtle combination of the nonfunctional parameters of the service with those of the functional parameters.

The choice of parameters that form the context of the Web service result in: the location of the service and the parameters of the types of devices whose the service can adapt to information that it diffuse.

7.3 Device ontology

Today, the user accesses the Web service through a PC, a laptop, PDA, etc. To adjust information provided by the Web service

to the capabilities of the device used, we have implemented an ontology of device description. This ontology provides a common vocabulary for service providers to describe the various types of devices that can support the data sent by Web services. It consists of: the class of description of the device, the class hardware and the class software.

The class of description of the device provides the basic information about the connection device such as the type, model and manufacturer name.

The Hardware class fixes the size and screen resolution of the device and the display type (color or not).

Software class identifies the type and version of operating system and browser used, respectively.

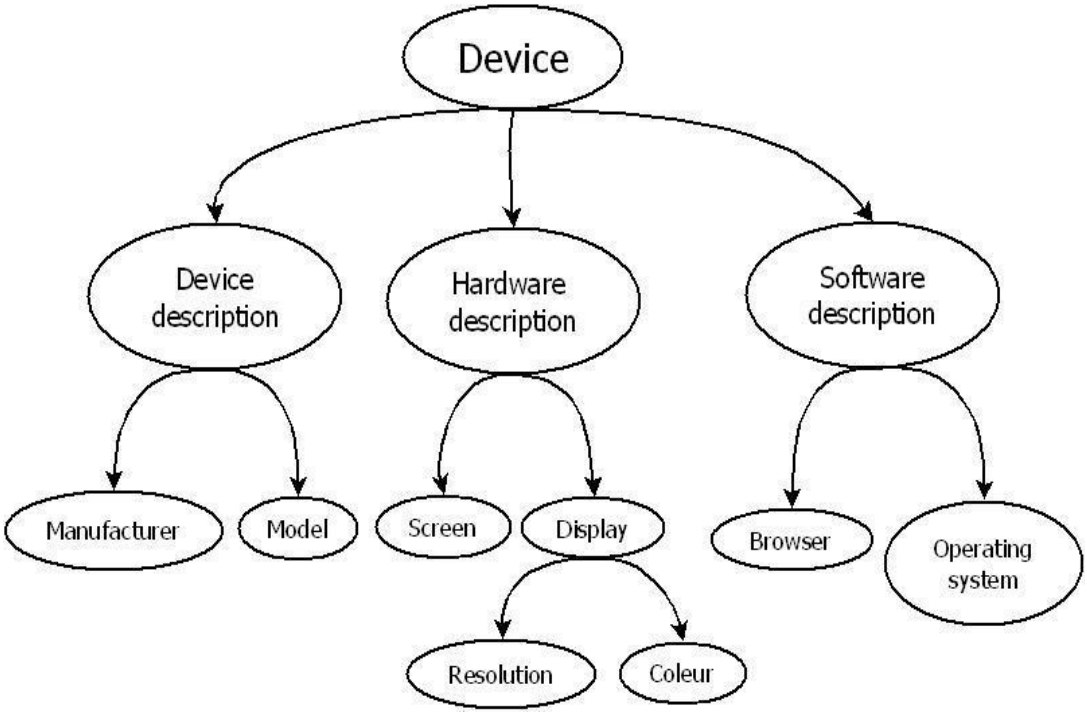

Figure 11. Device ontology

8. Development tools

The development of the proposed architecture is made of four main steps:

• Development of ontologies proposed (Different ontologies that we have proposed are realized through the ontology editor Protégé 2000),

• Development of publication interface of semantic descriptions OWL-S and UDDI registry,

• Development of user interface for Web services discovery,
• Development and deployment of multi-agent system (using JADE) and the reasoning system RACER (The reasoning tool chosen is the reasoner RACER).

9. Illustrative example

In this section, we give an illustrative example to clarify the discovery of mobile web services. For this, we assume that there are four Web services sales: SWS1, SWS2, SWS3 and SWS4 published on the Web. Their functional parameters (inputs, outputs) are:

• SWS1 have two inputs "*bus*" and "*parts*"; and one output *"price*" ;

• SWS2 have two inputs "*car*" and "*parts*"; and one output "*price*";

• SWS3 have two inputs "*car*" and " *parts*"; and one output "*price*";

• SWS4 have two inputs "*unit*" and "*material*"; and one output "*price*".

On the other hand, we assumed that the location of SWS1, SWS2, SWS3 and SWS4 is Paris, Biskra, Montreal and New York respectively, and these type of device used is PDA

We assumed also, a mobile user who is looking for a web service that takes as input "*Car*" and "*Parts*" and offer as output "*Price*". Moreover, the location of user is Biskra and the user connects the client system using PDA. Figure 12 shows an ontology example. In order to satisfy the user request, our method of discovery is accomplished in two steps.

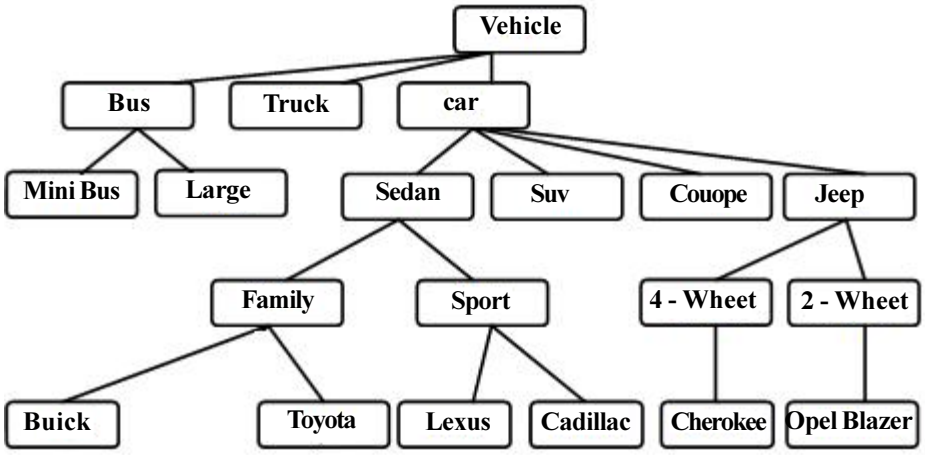

Figure 12. Example of vehicle ontology

9.1 Step 1: Semantic Search

At this step, the Researcher Agent initializes the system of reasoning with ontologies of the domain. Which will be used to calculate the degree of correspondence between the semantic concepts referred to input and output parameters, specified in the query and those Web services. In our example, there are two web services satisfy the user query with are SWS2 and SWS3.

9.2 Step 2: Selection according to the context

This step is devoted to filter the set of Service Web while selecting Web services whose parameters of the context (location and type of device) are adequate with those determined at the request of the user. In our example, the device of the user is able to adapt the result of two Web services (SWS2 and SWS3). But the location of the user is different than the location of SWS3. Finally, there is one semantic Web service (SWS2) satisfies the user query, whose context is adequate with that of the user.

10 Conclusion

In this paper, we presented an approach using the Mobile Agent to the discovery of mobile Web service. The developed system adopts a register enriched by semantic information providing a match semantics of the query with the descriptions of Web services. Given the diversity of users and the conditions of accession to Web services, other parameters must be taken into account in the discovery, such as: the type of device used (PDA, laptop, etc..), The user preferences, location of the user, etc.. All these parameters form a particular context of use. In our study we described the various parameters of the context of the user and services across ontologies (ontologies of user context, Web service and parameters of device). The advantages of our system are: (1) a single interaction, users can invoke a set of services with only one interaction with the fixed network, (2) users can be disconnected during the services discovery. (3) our architecture ensure the delivery of service results to the user, (4) the results are adequate with the user's mobile device, (5) the dynamic behavior of Mobile Agent improves system robustness and fault tolerance, (6) our architecture ensure better resource utilization, (7) our system is open, allows us to add new services, staff and users.

In future work, we will carry out extensions to the architecture in question to allow the contextual composition of mobile Web services using the mobile agent.

References

[1] Booth, David., Haas, Hugo., McCabe, Francis., Newcomer, Eric., Champion, Michael., Ferris, Chris., Orchard, David. (2004).Web services architecture. W3C.

[2] Akkiraju, R., Farrell, J., Miller, J., Nagarajan, M., Schmidt, M., Sheth, A., Verma, K. (2005). Web service semantics: Wsdl-s. Technical report, IBM.

[3] Martin, David., Burstein, Mark., Hobbs, Jerry., Lassila, Ora., McDermott, Drew., McIlraith, Sheila., Narayanan, Srini., Paolucci, Massimo., Parsia, Bijan., Payne, Terry., Sirin, Evren., Srinivasan, Naveen., Sycara, Katia. (2004). Owl-s: Semantic markup for web services. Technical report, W3C.

[4] Motta, Enrico., Domingue, John., Cabral, Liliana., Gaspari, Mauro. (2003). Irs-II : A framework and infrastructure for semantic web services. *International Semantic Web Conference*, p. 306–318.

[5] Fensel, D., Bussler, C. (2002). The web service modeling framework wsmf. *Eletronic Commerce: Research and Applications*, 1 (2) 113–137.

[6] Farley, P., Capp, M. (2005). Mobile web Services. *BT Technology Journal*, 23 (2) 202-213.

[7] Gehlen, Guido. (2007). Mobile Web Services - Concepts, Prototype, and Trafic Performance Analysis . Thèse de doctorat, Université de RWTH Aachen, Allemagne.

[8] Hacini, Salima. (2008). Sécurité des Systèmes d'Information : Mise en œuvre de la confiance et de l'adaptabilité pour la protection de l'agent mobile. Thèse de doctorat, Université de Mentouri, Constantine.

[9] Lange, Danny B., Oshima, Mitsuru. (1999). Seven Good Reasons for Mobile Agents, *Communications of the ACM*, 42 (3) 88-89.

[10] Ishikawa, Fuyuki., Yoshioka, Nobukazu., Tahara, Yasuyuki., Honiden, Shinichi. (2004). Mobile Agent System for Web Services Integration in Pervasive Networks, *In*: the proceedings of the International Workshop on Ubiquitous Computing (IWUC 2004), p.38-47, Porto, Portugal.

[11]Ishikawa, Fuyuki., Yoshioka, Nobukazu., Tahara, Yasuyuki., Honiden, Shinichi. (2004). Behavior Descriptions of Mobile Agents for Web Services Integration. *In*: Proceedings IEEE International Conference on Web Services (ICWS 2004), p. 342-349, SanDiego, CA.

[12] Montanari, R., Tonti, G., Stefanelli, C. (2003). A Policy-based Mobile Agent Infrastructure, *In*: the proceedings of the 3rd IEEE International Symposium on Applications and the Internet Workshops (SAINT03) IEEE Computer Society Press, Orlando, USA.

[13] Montanari, R., Tonti, G., Stefanelli, C. (2003). Policy-based Separation of Concerns for Dynamic Code Mobility Management, *In*: the proceedings of the 27th International Computer Software and Applications Conference, (COMPSAC'03), *IEEE Computer Society Press*, Dallas.

[14] Gibbins, Nicholas., Harris, Stephen., Shadbolt, Nigel. (2004). Agent based Semantic Web services, *Journal of Web Semantics*, vol. 1

[15] Kagal, Lagana.al. (2002). Agents making sense of the semantic web, *In*: the proceedings of the First International Workshop on Radical Agent Concepts, (WRAC 2002), McLean, VA, USA.

[16] Arabshian, Knarig., Schulzrinne, Henning. (2006). Distributed context-aware agent architecture for global service discovery. *In*: SWUMA.

[17] Doulkeridis, Christos., Loutas, Nikos., Vazirgiannis, Michalis. (2005). A System Architecture for Context-Aware Service Discovery. *In*: International Workshop on Context for Web Services CWS-05.

[18] Paolucci, Massimo., Kawamura, Takahiro., Payne, Terry R., Sycara, Katia P. (2002). Semantic matching of web services capabilities. *In*: ISWC '02: Proceedings of the First International Semantic Web Conference on The Semantic Web, p. 333–347, London, UK.