A semantic approach for discovering eGovernment services

EFTHIMIOS TAMBOURIS, NIKOLAOS LOUTAS, VASSILIOS PERISTERAS AND KONSTANTINOS TARABANIS
Information Systems Laboratory
University of Macedonia
Egnatia 156, Thessaloniki, 54006
GREECE

Abstract: Although eGovernment has been a vivid research area for over a decade, the efficient discovery of the public services that address specific PA clients' needs still remains a challenge. In this paper we propose a semantic approach for addressing this problem, thus making easier the day-to-day interaction between public administration and the PA clients. The proposed approach is to be implemented by means of an eGovernment portal. The portal's components and its architecture are presented and explained in this work. Using the portal, PA clients will be guided through an online structured discussion (i.e. a set of questions and answers) with public administration. As a result of this discussion the portal provides PA clients with all necessary information on the appropriate service(s) that cover their needs. Furthermore, the appropriate service(s) instance(s) which can be invoked though the portal are electronically available.

Key-Words: portal, Semantic Web, OWL, ontology, architecture, eGovernment, Semantic Web Services

1 Introduction

The process of discovering the public services that can fulfill a citizen's need can be a tedious and cumbersome task. As presented in [1-3], the citizen starts by having a need, but (s)he usually does not know which Public Administration (PA) services are currently available to address this need.

Usually, all PA services are somehow related to specific needs since the former have been conceived and organized to address the latter. But this need-to-service link is neither straightforward nor direct. As the organization of administrative space has been mainly functionally centered on broad fields of operation (e.g. health, education, development), a service as provided by a PA agency may cover several needs and a need may be covered by several services. Usually no PA unit can provide by itself the holistic service view needed by the PA clients. To facilitate the communication between the two actors, there is a need to map services to needs and vice-versa.

Sometimes, the mapping between a need and a service is clear. For example, if someone needs to drive a car, a driver's license needs to be obtained. In most administrative systems there is usually one well-known public service type that produces a driver's license as its output, so the mapping can be relatively easy. But even in this case, identifying the specific *service instance* (or *version*) that has to be executed is not a trivial task. For example we have different instances of the driver's license service type for people over 60, for disabled people etc.

On the other hand, there are even more complex cases, where the needs-to-services mapping is not straightforward. For example, consider the case of a blind, married man with children, who would like to know for which social benefits he is eligible. Currently, administration faces serious problems trying to address complex types of needs-to-services matching, like this one. Inside the administrative system there may exist numerous services provided by different agencies that match the need presented. But how could the PA client find all the services available for addressing his/her specific need?

In this work we propose an approach that will help the PA clients express their needs and discover the PA services that address these needs. We facilitate and simplify the discovery process by providing a user-friendly, self-explanatory interface that supports the PA clients and offers them the necessary guidance throughout the process. The discovery process is facilitated by a structured discussion (a set of questions and answers) between the PA client and the portal. This discussion is supported by a set of ontologies and parsing algorithms that using the appropriate reasoning mechanisms can infer the appropriate public service instance that fulfils a PA client's needs based on his/her answers.

The rest of the paper is organized as follows: In section 2 we present some related work in the field

while in section 3 we briefly present a relevant PA service model and a semantic technology used in our work. In Section 4 a detailed description of the portal's architecture can be found. Finally, the conclusion and future work are given in section 5.

2 Related work

The eGov-Bus project (www.egov-bus.org) aims at developing an eGovernment platform that will implement a software environment providing user-friendly, advanced interfaces supporting life events of citizens or enterprises. This work is based on workflow processes. Moreover, language technologies will be used for supporting the full text categorization facility and providing the speech recognition/generation functions.

The OneStopGov project (www.onestopgov-project.org) is currently developing a life-event portal that supports the active, citizen-centric approach. In the OneStopGov portal life-events are implemented using generic workflow Web technologies [5].

Rolland et. al propose in their work in the modeling area an approach that couples modeling and scenario authoring. In fact in this work scenarios are used to discover goals [6].

In this paper we extend our earlier work [4], where we propose a way for supplying the citizen with the correct information from the PA.

3. Background

In this section we present briefly the Governance Enterprise Architecture (GEA) and the Web Ontology Language (OWL).

The GEA models, and more specifically the GEA PA Service model, provide us with the necessary theoretical background in order to model the public services provided by the portal.

OWL provides us the means for using semantics in our approach. In fact OWL will be used in order to encode the ontologies that are one of the basic technologies used in the portal.

3.1 The Governance Enterprise Architecture

The Governance Enterprise Architecture (GEA) proposes a generic domain model for PA. This model defines common aspects and generic features of the domain, with emphasis on service and process models.

In our work we have used the following GEA concepts:

A *Client* that is a citizen, a business or another PA, requests a service from a *Service Provider*. Every service requires a set of *Evidences* (information) for its execution. *Evidences* are used to check the *Preconditions* of the service and are contained in *Evidence Placeholders*. The most typical type of *Evidence Placeholders* in the PA domain is administrative documents. After its execution, every service produces an *Outcome*, which consists of *Output* that is the acquisition of information by the client of the service after the execution of the service, *Effect* that is a change in the state of the world that is caused by the execution of a service and *Consequence* that is information related to the executed service that is of interest to a third party. A detailed presentation of GEA can be found in [1-3].

3.2 The Web Ontology Language (OWL)

OWL is an ontology language for the Semantic Web, developed by the World Wide Web Consortium (W3C) Web Ontology Working Group [7-8]. In OWL, an ontology is a set of definitions of classes and properties. OWL has the ability of applying constraints on the way those classes and properties can be employed.

There are three sublanguages of OWL; OWL Lite, OWL DL and OWL Full.

OWL Lite supports those users primarily needing a classification hierarchy and simple constraints.

OWL DL (Description Logic) supports those users who want the maximum expressiveness while retaining computational completeness[7].

OWL Full extends OWL DL by adding the syntactic freedom of RDF with no computational guarantees. Currently there are no reasoners available for OWL Full.

4. The Portal's Architecture

The portal proposed in our work can play the role of a central point from where public services can be available. Therefore, the portal has to be set up and maintained by a central PA agency, i.e. a ministry. Through the portal the PA clients can have access to all the services provided by PA and to all the information necessary for these services. As a result the portal will be able to answer to questions like the following:

- Which service should I use if I want to drive a car?

- Am I entitled to an unemployment benefit if I am unemployed for 5 months?
- Which administrative documents are necessary for setting up a new business?

The added value of our work relies on the way these questions are answered. This is done via a structured discussion between the PA client and the portal. This discussion is empowered by the use of the tree ontologies described later in detail. These ontologies contain the questions that are used in this structured discussion. The questions stem from the preconditions (as defined in GEA) that have to be fulfilled in order for a specific service to be executed. For example, if one precondition is "The applicant has to be an adult", then the respective question posed to the citizen would be "What is your date of birth?" and from the citizen's answer his/her age can be inferred. After collecting the citizen's answers to a set of such questions the portal discovers the specific service instance that is suitable for the citizen (section 4.3).

In our approach the PA clients are given the opportunity to have a registered profile in the portal. In this case the information that is needed in order to answer the portal's questions will come either from the PA client's profile or by directly asking questions to the PA client.

Many PA services often need to have access to private personal data. Therefore the management of the PA clients' profiles, as well as the way the data that are provided by the PA clients at runtime, need to be treated very carefully. Thus, all the security and privacy regulations that are proposed in the relevant legislation have to be taken into account.

In the following sections we will start by presenting a general overview of the portal's components. We will continue by providing a more detailed presentation of the different types of ontologies that are used by the portal. Ontologies play a key role in our approach and it is them that give the approach its semantic functionalities. Finally, we will conclude with a detailed presentation of the algorithm that implements the structured discussion between the PA client and the portal.

4.1 The Portal's Conceptual Architecture

Here we present the components which comprise the portal's conceptual architecture. The business case of the needs-to-services task was described in the introduction. The portal should be able to help the PA client find all the public services that address the need (s)he expresses taking into account his/her profile.

With the term user profile we refer to a set of characteristics of the individual, such as age, marital status, place of residence etc., that can be either stored in the portal's database (this means that the PA client has registered in the portal) or can be provided by the PA client to the portal at runtime during their interaction [4].

The following actors have been identified in the portal:

- The *PA client*, which corresponds to the GEA Physical Entity concept, can be either a citizen or a business. For the time being our work is focused on citizens. Therefore the term PA client and citizen have equivalent meaning throughout this work. In our case citizens use the portal in order to express their needs and find the services that fulfill these needs.
- The *PA employee* is a civil servant that works for a central PA unit, such as a ministry. The PA employee adds (and updates) the content of the portal.

The portals basic components are shown in Figure 1. We will explain the functionality of each component and we will present how these components communicate with each other.

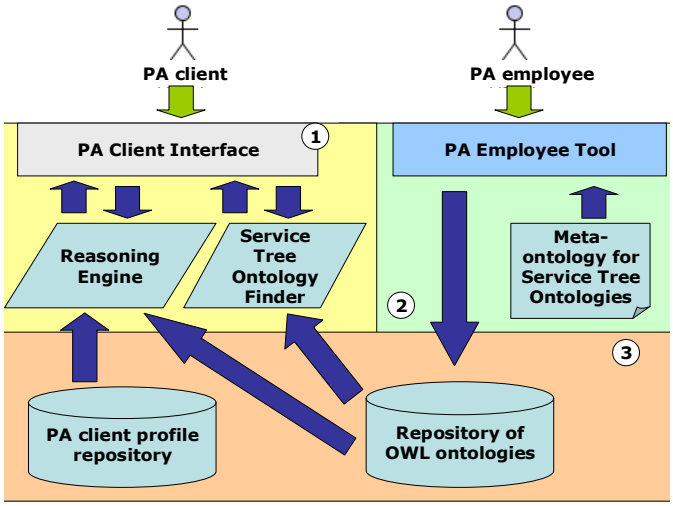

Figure 1. The portal's conceptual architecture

The portal is divided into three basic sub-systems, these are the PA client (1) and the PA employee (2) sub-systems and a common repository infrastructure (3).

The PA client sub-system of the portal consists of the following components:

The PA Client Interface (PACI). The PACI provides the means that citizens can use in order to interact with the portal. The main functionality of the PACI is to present to the citizen the questions asked by the *reasoning engine* (described below)

and collect his/her answers. The answers are then sent to the reasoning engine. Moreover, the PACI should be able to transform the questions sent to it by the reasoning engine from simple strings into html pages, so that they can be presented to the PA clients. Finally, after the results of the discovery process have been returned, they are presented to the PA clients via the PACI.

The Service Tree Ontology Finder (STOF). The STOF presents to the PA client all the available public services and assists him/her to choose the correct Service Tree Ontology. The PA client selects some keywords from a controlled vocabulary in order to express his/her need. The portal reads these keywords and sends them to the STOF. Then, using them the STOF queries the Repository of OWL ontologies and finds the Service Tree Ontologies that match the PA client's need. Finally, the PA client is asked to select the desired one.

The Reasoning Engine. The RE is the core component of the portal as it carries out the traversal of the Service Tree Ontology.

The RE traverses the Service Tree Ontology and each time the next node has to be decided until a leaf node is reached.

In the case of the internal node the RE should verify the node's preconditions. Therefore, the RE first checks if the necessary information can be found in the PA client profile. If not it takes the appropriate question from the Service Tree Ontology and forwards it to the PACI so that the question can be posed to the PA client.

As described above the execution of the RE is terminated successfully when it reaches a leaf and the outcome is a specific service instance. If the citizen is not eligible for none of the services of this Service Tree Ontology, the RE terminates its execution and returns a failure message.

A detailed description of the tree ontology parsing algorithm that is executed in the RE is presented in section 4.3.

Apart from its PA client sub-system the portal has a PA employee sub-system as well, which consists of the following components:

The PA Employee Tool. This tool could provide the PA employee with various functionalities like: registering a new Service Tree Ontology and/or a new PA Client Ontology in the portal, editing or deleting an existing ontology etc.

The meta-ontology for Service Tree Ontologies. This meta-ontology provides a template that can be used in order to guide the PA employee whenever (s)he registers a new Service Tree Ontology in the portal.

Finally, the common repository infrastructure contains:

The Repository of OWL ontologies. This is a repository where all the Service Tree Ontologies and the PA Client Ontology will be stored.

The PA client profile repository. This is a repository where the profiles of all citizens who have registered in the portal are stored. The PA client profile repository will be implemented using a relational database.

4.2 The portal's Ontologies

In this section we will present the different ontologies that will be developed and used by the Portal. These are:

- The meta-ontology for Service Tree Ontologies, which provides the building blocks for creating Service Tree Ontologies for specific PA services;
- The Service Tree Ontologies each of which describes the various subtypes of a PA service;
- The PA Client ontology, which models the characteristics of PA clients. In our case this ontology models the characteristics of citizens.

All the aforementioned ontologies will be developed using OWL.

4.2.1 The Meta-ontology for Service Tree Ontologies

After studying extensively the relevant legislation and the provision of public services, we found out that public services have a tree-like structure. Thus, in the root node there is the generic concept of a service and in the following nodes exist different specializations of the generic service depending on specific characteristics of the citizens who are entitled to each one.

In order to validate our statement, we have used as an example the naturalization PA service in Greece, which is provided by the 13 Greek Regions. In this case the service differentiates based on attributes like the applicant's citizenship, age, place of birth etc. For example, if the applicant has the citizenship of another EU member state or if (s)he has a Greek parent the service requires different inputs and has different execution paths.

Thus, starting from an abstract service type, which is the root of the tree ontology, the subtypes become more and more specific (internal nodes) as we go down the tree. Finally, when a leaf node is reached it means that a specific service instance has

been reached. Summarizing, the aim of this process is to capture the citizen's need and to help him/her track down a specific service instance that address his/her needs starting from an abstract service type and going down a tree-like structure.

The way that the tree ontology is traversed will be presented in detail in the following sections.

The classes of the meta-ontology for service tree ontologies are:

The *Node*. This class represents nodes of the tree. All the nodes of the tree are actually service subtypes with different levels of granularity, starting from abstract and going to more specific ones. We define the following properties:

The *hasDescription* property provides a brief description of the node, as to what the node represents in the tree. The *hasParentNode* property shows the parent node of the current node. The *hasQuestion* property lists the question(s) to be asked to the user.

The *hasCondition* property is used to validate the answer from the question asked in the parent node. This attribute helps decide each time which will be the next node that will be visited. In fact each *hasCondition* corresponds to the preconditions of the current service subtype.

All the nodes of the tree ontologies, either internal nodes or leaves, are subclasses of the Node class and inherit all its properties.

The *InternalNode*. This class represents the internal nodes of the tree. Apart from the attributes that they inherit from *Node*, InternalNodes have also the *hasChildNode* property which shows the child nodes of the current node. There can be more then one child nodes.

The *LeafNode*. This class represents the leaf nodes of the tree ontology and thus the list of specific service subtypes that exist in this ontology. Apart from the properties that they inherit from *Node*, *LeafNodes* have also the *hasOutcome* property which defines the Outcome, as it is defined by GEA, of each service instance.

The *Question*. This class models a question that will be asked to the PA client and it is given as value to the *hasQuestion* property of the *Node*.

The Query. This class models a SPARQL query which expresses the preconditions of a service and it is given as value to the *hasCondition* property of the *Node*.

4.2.2 The Service Tree Ontologies

These ontologies depict the various subtypes of a specific public service.

The development of these ontologies is based on the concepts defined by the meta-ontology for tree ontologies. Thus, all the concepts here are individuals of *InternalNode* or *LeafNode* or the *Question* or the *Query* classes.

4.2.3 The PA Client Ontology

The PA Client Ontology provides a set of classes for describing the information related to the profile of a citizen. The core class in this ontology is the PA client class. Other classes of this ontology are: age, sex, marital status, nationality, number of children, place of residence etc.

4.3 The Service Tree Ontology parsing algorithm

In this section we present the algorithm that parses the Service Tree Ontology and returns the specific service instance. This algorithm provides the core functionality of the portal, since it implements the structured discussion between the PA client and PA.

The algorithm's inputs are: a Service Tree Ontology and the citizen's answers to the posed questions or information from the citizen's profile.

The algorithm's output is the service instance that matches the citizen's need.

The process is described below.

Starting from the root of the selected tree ontology, the portal guides the user so that the service subtypes are stepwise refined. The reasoning engine loads the tree and starts traversing it.

If the current node is an internal node the RE checks the preconditions of the node and tries to find the appropriate information in order to validate them. The RE can acquire the information needed to check the preconditions either from the information that the portal stores about the citizens (profiles) or by directly asking the citizen. The latter is used in the case that the information are either not available in the citizen's profile or the citizen does not have a profile.

In case the citizen has to be asked, the RE reads the question from the node and sends it to the PACI.

The citizen answers the question. The PACI forwards the answer to the RE and the PA Client Ontology instance is populated.

Based on the values of the PA Client Ontology instance the RE evaluates the preconditions of the node's children in order to decide which child will be the next node to be visited.

If the preconditions cannot be evaluated (returns false) the process is terminated and a failure message is returned to the citizen.

If the next node is an internal node the process described earlier is repeated.

If the next node is a leaf the algorithm has reached a complete, successful termination and the corresponding service instance is returned to the citizen.

5 Conclusion - Future Work

An approach that enables the semantic service discovery has been presented. The semantic portal is used to solve the mapping of needs to services problem. We have presented the ontologies used and the system architecture. The ontologies are expressed by means of OWL. The meta-ontology for Service Tree Ontologies can be used in several cases in the PA domain, thus ensuring the application reusability. The algorithm that parses the ontology and discovers the specific service instance has also been presented.

In our next steps we plan to implement a prototype for the portal described in this work using semantic web technologies. An example of the Service Tree Ontologies will be given based on the naturalization PA service in Greece. In a second stage another application will be able to execute the public services found, but this is for the moment out of the scope of this work. Finally, we plan to test the portal using different Service Tree Ontologies for different PA services and have it evaluated by its actual users, namely PA clients and PA employees.

Acknowledgments:
The work reported in this paper is supported by Pythagoras II-Funding of research groups in the University of Macedonia, Priority Action 2.2.3.e, Action 2.2.3, Measure 2.2, to be implemented within the framework of the Operational Programme "Education and Initial Vocational Training II (EPEAEK)" and co-financed by the European Union [3rd Community Support Framework, 75% financed by the European Social Fund 25% National Resources].

References:

[1] V. Peristeras, The Governance Enterprise Architecture - GEA - for Reengineering Public Administration, PhD Thesis in Business Administration, University of Macedonia: Thessaloniki, 2006.

[2] Deloitte, Citizen Advantage: Enhancing Economic Competitiveness Through E-Government, [online] http://www.deloitte.com, 2003.

[3] V. Peristeras, K. Tarabanis, Advancing the Government Enterprise Architecture - GEA: The Service Execution Object Model, *Lecture Notes in Computer Science*, Vol. 3183, 2004, pp. 476-482.

[4] S. K. Goudos, V. Peristeras, K. Tarabanis, Mapping Citizen Profiles to Public Administration Services Using Ontology Implementations of the Governance Enterprise Architecture (GEA) models, *Proceedings of 3rd Annual European Semantic Web Conference*, 2006, pp. 25-37.

[5] E. Tambouris, K. Tarabanis, W. Izdebski and M. Momotko, Life-events revisited: Conceptualization and Representation using Generic Workflows, Workshop *Proceedings of the 5th EGOV conference*, Universitätsverlag Rudolf Trauner, Austria, 2006.

[6] C. Rolland, C. Souveyet, , C. B. Achour, Guiding Modeling Using Scenarios. *IEEE Trans. Softw. Eng*. 24, 12, 1998.

[7] M. Dean et al., OWL web ontology language reference, W3C Recommendation, [online] http://www.w3.org/TR/owl-ref/ , 2004.

[8] D. L. McGuinness, F. Harmelen, OWL Web Ontology Language Overview, W3C Recommendation, [online] http://www.w3.org/TR/owl-features/ , 2004.