

# AGENT APPROACH FOR SERVICE DISCOVERY

**Berdjough Chafik**

Centre de Formation Professionnelle El-Meghaier  
Wilaya El-OUED, ALGERIE  
**Berdjough2006@yahoo.fr**

**Kazar Okba**

Département d'informatique  
Faculté des sciences et sciences de l'ingénieur  
Université Mohamed Khider 07000 Biskra, ALGERIE  
**kazarokba@yahoo.fr**

## ABSTRACT

Current technologies for Web Services are based on syntactical descriptions and, therefore, lend themselves to only limited amount of automation. Research efforts in Semantic Web Services, such as OWL-S, try to overcome this major deficiency by providing a complete semantic description for Web Services and their related aspects. In this paper we demonstrate the use of agent technology, Web Service standards and Web semantic to enable automatic service discovery.

**Keywords:** Web service, multi-agents system, semantic web

## 1 INTRODUCTION

Today, the Web is just an enormous warehouse of text and images, its development has also made it a service provider. The concept of "Web service" is essentially an application available on the Internet by a service provider and accessible by clients through standard Internet protocols. In essence, Web services are autonomous software components and self-descriptive and thereby constitute a new paradigm for application integration.

Currently, Web services are implemented through three technology standards: WSDL, UDDI and SOAP. These technologies facilitate the description, discovery and communication between services. However, this basic infrastructure does not yet allow Web services to keep their promise of a largely automated management. This automation is essential to meet the requirements to scale and to reduce development costs and maintenance services. Basically, it must accommodate a means for describing Web services in a manner understandable by a machine.

The Semantic Web [1] is a vision of future Web in which information has a semantic understandable by a computers. Applied to Web services, the principles of the Semantic Web should enable to describe the semantics of their functionality, and reasoning are therefore induced a proposal for automation of various tasks of their life cycle.

Combining the technologies of Web services and Semantic Web has led to the concept of semantic Web services.

The discovery of Web services is an emerging area of research. Initially, the discovery is made in the UDDI registry, it is based primarily on research syntactic WSDL descriptions of Web services. But with the development of Semantic Web technologies, the techniques for discovery have become essentially semantic. This semantics is provided through one of ontologies important technologies of the Semantic Web. Thus, software agents can be developed to reason about these ontologies, making the discovery of Web services dynamic and automatic. In this work, we propose an approach to discovery of semantic web services using agent technology and ontologies.

## 2 EMERGING TECHNOLOGIES

### 2.1 Semantic Web and ontology

The Semantic Web [1] is an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation. In order to realize the Semantic Web vision a set of standard technologies have been defined (the Semantic Web layered architecture):

- the Syntactic Layer (XML),
- the metadata layer (RDF/RDFS),
- the semantic layer (Ontology languages),

- the Logical Layer (automatic reasoning),
- Proof and trust layer (proof)

The technology object which is fundamental for the realization of the Semantic Web is the ontology.

Ontology is term borrowed from philosophy meaning "systematic explanation of existence". Ontology is similar to a dictionary or glossary but with a large and detailed structure, that allows machines to process its contents.

Bertrand [2] defines ontology as "These are formal representations of domain knowledge in the form of terms with semantic relations".

In the Semantic Web, the ontology allows the user during a Web search to access not only to documents related to keywords in the query, but also those that are related ontologically (semantically) to them, this makes the search more relevant. It aims to describe concepts and relationships that bind them, and with deduction rules to make them more understandable and usable by the different agents (human or software).

## 2.2 Ontology Web language for services (OWL-S)

OWL-S (formerly known as DAML-S) [3] is ontology for web services and it has been developed to enable the following tasks: automatic service discovery, automatic service invocation and automatic service composition. The service discovery is improved using ontologies because the information needed to perform this task is expressed using a machine-processable form. A computer can access the description of a web service and it can know exactly what the service does thanks to the shared concepts contained in the ontologies used in the description.

A service described using OWL-S provides three types of knowledge: Service Profile, Process Model and Service Grounding. The ServiceProfile describes what the service does, including functional information such as inputs, outputs, and other non-functional information (category, classification). It is normally used during the automatic discovery of Web services. The Process Model describes how the service works; it is an abstract vision of the service operation. Finally, the ServiceGrounding tells how to access the service; it contains all the information related to the real implementation of the service and is used to invoke it automatically.

## 2.3 Role of multi-agent systems

A multi-agent system (MAS) is a loosely coupled network of software agents that interact to solve problems and function beyond the capabilities of any singular agent in the set-up. The agents in a multi-agent system may be distributed on different

computers (or nodes), where each computer owns its resources.

The characteristics of MAS are that each agent has incomplete information or capabilities for solving the problem and, thus, has a limited viewpoint; there is no system global control; data are decentralized; and computation is asynchronous[4].

An MAS has the following advantages over a single agent or centralized approach like distributed systems [5]:

- An MAS distributes computational resources and capabilities across a network of interconnected agents. Whereas a centralized system may be plagued by resource limitations, performance bottlenecks, or critical failures, an MAS is decentralized and thus does not suffer from the "single point of failure" problem associated with centralized systems.
- An MAS allows for the interconnection and interoperation of multiple existing legacy systems. By building an agent wrapper around such systems, they can be incorporated into an agent society.
- An MAS models problems in terms of autonomous interacting component-agents, which is proving to be a more natural way of representing task allocation, team planning, user preferences, open environments, and so on.
- An MAS efficiently retrieves, filters, and globally coordinates information from sources that are spatially distributed.
- An MAS provides solutions in situations where expertise is spatially and temporally distributed.
- An MAS enhances overall system performance, specifically along the dimensions of computational efficiency, reliability, extensibility, robustness, maintainability, responsiveness, flexibility, and reuse.

## 3 OUR APPROACH

The proposed architecture is an extension of service-oriented architecture (SOA). This architecture is based on agents for discovering Web services.

This architecture (shown in Fig. 1) incorporates software components and operating a domain ontology is used during the discovery phase of Web services, it facilitates the automatic discovery of services since it allows to refine the search process which matches a request and service offerings. The use of this ontology allows the implementation of filtering mechanisms (comparison) between a request and offers to implement anything other than simple equality.

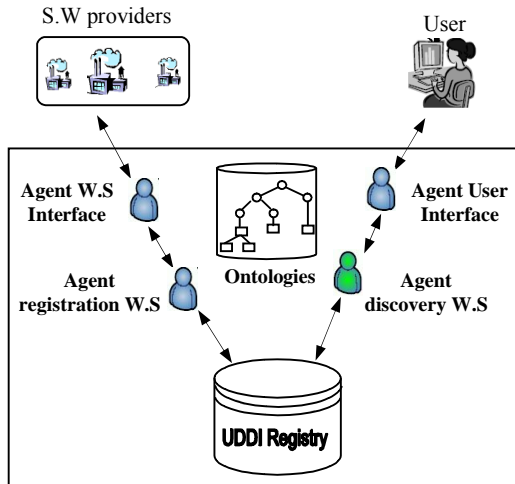


Figure 1: Proposed multi-agent architecture

### 3.1 Architecture description

#### 3.1.1 Agent Web service interface

This software agent acts as an interface between the system and the Web service provider, such that for each Web service agent is associated. Agent Web service interface allows the recording of the description on the Semantic Web service. Moreover, it allows updates information on the Web service.

The internal architecture of the agent Web service interface consists of three modules and a registry backup, as shown in Fig. 2.

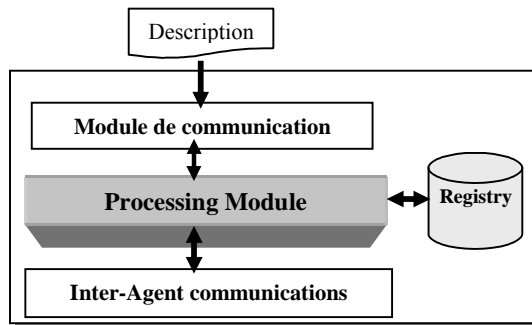


Figure 2: Architecture of the Web service interface agent

#### 3.1.2 Agent registration Web services

The role of this agent is the preservation of semantic descriptions of Web services in the UDDI registry, it contains two modules and an interface as shown in Fig. 3.

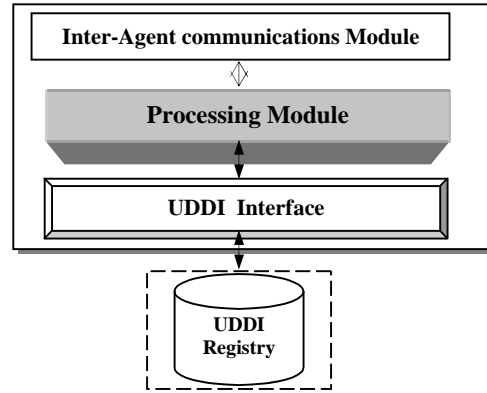


Figure 3: Architecture of agent registration Web services

#### 3.1.3 Agent user interface

The agent user interface is the gateway to query the system. It provides the user with the form to do a query.

This is the agent who will initiate the discovery, by issuing to the Agent discovery, a request consists of inputs, outputs, a reference ontology domain to use (e.g. the ontology of tourism) and presents the results tailored to the preferences of users after treatment.

The internal architecture of the agent user interface is composed of three main modules and a registry backup as shown in Fig. 4.

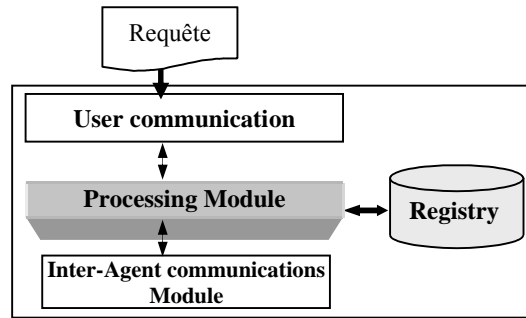


Figure 4: Architecture de agent user interface

#### 3.1.4 Agent discovery Web services

It is a software agent that allows the discovery of descriptions of Web services satisfying the request sent by the agent user interface on the semantic.

The internal architecture of the discovery agent is composed of two modules and a base of storage services for storage the semantic descriptions of services provided by UDDI as shown in Fig. 5. They are as follows:

- **Inter-Agent communications Module** : He received from the agent user interface the query in the form of a message and after that, he calls the module of treatment. It also receives requests for transmission of messages from the module of







