# UDDI Service in eGovernment Environments

Luis Álvarez Sabucedo and Luis Anido Rifón
*Universidade de Vigo, Dpto. Telemática*
*Lagoas Marcosende, Vigo (Spain)*
*+034 986 81 70 43*
*{lsabucedo,lanido}@det.uvigo.es*

**Abstract**: Nowadays, Ontology based projects are coming from promising ideas to useful project where a lot of resources are being devoted to. This new support brings up not only new functionalities to empower projects but also new challenges to overcome. The weakest point in this infrastructure is UDDI services. Currently these services need some improvements in order to meet ontology-based technologies full potential. This paper deals with these services and how improvements may be deployed in eGovernment environments.

## 1 Introduction

During last years we are witnessing an amazing development of eGovernment projects and initiatives. In this tendency, several web-based initiatives and ad-hoc solutions are currently being deployed. A quick review of current state of art in this area will lead us to conclude that several projects and solutions are providing support to citizens in the most usual transaction. Even more, several developed platforms aimed to improve the eGovernment status are profusely arising all over Europe.

Taking current researches into account and the actual working lines, we may foresee in close future global project where final goals are addressed and fulfilled in an integral perspective in eGovernment framework due to the vast amount of resources that are being devoted to. Nevertheless, several steps remain in the way to this major goal. Most of them are related to the adoptions of new paradigms and technologies currently in developing stage and the provision of front and back-office projects that undertake solutions in an integral way.

One of the major enhancements for eGovernment projects is related to the inclusion of semantic support for operations. This feature will provide us a handful of functionalities that will become a must in any solution within a few years. This paper will deal with in the inclusion of ontology-driven architecture in eGovernment projects to provide the required infrastructure for the accessing service to agents in the network.

This paper is organised as follows: firstly, we will discuss an architecture proposal for a network where eGovernment services may be provided. Later we will present the basis for UDDI mechanism and following a brief explanation of the basis of OWL-S. The paper is completed with an overview of the OWL-S server itself and how UDDI Server may be used in this context. Finally, conclusions on the application of presented ideas on eGovernment are provided.

## 2 Architecture proposal

In order to deal with eGovernment projects we must be aware of several concerns that may not be patent in other environments and that will condition the approach to the problem:

- Costumers for projects are also the stakeholders of projects as they pay it by its taxes. So it is a must to provide contents accessible for all of them.
- Procedures must be auditable for both citizens and administration.
- Data integrity and data confidentiality are under strict constraints.
- Data interchange may involve a large chain for data processing and must be kept under control on each single step.

Bearing in mind these ideas, we foresee architectures for these projects with the following characteristics:

- Open Source. Software elements must be available for every user involved in the project and no assumption may be done about the operative system or required tools. Of course, it is not acceptable to force the adoption of some certain programs when there are available free and open alternative.
- Adoption of open standards. By using open standards in every layer of projects, it is possible to guarantee the maintenance and the support for new improvements in the scope of constant researching.
- Interaction of any agent. The interaction must be supported even with agents not provided by the administration. Thus, it will be possible to develop agents by anybody that may become part of the system. This feature will largely increase eParticipation as you allow citizens to really take part of Public Administration by mean of their contributions.
- Support for multiplatform devices. Agents in the system may use different network supports, i.e., agents may use wired network, WAP devices, Wi-Fi devices or any other support as they work with open standards.
- Ontology-driven. By using this feature, we will achieve a high level of interoperability as machine will be able to process data with little or no human participation. This also provides interesting advances on server composition and mechanization of procedures by mean of autonomous systems. This feature requires a lot of efforts to implement it in a proper way.

The general architecture of the eGovernment network is presented in Figure 1. We can outline the most outstanding elements on the figure:

– Service points. These elements are the final responsible for the service provision. They are on charge for the execution of services. This role is usually assigned to PAs (Public Administrations).

– Agents. A wide range of agents may be involved on the interchange of information. As a matter of fact, support for service should be completely independent of user’s platform. Thus, we will support an adapted service for each client: WWW-based clients, stand alone clients, mobile clients, etc. Of course the nature of the client will condition the way this service is provided. Anyway, we must bear in mind that agent’s requirements and facilities may change from one to another.

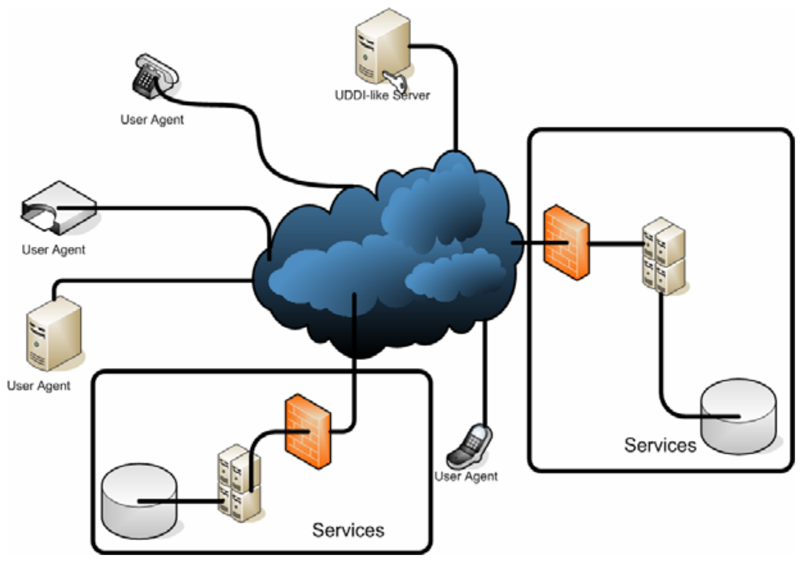

**Figure 1. Network model**

- UDDI-like server. We need in this architecture some elements that solve the problem of locating the server. In this way, we should provide a *yellow page server*. This service is nowadays being carrying out by UDDI server. These servers are responsible for locating the proper server to each request made by the user by looking up in an internal table of data. Nevertheless, this service will no fulfil our requirements as will discuss later on.

In this architecture, we find a weak point if no enhancements are made on the current-fashion UDDI Server. Main concerns are related to the provision of ontology-driven services and security issues among other secondary considerations. Thus, we must review some interesting topics related to this server:
- Semantic support. We find no support for ontology-driven information as this kind of servers are not oriented towards semantic data.
- Security. As these servers are oriented to be just a yellow pages service, there is no support for several services that should be provided in eGovernment environments such as non-repudiation on origin or integrity.
- Lack of support for particular specifications. eGovernment projects requires the possibility to express some particular concerns and requirements that are no available in the UDDI protocol.

For the provision of these services we must make a review on UDDI servers to point out the weak points we should enhance.

## 3 UDDI Overview

The Universal Description, Discovery and Integration (here after UDDI [4]) is an open initiative supported by OASIS[1] aimed to encourage the provision of service for the discovery and interaction among different entities involved in any particular service. We can distinguish three different services on a UDDI server:
- White Pages. This service deals with name and contact information.
- Yellow Pages. This service provides a categorized way to organize the provided services.
- Green Pages. This service addresses technical details about the service.

UDDI is expected to be a key component on the core of Web Services. This service is expected to be addressed by using SOAP[6] and, as final result, clients of this services will get a information in WSDL[7] format that will let them to properly invoke services.

To provide these services, UDDI uses mainly two data structures: `businessEntity` and `businessService`. The `businessEntity` component will provide information to be managed by the white pages service such as name, description and contact information. On the contrary `businessService` will carry the information for the green pages services. Thus, we can find information about the service actually provided. In this mission, we can find a very interesting element that will play a main role: `tModelInstanceDetails`. The former one is the responsible for the provision of how the service must be addressed; this mainly is done by mean of a WSDL specification of the features for the service.
To fulfil services provided by the yellow pages, UDDI provides the `categoryBag`. In short, this is an entity that may be included in both `businessService` and `businessEntity`. This element allows the system to categorize the functionality of the service in a systematic way.

For the sake of brevity we will not get deeper into the UDDI specification, for more information [5]. But from the presented features, we can state that UDDI provides a quite useful support for the discovery and invocation of services. Nevertheless, some lacks and drawbacks are also patent:

– No security concerns were taken into account. This service does not provide support for reliable communications or certifications for the submitted information. Although several researching groups are currently dealing with the formers by improving SOAP security features; some additional points should be considered for the inclusion of this element on a eGovernment structure as later on presented.
– No support for semantic contents is possible. UDDI servers are designed just for the provision of information based on XML contents[10] and no ontology-driven mechanisms are considered. Thus, it is no possible the automatic discover of services by machines as it actually happen on semantic services.

## 4 OWL-S for service description

Ontology Web Language – Service (here after OWL-S[3]) is an ontology for describing Web Service with semantic information. As far as this technology spreads, services are described not just by human understandable information but also by machine readable data. This new perspective is getting momentum in the environment of Web Services and systems have available infrastructure for delivering semantic-based interaction. As result of this intelligent data interchange, systems may provide some interesting advantages from previous data-based systems:

- Web Service discovery and invocation. The difference from previous schemas such as WSDL and UDDI registry relays on the fact that now these operations may be actually undertaken by machines themselves with little o no human interaction.
- Web Service composition. Services can also be composed in non trivial way by using the semantic data that allows machines to connect outputs from a service as input for the next process in the production chain.

To accomplish these promising goals, OWL-S establishes three different levels for data related to Web Services:

- Service Profile
- Service Grounding
- Service Model

Service Profiles deals with the question “What it does?” This layer informs present information of the function of the Web Service. Indeed, each service publish on the server has its own instance the Service to express its own functionalities. The Service Grounding is related to the “how” the service can be accessed. Thus, this layer provides support for the semantic definition for data and also specifies protocols, message formats, etc. Finally, the service model is concern on present “how it actually works”. This layer provides information of low level to express how data must be encapsulated. It is based on WSDL formats to express this information. For the sake of brevity we suggest [5] for more information.

## 5 OWL-S Servers: new approach

As previously stated, there is a list of topics that should be taken into account when providing semantic support for eGovernment projects. To overcome this situation, we propose an OWL-S Server, an active network component responsible for providing data about services in OWL-S format that match constraints from client by using a reasoning module.

It is important to keep in mind that the aim for this contribution is not to develop a broker for semantic services, but a server to locate service with semantic description. We are trying to provide a server responsible for service localization taking into account semantic descriptions. We can not consider this as a broker as it does not perform operation on the behalf of other agents but finding its requested services.

Besides of the operations available on UDDI servers, our OWL-S server provides also several additional services: certification of the submitted data, support for security capabilities and additional facilities for advanced services in collaborative environments.

We will present how an interaction between a client device and the server should go on in order to deploy an access service for any further service. Firstly, let make some assumptions: there is a client in the system that needs to deliver an operation in this framework. This operation is expressed in the terms of a given ontology that we will assume suitable for the problem and known by the full environment. Under these constraints, the operation will be made of the following steps:

1. The client begins the operation by requesting information about how to invoke the operation in terms of OWL-S data. The semantic information about the service and the needed requirements is provided to the server.
2. Once the request is in the server, the Logical Reasoner deals with it. By using an inference engine, the system gathers the proper information requested and matches it with the already available data. If the request can be fulfilled according to the available information and limitations from the environment, the petition will be answered.
3. The client, the agent that begins the interaction, will be able to request the final service addressing to the real service server due to the information obtained from the OWL-S server. Of course, the server responsible for implementing these operations may be placed somewhere else on the network as an entry point for services themselves

There is two particular points we would like to make clear as they are fundamental for the proper working of the prototype:

- When speaking of matching the proper service, we must point out that some details. It is not possible to state that a service exactly meets the requirements made by the client as they are defined in a semantic way and misunderstandings may arise. As consequence, we will reply the client with all similar services the OWL-S server could find and match according to our reasoner.
- The information the server looks through is already stored in the server. Indeed, the server will not produce any polling process where servers included in the network are required to send the OWL-S data that defines their services. On the contrary, services that desired to be located by our server must previously register themselves in the system. The reason for no providing an automatic register or polling process is due to security concerns. Before hand, the server will be responsible for the data sent to the client; so, some human verification must be done in advance to assure the veracity of the information sent.

It is quite usual to cipher the information by using XMLSignature[9] and many other techniques for hiding or signing the information by mean of Web Service Enhancements for Security[8]. We will not deal with this aspect of the problem as we consider information from another point of view and the chance for these features will be taken for granted.

The key component in this server is the logic reasoner. This component is the responsible for matching the incoming request and the already available data on the server. If, after contrasting the request made by the client and the information about the service, it assumes that the service may be equal, or, at least, close the information about the service will be included in the response sent to the client.

Other main characteristic for the OWL-S server is use of a extension for the OWL-S semantic possibilities to express common features of services in eGovernment environments. The OWL-S Ontology by itself provides support to express several concerns related to the operations: Inputs, Outputs, Preconditions and Effects for operations (IOPEs). Nevertheless, there is a lack of support for certain operation related to eGovernment operations. As consequence, we propose the extension of the provided capacities by mean of a new ontology. This new ontology, expressed by mean of the OWL[2] language, is made of classes and properties that allow us to express properly important features. Main features for these enhancements include:

- Definition of a set of `ObjectProperty’`s to express features linked to services:
  - Support for requesting security services: encryption and electronic signature.
  - Making explicit the maximum data spread
  - Requesting signed acknowledge of submitted questions.
- Definition of a set of `Class` to characterize actors involved in operations:
  - Characterization of citizens.
  - Characterization of PAs.

So, by using these features, agents can convey on the description of services that really suit their needs or constraints.

## 6 Dealing with UDDI Servers

This solution presented in the previous point is the final goal of the project. We expect that in close future, eGovernment solutions will be developed bearing in mind the presented ideas. The problem we are facing now is legacy systems. There are already UDDI servers working that should be available also under the presented conditions for all agents included in the network.

The idea to make this possible is to encapsulate the full UDDI server under the same interface already presented in the previous point. We will not translate UDDI data into OWL-S as there is a lot of missing information that can not be gathered from the UDDI data format. On the contrary, we will include the UDDI information, just as it is currently stored, into the pool of possible services during the searching phase. Thus, the logical reasoner may search for suitable services in services expressed in OWL-S format but also services expressed under UDDI terms.

From the previous analysis on UDDI data we can conclude that UDDI is not rich enough to express the same data as OWL-S format; mainly only data for Service Model and partially for Service Profile layer is available. In this context we must assume the more restrictive criterion and just include the referred service in the response in case of open restrictions for searches.

Once the information to be sent to the user from the UDDI pool is located, we must include it into the information to be submitted to the client in OWL-S format as the client does not have to be aware of how the service is referred to in our network. To do this we must create an OWL register from the available UDDI data, a kind of wrapper that gives the same appearance but with the obvious gaps of information. We can only compose in some way data for the layers Service Profile and Service Model, so information for layer Service Grounding is send with default values. The goal is not to make it completely transparent for the client but to allow agents in the system to handle the information as any other piece of native OWL-S data.

## 7 Conclusions

eGovernment projects are experiencing a huge impulse towards more mature projects. Not just final solutions or ad-hoc projects, but also full network projects are becoming a reality due to the large amount of efforts devoted to this area of researching. As more complex requirements get and more elements are involved in the project, more concerns must be beard in mind. The current tendency for this kind of projects may led us towards complex environments whit little human intervention and so, a large machine capacity to act on the behalf of human clients. In the light of these facts, we can conclude that knowledge data management and ontology-driven frameworks will play a main role.

After a review of the elements involved in the actual-fashion networks, we find a special point of study where some sources must be devoted to overcome the current situation: the access to services with a semantic support. In this paper, we propose an improvement for service location servers on data networks oriented towards eGovernment services. The main upgrades in this kind of server includes support for security enhancements, semantic support for services and advanced features to support advanced features in a broker-fashion way. The outcomes for this design will lead us into a new dimension for supporting service in, mainly but no only, eGovernment environments. Nevertheless, there are some limitations related to the model proposed and some improvements must be done to increase the accuracy of the semantic matcher by refining the algorithm for matching requests.

If we scan the future horizon for eGovernment projects, we may see services provided through facilities available with semantic support. This will lead operation to upper level of mechanization and integration. Outcomes from this advance will not only include technical field but also, and more important, social and cultural field. The community will be able to achieve a higher level of democracy, understood as civil participation on governmental issues, as they will be able to access to services in a deeper manner: more facilities to access public services (public grants, public job offers, etc), to provide information to and from the administration, more transparency on public management, etc.

The final goal for this trend of development will be achieved when user may develop their own clients to operate in a global semantic knowledge network. To achieve this ambitious goal there is a long trip but the first step must be given and the one presented in this paper is a cornerstone to make this possible.

## 8 REFERENCES

[1] OASIS. Web available at http://www.oasis-open/home/index.php

[2] OWL. Ontology Web Language (OWL). W3C. Web available at http://www.w3.org/2004/OWL/

[3] OWL-S. Ontology Web Language-Service (OWL-S). Web available at http://www.daml.org/services/owl-s/1.1/

[4] UDDI. Universal Description, Discovery and Integration. OASIS. Web available at http://www.uddi.org/.

[5] UDDI, SOAP, and WSDL: The Web Services Specification Reference Book. Prentice Hall Professional Technical Reference. 2002. ISBN:0130857262.

[6] SOAP. SOAP version 1.2. W3C. Web available at http://www.w3.org/TR/soap12-part1/

[7] WSDL. Web Service Description Language. W3C. Web available at http://www.w3.org/TR/wsdl/

[8] WS-Security. The WS-Security Specification. Web available at http://www.ibm.com/developerworks/library/ws-secure/

[9] XML. eXtensible Markup Language.W3C. Web available at http://www.w3.org/XML/

[10] XML Signature. W3C. Web available at http://www.w3.org/Signature/