

INTERNATIONAL JOURNAL ON SEMANTIC WEB AND INFORMATION SYSTEMS

April-June 2012, Vol. 8, No. 2

Table of Contents

REVIEW ARTICLES

1 Towards Practical ABox Abduction in Large Description Logic Ontologies

*Jianfeng Du, Guangdong University of Foreign Studies & State Key Laboratory of Computer Science,
Institute of Software, Chinese Academy of Sciences, China*

*Guilin Qi, Southeast University & Key Laboratory of Symbolic Computation and Knowledge Engineering
of Ministry of Education, Jilin University, China*

*Yi-Dong Shen, State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of
Sciences, China*

Jeff Z. Pan, University of Aberdeen, UK

34 A New Model to Compute the Information Content of Concepts from Taxonomic Knowledge

David Sánchez, Universitat Rovira i Virgili, Spain

Montserrat Batet, Universitat Rovira i Virgili, Spain

51 Adding Semantic Annotations into (Geospatial) RESTful Services

Victor Saquicela, Ontology Engineering Group, Universidad Politécnica de Madrid, Spain

Luis M. Vilches-Blázquez, Ontology Engineering Group, Universidad Politécnica de Madrid, Spain

Oscar Corcho, Ontology Engineering Group, Universidad Politécnica de Madrid, Spain

Adding Semantic Annotations into (Geospatial) RESTful Services

Victor Saquicela, Ontology Engineering Group, Universidad Politécnica de Madrid, Spain

Luis M. Vilches-Blázquez, Ontology Engineering Group, Universidad Politécnica de Madrid, Spain

Oscar Corcho, Ontology Engineering Group, Universidad Politécnica de Madrid, Spain

ABSTRACT

In this paper the authors present an approach for the semantic annotation of RESTful services in the geospatial domain. Their approach automates some stages of the annotation process, by using a combination of resources and services: a cross-domain knowledge base like DBpedia, two domain ontologies like GeoNames and the WGS84 vocabulary, and suggestion and synonym services. The authors' approach has been successfully evaluated with a set of geospatial RESTful services obtained from ProgrammableWeb.com, where geospatial services account for a third of the total amount of services available in this registry.

Keywords: DBpedia, GeoNames, Geospatial RESTful Service, Semantic Annotation, World Geodetic System (WGS84)

INTRODUCTION

In recent years, since the advent of Web 2.0 applications and given some of the limitations of “classical” Web services based on SOAP and WSDL, Representational State Transfer (REST) services have increased their presence on the Web (Pautasso, Zimmermann, & Leymann, 2008). Machine-oriented Web applications and APIs that are conformant to the REST architectural style (Fielding, 2000), normally referred to as RESTful Web services, have started ap-

pearing, mainly due to their relative simplicity and their natural suitability for the Web.

However, using RESTful services still requires much human intervention since the majority of their descriptions are given in the form of unstructured text in Web pages (HTML), which contain lists of available operations, their URIs and parameters (also called attributes), expected output, error messages, and some examples of their execution. This hampers the automatic discovery, interpretation and invocation of these services, what may be helpful to improve the efficiency in the development of applications that are based on their use.

DOI: 10.4018/jswis.2012040103

Traditionally, semantic annotation approaches for services (WSMO, OWL-S, SAWSDL, etc.) have focused on defining service description formalisms, and have been normally applied to WS-* services and their corresponding middleware. More recently, these (usually heavyweight) formalisms have started to be adapted into more lightweight approaches for the semantic description of RESTful services (Maleshkova, Kopecký, & Pedrinaci, 2009; Kopecký, Gomadam, & Vitvar 2008; Lathem, Gomadam, & Sheth, 2007), as described later in this paper. One of the reasons for making these annotations more lightweight is to increase their uptake, promoting the creation of such annotations by service developers and publishers, while still providing some added value to the syntactic descriptions currently available for both WS-* and REST approaches.

Nevertheless, most of the processes related to the annotation of such services still require a large amount of human intervention. In this paper we will mainly focus on RESTful service annotation, for which some approaches already exist (e.g., (Maleshkova, Pedrinaci, & Domingue, 2009; Alowisheq, Millard, & Tiropanis, 2009). In these approaches, humans firstly need to understand the informal descriptions provided in the RESTful service description pages, and then the semantic annotation of these services is done mostly manually, with or without tool assistance.

In this paper, we describe our approach to address the challenge of automating the semantic annotation of RESTful services by: (1) obtaining and formalising their syntactic descriptions, what allows their registration and invocation, and (2) interpreting and semantically enriching their parameters, what allows generating semantically-enriched service descriptions in any of the available formalisms for RESTful semantic service description.

The main contribution of our work is the partial automation of the process of RESTful semantic annotation services, what improves the current state of the art in this area. We have defined a process to perform this type of annotation, and we propose the usage of diverse

types of external resources and services to help in this process: a cross-domain knowledge-base like DBpedia, two domain ontologies like GeoNames and the WGS84 vocabulary (since we are focused on RESTful services on the geospatial domain), and suggestion and synonym services.

The remainder of this paper is structured as follows: First, we present the background and related work in the context of semantic annotation of WS-* and RESTful services. Then, we describe our RESTful service annotation approach, explaining how we structure the process into syntactic and semantic annotation, and how we derive such syntactic and semantic descriptions making use of external resources and services. Next, we present the evaluation of our system in the context of geospatial services. Finally, we present some conclusions and identify future lines of work.

BACKGROUND AND RELATED WORK

In this section, we provide a brief introduction to WS-* and RESTful services, and describe existing approaches related to the semantic annotation of such types of services.

WS-* and REST Services

According to Haas and Brown (2004), “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”.

The World Wide Web Consortium (W3C) identifies two classes of Web services. On the one hand, the “Big” Web services technology stack (SOAP, WSDL, WS-Addressing, etc.), where the service may expose an arbitrary set of operations and specialised middleware is needed in order to handle service publication and message handling. On the other hand, RESTful Web

services, where REST stands for Representation State Transfer and was originally introduced by Fielding (2000). REST emphasizes the usage of Web architecture principles, among which the most important are:

- **Identification of resources.** Individual resources are identified in requests (e.g., by means of URIs) and are conceptually separate from the representations returned to service consumers (which could be sent in HTML, XML, JSON, etc.).
- **Manipulation of resources through these representations.** When a service consumer obtains a representation of a resource, including any metadata attached to it, it has enough information to modify or delete the resource on the server, provided it has permission to do so.
- **Self-descriptive messages.** Each message exchanged between service consumers and providers includes enough information to describe how to process the message. For example, the parser to be invoked by the service consumer may be specified by an Internet media type (previously known as a MIME type). Responses also explicitly indicate their cacheability.
- **Hypermedia as the engine of application state.** Clients make state transitions only through actions that are dynamically identified within hypermedia by the server (e.g., by hyperlinks within hypertext). Except for simple fixed entry points to the application, a client does not assume that any particular actions will be available for any particular resources beyond those described in representations previously received from the server.

The following methods from the HTTP protocol are defined for RESTful interactions:

- HTTP GET is used by service consumers to obtain a representation of a resource from a URI.
- HTTP DELETE is used to remove representations of a resource.
- HTTP POST is used to update or create the representations of a resource.
- HTTP PUT is used to create representations of a resource.

Formalisms and Tools for the Semantic Annotation of Web Services

As discussed in the introduction, much work has been done in the state of the art on semantically-enabled Web Services. In this section we will describe their main characteristics and limitations.

Most of this work focused initially on WS-* services, as they were for a long time the most popular among service developers. OWL-S (Martin et al., 2004) and WSMO (<http://www.wsmo.org/>) are probably the most prominent approaches for semantic annotation. They proposed their own ontologies to describe services, their operations, input and output parameters, etc. In the case of WSMO a new description language, WSML, was also developed. Tools were made available to assist service providers in the semantic annotation process (e.g., WSMO editor, OWL-S editor, etc.), facilitating the generation of these descriptions, and middleware was also made available for the discovery, composition, mediation and invocation of such types of services.

Given the low adoption of these frameworks outside the research community, mainly because of their complexity and the use of non-standard representation languages, SAWSDL (Kopecký, Vitvar, Bournez, & Farrell, 2007; Martin, Paolucci, & Wagner, 2007) was proposed later in order to allow including such descriptions inside WSDL documents (the standard way of describing syntactically WS-* services) through the *modelreference* annotation.

According to Paolucci, Wagner, and Martin (2007), these approaches (OWL-S, WSMO, and SAWSDL) provide semantic annotation attributes for WSDL, which are meant to be used in similar ways. OWL-S and SAWSDL express the semantics of inputs and outputs

of WSDL operations. Furthermore, SAWSDL does it via a direct annotation of the types and elements, while the OWL-S Grounding maps the content of inputs and outputs to their semantic representation in the *Process Model*. With respect to WSMO, it proposes high-level objectives and approaches similar to those of OWL-S, but it focuses on goals mediation and choreography. In SAWSDL there is no explicit mention of precondition and effects that one can find in WSMO and OWL-S.

In the context of RESTful services, several description approaches exist, such as the Web Application Description Language (WADL) (Hadley, 2009), which allows describing syntactically RESTful services and the resources that they access. In terms of semantic descriptions, some relevant approaches are the extensions to WADL described in Battle and Benson (2008) and Freitas and Ferreira (2009).

Other semantic RESTful service description formalisms are MicroWSMO (Maleshkova, Gridinoc, Pedrinaci, & Domingue, 2009), which uses hRESTS (HTML for RESTful services) (Maleshkova, Gridinoc, Pedrinaci, & Domingue, 2009; Kopecký, Gomadam, & Vitvar, 2008); and SA-REST (Maleshkova, Pedrinaci, & Domingue, 2009; Lathem, Gomadam, & Sheth, 2007), which uses SAWSDL (Maleshkova, Kopecký, & Pedrinaci, 2009) and RDFa (Adida, Herman, & Sporny, 2012) to describe service properties. All of these approaches share the characteristic for being more lightweight than WS-* semantic annotation approaches, following the spirit of REST architectures. Some tool support exists for these approaches as well. For instance, we can mention the SWEET tool (<http://sweet.kmi.open.ac.uk/>) (Maleshkova, Pedrinaci, & Domingue, 2009).

Processes for the Semantic Annotation of Web Services

As aforementioned, several tools have been made available for the semantic annotation of services. Many of these tools are editors that help service developers in the manual creation

of such descriptions according to the underlying languages and vocabularies.

There is also work described in the literature that is focused on the processes to be followed for semantic annotation and the provision of support to those processes that goes beyond the development of graphical editors. Without the aim of being exhaustive, we describe here some relevant work that is close to our proposal.

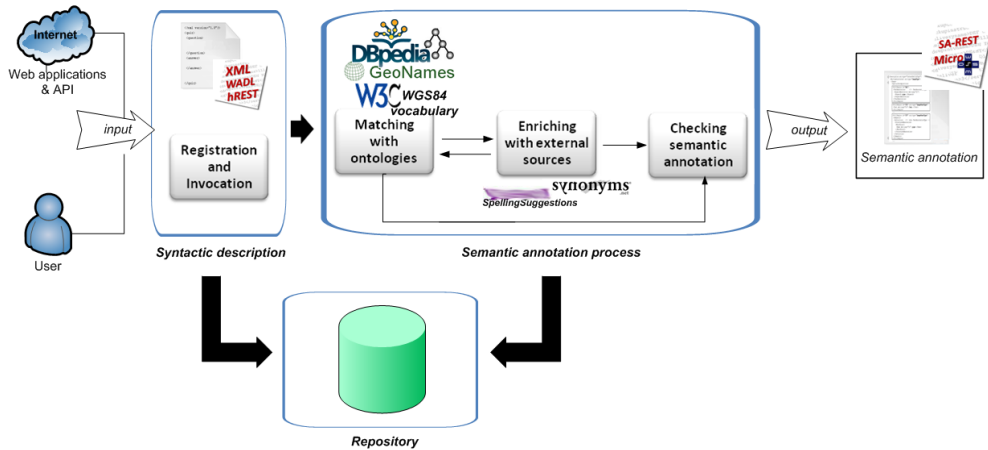
Kino (Ranabahu, Parikh, Panahiazar, Sheth, & Logan-Klumpler, 2011) is a document management system, currently used mainly in Biology realm, which uses SA-REST annotations for services and provides faceted search functionalities. The annotation process proposed for this system is mostly manual, using a browser plug-in, although assisted by the use of synonym services that contribute to improve recall during the annotation process.

Finally, another approach for service description that aims at automating the annotation process, and hence can be considered closer to our work, is presented in Heß and Kushmerick (2003). This approach classifies, using Naïve Bayes, service parameter datatypes using HTML forms and Web services. Our work is similar to this one because it generates annotations for input parameters, invokes Web services using these parameters, and generates output data when an execution is successful.

AN AUTOMATED APPROACH FOR THE SEMANTIC ANNOTATION OF RESTful SERVICES

In this section, we present the core of our approach for the semantic annotation of RESTful services, which is focused on the annotation of functional descriptions (Fensel, Fischer, Kopecky, Krummenacher, Lambert, & Vitvar, 2010) associated with those types of services. We have developed a system that gives support to this approach, and which is summarized graphically in Figure 1. Our system consists of the following three main components: 1) invocation and registration modules, which are in

Figure 1. Our RESTful service semantic annotation approach



charge of the syntactic description of the service; 2) repository, which stores service descriptions; and 3) semantic annotation components, which make use of diverse external resources in order to obtain semantic annotations and increase the recall in finding such semantic annotations. Next, we briefly describe these components, illustrating the descriptions with two sample services in the geospatial domain, which will be used as a running example.

Running Examples: Two RESTful Services in the Geospatial Domain

Nowadays, one of largest online repository of information about Web 2.0 mashups and APIs is ProgrammableWeb.com (Adamczyk, Smith, Johnson, & Hafiz, 2011). As of March 2011, this aggregator site provides information on 5,721 mashups and 3,108 APIs that were registered since September 2005. Mashups tagged as “mapping” represent a 33% of the total number of mashups (2,468 mashups), which shows the importance of geospatial information in the generation of REST-based applications. With respect to APIs, GoogleMaps is the most used (41%, that is, in 2,206 mashups). The following services, taken from the aforementioned site, are two representative geospatial RESTful services:

Service 1.

<http://ws.geonames.org/countryInfo?country=ES>

This service retrieves information related to a ‘country.’ More specifically, it returns information about the following parameters: ‘capital,’ ‘population,’ ‘area’ (km²), and ‘bounding box of mainland’ (excluding offshore islands). In our example, we retrieve information about Spain (ES).

Service 2.

http://api.eventful.com/rest/venues/search?app_key=p4t8BFcLDtCzpxdS&location=Madrid

This service retrieves information about places (venues). More specifically, given a city name it returns parameters like: venue_name, region_name, country_name, latitude, longitude, etc. In our example, we retrieve information about Madrid.

Syntactic Description of RESTful Services

RESTful services are normally described in an unstructured manner in sites like *programmableWeb*. These descriptions contain their URLs,

Table 1. XML response of two sample RESTful services

Service 1	Service 2
1: <geonames> 2: <country> 3: <countryCode>ES</countryCode> 4: <countryName>Spain</countryName> 5: <isoNumeric>724</isoNumeric> 6: <isoAlpha3>ESP</isoAlpha3> 7: <fipsCode>SP</fipsCode> 8: <continent>EU</continent> 9: <capital>Madrid</capital> 10: <areaInSqKm>504782.0</areaInSqKm> 11: <population>40491000</population> 12: <currencyCode>EUR</currencyCode> 13: <languages>es-ES,ca,gl,eu</languages> 14: <geonameId>2510769</geonameId> 15: <bBoxWest>-18.169641494751</bBoxWest> 16: <bBoxNorth>43.791725</bBoxNorth> 17: <bBoxEast>4.3153896</bBoxEast> 18: <bBoxSouth>27.6388</bBoxSouth> 19: </country> 20: </geonames>	1: <venue id="V0-001-000154997-6"> 2: <url>http://eventful.com/madrid/venues/la-ancha-/V0-001-000154997-6</url> 3: <country_name>Spain</country_name> 4: <name>La Ancha</name> 5: <venue_name>La Ancha</venue_name> 6: <description></description> 7: <venue_type>Restaurant</venue_type> 8: <address></address> 9: <city_name>Madrid</city_name> 10: <region_name></region_name> 11: <region_abbr></region_abbr> 12: <postal_code></postal_code> 13: <country_abbr2>ES</country_abbr2> 14: <country_abbr>ESP</country_abbr> 15: <longitude>-3.68333</longitude> 16: <latitude>40.4</latitude> 17: <geocode_type>City Based GeoCodes 18: </geocode_type> 19: <owner>frankg</owner> 20: <timezone></timezone> 21: <created></created> 22: <event_count>0</event_count> 23: <trackback_count>0</trackback_count> 24: <comment_count>0</comment_count> 25: <link_count>0</link_count> 26: <image></image> 27: </venue>

and optionally natural language descriptions, tags, and execution examples.

Our first step starts with the URL of a RESTful service. Such URL may have been discovered by a user by browsing this site, or may have been sent to the user by a friend, or may have been crawled automatically.

Given such URL, our system tries to invoke the RESTful service with some sample parameters, which are normally provided together with such URLs, and analyzes the response to obtain a basic syntactic description of the parameter set, used as inputs and outputs. If the provided URL cannot be invoked successfully (e.g., the service is not available any more), our annotation process cannot continue without further human intervention.

Such service invocation and the processing of results are performed using the Service Data Object (Geyer, 2007) API (Resende, 2007),

which allows abstracting from the formats¹ in which the results are made available (e.g., JSON, XML, etc.), navigating through those results, and extracting the corresponding parameters. The results of a sample invocation of our sample services are showed in Table 1.

These results can already be expressed in many different formalisms and formats: in description languages like WADL or hRESTS, stored into a relational model, in HTML, etc. For instance, Table 2 shows part of the WADL description of *Service 1*, generated using the SoapUI API (<http://www.soapui.org/>).

This WADL description contains additional information about the main characteristics of parameters, such as datatype, required, style, default language, etc. We can also use the SoapUI API to generate HTML descriptions of those services (Figure 2). Such HTML descriptions can be used as a starting point for tools like

Table 2. WADL description

```

1: <?xml version="1.0" encoding="UTF-8"?>
2: <application xmlns="http://wadl.dev.java.net/2009/02">
3: <doc xml:lang="en" title="1"/>
4: <resources base="http://ws.geonames.org">
5: <resource path="countryInfo" id="countryInfo">
6: <doc xml:lang="en" title="countryInfo"/>
7: <param name="country" default="ES" type="xs:string" required="false" style="query" xmlns:xs="http://
www.w3.org/2001/XMLSchema"/>
8: <method name="GET" id="Method 1">
9: <doc xml:lang="en" title="Method 1"/>
10: <request/>
11: <response status="0">
12: <representation mediaType=""/>
13: </response>
14: <response status="200">
15: <representation mediaType="text/xml" element="geonames"/>
16: </response>
17: </method>
18: </resource>
19: </resources>
20: </application>

```

SWEET, which base their annotation process on HTML pages describing services. This shows that our syntactic annotation process already provides some added value that can be used by other approaches and tools.

As shown in Figure 1, our approach considers the use of a repository where these descriptions can be stored. We use the object-oriented model shown in Figure 3 for such repository.

Once the RESTful service is registered in our repository and the WADL and HTML descriptions are generated, if required, we propose to find additional information that can be useful for subsequent phases of the annotation process. For example, allowed values for some of the parameters. This can be done, for instance, by invoking the service without any of its associated parameters. For such a type of invocation, the service may return a fault, which does not provide additional information, but it could be also the case that it provides a list of allowed values, or redirects us to a help page with such values, as defined by the service provider. For example, the following invocation of *Service 1* with no parameters: <http://ws.geonames.org/countryInfo?> provides us with a list of countries

that can be used as values for the countryInfo input parameter.

In this process our system also considers the invocation of service URLs like <http://www.foo.org/weather/Madrid>.

These services belong to a specific RESTful entity and they are always invoked with its associated parameters.

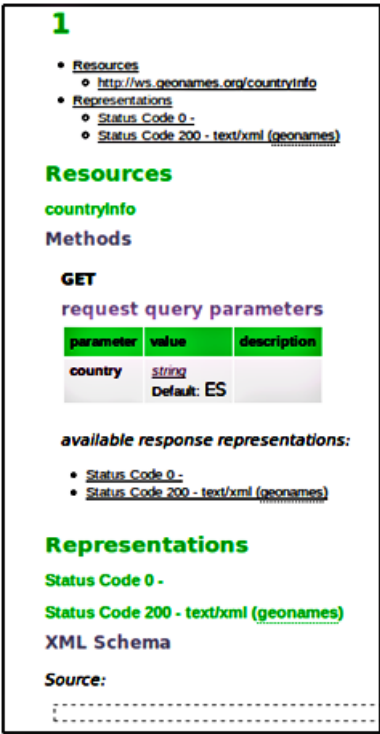
The results obtained from these types of invocations are also stored into the object-oriented model, together with the information that had been previously stored.

Semantic Annotation of RESTful Services

Table 3 shows the output parameters for the two services that we use as examples. While it is easy to observe that there is some similarity between diverse parameters (e.g., countryName and country_name) and that they return similar values (Spain), from a syntactical point of view, these parameters are considered to be different. This is one of the problems that we aim at solving during semantic annotation.

Semantic annotation consists in linking the input and output parameters that we have iden-

Figure 2. HTML description of a RESTful service (Service1')



tified during our syntactic annotation process with ontology-based descriptions that represent the type of information that they require or provide.

The semantic annotation process of RESTful services presents some difficulties, as described by Maleshkova, Kopecký, and Pedrinaci (2009) and Alowisheq, Millard, and Tiropanis (2009). These difficulties are related to lack of machine-processable descriptions and to the lack of documentation that these services normally exhibit, what requires a large degree of human intervention for discovering and using a service.

In our process, the starting point for semantic annotation is the list of syntactic parameters obtained previously. We follow a heuristic approach that combines a number of external services and semantic resources to propose annotations for the parameters, as shown in Figure

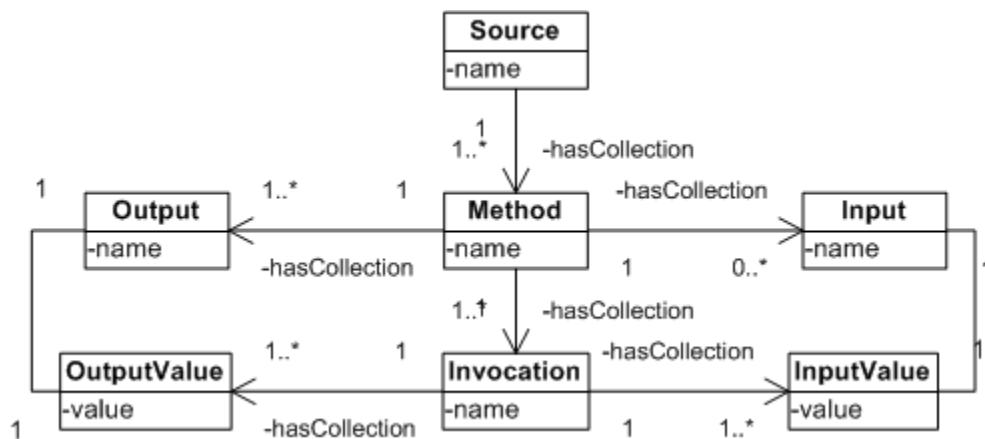
4. Next, we describe the main components of our semantic annotation process.

A Model for Describing RESTful Service Parameters Semantically

In order to describe semantically these services, we define a model to represent the relationships of the different service parameters with the diverse resources used for semantic annotation. This model is based on the service description approaches used by hRESTS, SAWSDL, and microWSMO. Moreover, our model considers values (instances) associated to different parameters as additional information about services, something that is not considered by any of these approaches. These instances are obtained from semantic data sources, such as DBpedia and GeoNames.

Some of the elements of this model are domain-independent, while others are domain

Figure 3. Model for RESTful service description



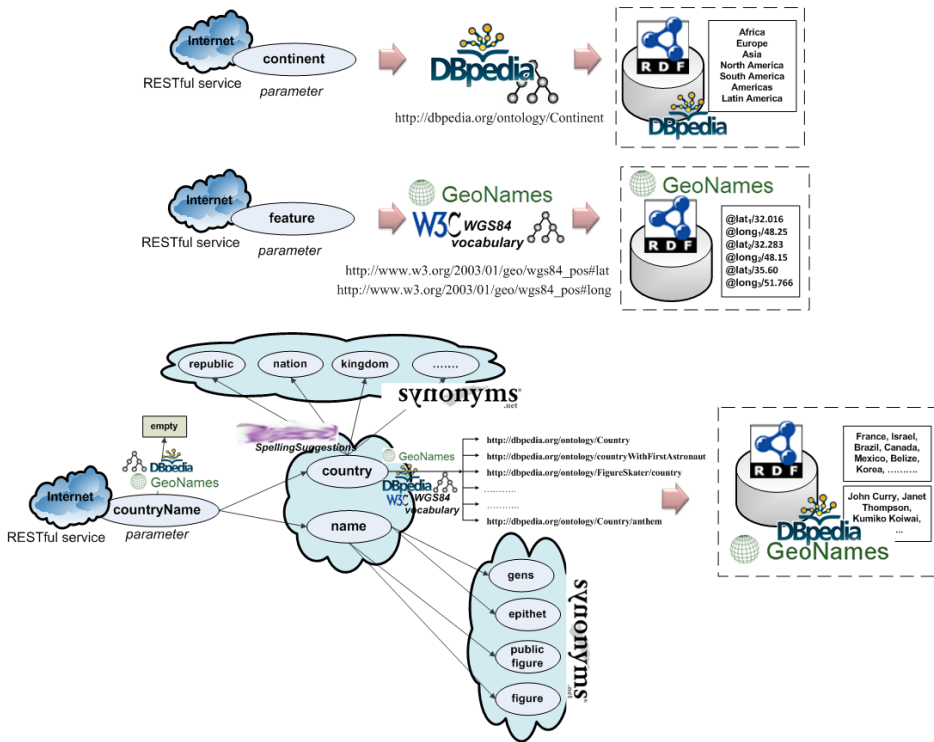
dependent. With respect to the domain-independent component, we use DBpedia as the main source of background knowledge for supporting the semantic annotation process. This is complemented by the domain-dependent component. In the context of the shown examples, we use GeoNames as a source related to geospatial information. This model (Figure 5), where each parameter has a relationship with input or output of a service, defines the following components:

- Parameter.** This class provides a list of all parameters (inputs and outputs) collected from different services. Likewise, we search for additional information for each parameter, such as suggestions and synonyms, so as to enrich the initial description of parameters. The relation *hasCollection* relates *Parameter* with *DBpediaOntology*. Every parameter can be related to any number of DBpedia classes or properties (from 0 to N).
- Ontologies.** This class contains classes and properties of the DBpedia and GeoNames ontologies, and WGS84 vocabulary related to the parameters of each service. This class is related to the classes *DBpediaInstance* and *GeonamesInstance* by means of the relation *hasCollection*. Ontologies can be related to any number of DBpedia or GeoNames instances (from 0 to N).

Table 3. Syntactic description of our two sample RESTful services

<ul style="list-style-type: none"> Service 1: <code>countryInfo(\$country,bBoxSouth,isoNumeric,continent,fipsCode,areaInSqKm,lang uages,isoAlpha3,countryCode,bBoxNorth,population,bBoxWest,currencyCode,bBoxE ast,capital,geonameId,countryName)</code> Service 2:
<pre>rest/venues/search(\$location,\$app_key,id,link_count,page_count, longitude,trackback_count,version,venue_type,owner,url,country_ name,event_count,total_items,city_name,address,name,latitude,pa ge_number,postal_code,country_abbr,first_item,page_items,last_i tem,page_size,country_abbr2,comment_count,geocode_type,search_t ime,venue_name)</pre>

Figure 4. Semantic annotation process



- **DBpediaInstance.** This class collects values from the DBpedia SPARQL Endpoint, where a parameter may have one or more associated resources.
- **GeonamesInstance.** This class collects geospatial information related to latitude, longitude, and bounding box parameters from a GeoNames SPARQL Endpoint that we set up with information (RDF) of this data source.

The information related to each parameter of the RESTful service (semantic annotations) is stored only once in the system repository. By doing this, we avoid the duplication of information related to the same parameters, hence storing annotations independently of services and increasing the efficiency of our system and the reusability of our annotations.

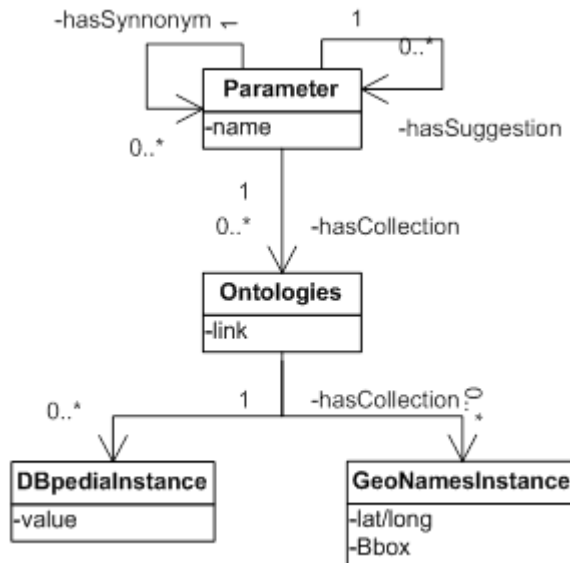
Ontological Resources

In our approach we use three widespread ontologies, such as DBpedia, GeoNames and WGS84 vocabulary.

DBpedia is a community effort to extract structured information from Wikipedia and to make this information available on the Web. The DBpedia ontology (Auer et al., 2005) is a shallow, cross-domain ontology, which has been manually created based on the most commonly used infoboxes within Wikipedia. The ontology currently covers over 259 classes which form a subsumption hierarchy and are described by 1,200 different properties. Moreover, this ontology currently contains about 1,478,000 instances (413,000 instances belong to “Place” class).

GeoNames is a geographical database that contains over 8 million geographical names. The structure behind the data is the GeoNames

Figure 5. Model for the description of geospatial RESTful service parameters



ontology (http://www.geonames.org/ontology/ontology_v2.2.1.rdf), which closely resembles the flat-file structure. A typical individual in the database is an instance of type Feature and has a Feature Class (administrative divisions, populated places, etc.), a Feature Code (subcategories of Feature Class) along with latitude, longitude, etc. associated with it.

The **WGS84 Vocabulary** (http://www.w3.org/2003/01/geo/wgs84_pos) is a basic RDF vocabulary that provides a namespace for representing lat(itude), long(itude) and other information about spatially-located things, using WGS84 as a reference datum.

Using Ontological Sources in the Annotation Process

At this stage, the list of syntactic parameters previously obtained is used to query DBpedia and a locally-installed GeoNames SPARQL Endpoints, and retrieve associated results for each parameter, as follows:

First, the system retrieves all the classes from the DBpedia ontology whose names have a match with each parameter of the RESTful

service. In this matching process we test two different approaches:

- On the one hand, we use an exact match to compare parameters of the RESTful service with the labels of the ontologies' classes and properties.
- On the other hand, we consider different similarity metrics to compare parameters with the labels of the elements of these ontologies. Each metric evaluates to a similarity value in the range [0, 1], with higher values indicating a greater similarity. The used metrics are classified in two groups²: equality and similarity. The former collects metrics that evaluate to a similarity value like 0 or 1. This group is composed of a set of metrics, such as: Block distance, Cosine similarity, Dice's Coefficient, Jaccard Similarity, Matching Coefficient, and Overlap Coefficient. The latter gathers metrics that evaluate to a similarity value between 0 and 1. This group is composed of diverse metrics, such as: Jaro Winkler, Levenshtein Distance, Monge Elkan dis-

Table 4. Results of used similarity metrics

Service Parameters	DBpedia ontology (classes)	Similarity metrics				
		Jaro Winkler	Levenshtein Distance	Monge Elkan distance	Needleman-Wunch distance	Smith-Waterman-Gotoh
country_name	Country	0,94	0,58	1,00	0,58	1,00
countryName	Country	0,95	0,64	1,00	0,64	1,00
city_name	City	0,89	0,44	1,00	0,50	1,00
population	populatedDensity	0,60	0,50	0,80	0,56	0,80
currencyCode	Currency	0,96	0,67	1,00	0,67	1,00

tance, Needleman-Wunch distance, and Smith-Waterman-Gotoh.

These proposals allow matching between strings, which are related to service parameters and ontology components, such as countryName, country_name, or country. An example of this matching process, using metrics of similarity group, is shown in Table 4.

If the system obtains correspondences from the matching process, it uses these DBpedia concepts individually to retrieve samples (concept instances) from the DBpedia SPARQL Endpoint.

Likewise, when a parameter matches an ontology class related to some geospatial information; such as latitude, longitude or bounding box our system retrieves samples from the local GeoNames SPARQL Endpoint. These geospatial parameters are retrieved in a combined manner (for instance, a latitude goes together with a longitude), that is, when the annotation process of a RESTful service requires information about latitude and longitude, our system invokes the GeoNames SPARQL Endpoint using the following query:

```
select ?lat ?lng
where
{?s <http://www.w3.org/2003/01/geo/
wgs84_pos#lat>?lat.
?s <http://www.w3.org/2003/01/geo/
wgs84_pos#long>?lng} limit ? offset ?
```

With this query our system requests a list of latitudes and longitudes. The values of this list are recovered randomly (using limit and offset SPARQL parameters), in order to avoid recovering always the same values for different queries.

The resulting information is suggested automatically to the system and registered as a possible value for the corresponding parameter. When a parameter matches more than one concept in the DBpedia ontology, our system only considers those concepts that have information (instances), and it automatically discards those ontology concepts without instances.

Next, the system tries to find correspondences between parameters of the RESTful service and ontology properties. If the system obtains some correspondences, it uses these DBpedia properties individually to retrieve information of the DBpedia or GeoNames SPARQL Endpoint, as described. Furthermore, this information is registered as a possible correct value for the corresponding parameter.

Finally, with the obtained classes and properties, the system calls the DBpedia and GeoNames SPARQL Endpoints to retrieve values (instances).

External Resources for Enriching the Semantic Annotations

Our system looks for matches with DBpedia (and GeoNames) classes and properties. Hence it is possible to have parameters with no corre-

spondences identified, since there are many lexical and syntactic variations that the parameter names may have, and because in some cases the information that is being requested may not be available in any of the external sources (suggestion and synonym services) that are consulted. In order to annotate semantically the parameters that did not match any DBpedia resource, we use additional external services to enrich the results. We describe the main characteristics of the external services that we consider.

Spelling Suggestion Services

Web search engines (e.g., Google, Yahoo, and Microsoft) usually try to detect and solve users' writing mistakes. Spelling Suggestion services, also called "Did You Mean," are algorithms which aim at solving these spelling mistakes. For example, when a user writes `countryName` these algorithms suggest `country` and `name` separately.

In our system we use the Yahoo Boss service (http://developer.yahoo.com/search/boss/boss_guide/Spelling_Suggest.html) to retrieve suggestions about the parameters that we have obtained in the previous steps and for which we have not obtained any candidate in our semantic resources. Thus, for each parameter that the system did not find a correspondence with classes or properties in DBpedia (nor GeoNames), this service is invoked to obtain a list of suggestions (alternative names for parameters) to query DBpedia (and GeoNames) ontology again. The output is registered and stored into the repository. Following the previous example, the parameter `countryName` is not found in the DBpedia ontology. Nevertheless, the added service allows separating this parameter in `country` and `name`, and then it calls to the DBpedia ontology and its related SPARQL Endpoint with these new strings for obtaining results.

Synonym Services

This external service (<http://www.synonyms.net/>) is incorporated into the system to retrieve possible synonyms for a certain parameter. This

service tries to improve the semantic annotation process when our system does not offer results for the previous steps, that is, when we still have parameters in a RESTful service without any potential annotations.

As an example, we may have a parameter called `address`. The invocation process uses the synonyms service to retrieve a set of synonyms of `address` such as *extension*, *reference*, *mention*, *citation*, *denotation*, *destination*, *source*, *cite*, *acknowledgment*, and so on. These outputs are registered and stored into the repository, and then, the service calls to the DBpedia (and GeoNames) ontology and their related SPARQL Endpoints for obtaining results again.

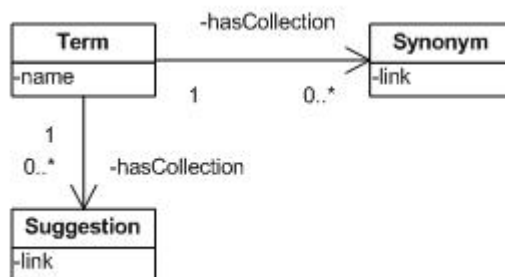
Both spelling suggestion and synonym services use the matching process described to find possible matches between the output of these services and the components of the used ontologies. The results of these matches are stored in a cache. This cache was developed to overcome query limitations of both suggestion and synonym services (approximately 1,000 invocations per day). Figure 6 shows the simple model built for this cache.

Checking the Correctness of the Semantic Annotation of RESTful Services

In order to check the collected sample individuals and the initial semantic annotations obtained as a result of the previous process, our system invokes the registered RESTful service and validates the input and output parameters.

For the validation of the *input parameters*, our system selects, for each parameter, a random subset of the example instances (of classes and/or properties) obtained from the DBpedia (and GeoNames) ontology. Next, it makes several invocations of the RESTful service iterating over these registered values. The system does not check this with all the possible combinations of collected instances for all parameters for two reasons: first, because of the combinatorial explosion that may be produced in such a case, and second because many RESTful services have invocation limitations.

Figure 6. Cache model



When a service has one or more than one input parameter, the system obtains randomly some instances of this parameter for the validation process. Each parameter generates a collection (list) of instances from our repository. Then, the system joins instances to obtain a table of all combinations of each parameter. Likewise, the geospatial parameters, specifically latitude and longitude parameters, are combined to obtain some values (instances) that can be used for this invocation.

If the service returns results from the invocation, then the service is considered as executable, and the corresponding annotations are marked as valid. If a service cannot be invoked successfully, the service is classified as non-executable and is automatically discarded from the list of services that can be automatically annotated.

For the validation of the *output parameters*, our system only takes into account executions with the correct inputs from the input sets that have been considered before. Next, the system compares the outputs obtained after execution with the information already stored in the repository due to the initial retrieval processes done before. If the output can be matched, our system considers that the output annotation is valid.

Finally, the correspondences established between the different parameters of the RESTful service and the DBpedia (and GeoNames) ontology are registered and stored in the repository, so that they can be used later. Besides, these correspondences are used to replace the syntactic description of the RESTful service with a *new*

version that is annotated semantically. In such a way, this annotation allows generating semantic descriptions or annotations of any of the types that were identified in the related work section (WADL, hRESTS, SA-REST, etc.).

Table 5 provides an abbreviated form of this description for our exemplar *Service 1*. Furthermore, Table 6 and Table 7 were generated using HTML description shown in Figure 2. Table 6 and Table 7 provide a semantic annotation piece of the aforementioned service (*Service 1*) according to the hRESTS and SA-REST formats, respectively. The detailed descriptions can be found in these websites (<http://www.oeg-upm.net/files/RESTFulAnnotationWeb/hrest.html>; <http://www.oeg-upm.net/files/RESTFulAnnotationWeb/sa-rest.htm> 1)

EXPERIMENTAL RESULTS

In order to evaluate our approach in the geospatial domain we have used 60 different RESTful services from <http://www.programmableweb.com/>, which we have selected randomly from those that were available and could be characterized to contain geospatial information by a manual lookup. The list of these services can be found in our experiment website (<http://www.oeg-upm.net/files/RESTFulAnnotationWeb/SourcesList/sources.ods>). In the syntactic registration of all these services in the system, by means of introducing the list of their URLs, our system successfully registered 56 of them into

Table 5. Semantic annotation of a RESTful service

```
($country,http://www.w3.org/2003/01/geo/wgs84_pos#lat,http://www.w3.org/2003/01/geo/wgs84_pos#long,isoNumeric,http://dbpedia.org/ontology/Continent,fipsCode,http://dbpedia.org/property/areaMetroKm,la
nguages,isoAlpha3,http://dbpedia.org/ontology/country,http://www.w3.org/2003/01/geo/wgs84_pos#lat,http://www.
w3.org/2003/01/geo/wgs84_pos#long,http://dbpedia.org/ontology/populationDensity,http://www.w3.org/2003/01/geo/
wgs84_pos#lat,http://www.w3.org/2003/01/geo/wgs84_pos#long,http://dbpedia.org/ontology/Currency,http://www.
w3.org/2003/01/geo/wgs84_pos#lat,http://www.w3.org/2003/01/geo/wgs84_pos#long,http://dbpedia.org/ontology/
capitalgeonameId,http://dbpedia.org/ontology/country)
```

the repository (4 services could not be registered due to an invocation error³). As a result of this syntactic registration, the system has produced a complete list of 1,014 total parameters (152 input parameters and 862 output parameters) with some duplications. Of the 1,014 parameters, our system obtains 394 parameters without duplications (52 input parameters and 342 output parameters). In order to detect possible duplicates in the combination with input and output parameters, the system performs the union of both (input and output) parameters. From this union, 369 parameters are obtained without duplications.

This analysis follows the three steps described in our annotation process. First, our system identifies correctly 191 of 369 different parameters without duplications by calling directly the DBpedia and GeoNames ontologies. Second, the system uses initial parameters plus

the suggestion service and calls the DBpedia and GeoNames ontologies. In this case, it adds 57 additional names for parameters to the initial ones and identifies 33 correspondences with used ontologies. Third, the system uses the initial parameters plus the synonyms service, and calls the DBpedia and GeoNames ontologies. It incorporates 1,147 additional names for parameters into the system and identifies 126 correspondences. Finally, the system combines all the resources that result from the enrichment process and calls again the DBpedia and GeoNames SPARQL Endpoint. Here it adds 1,573 more names for initial parameters and identifies 159 correspondences. A detailed view of these results is shown in Table 8.

With respect to the validation of input parameters⁴ (Table 9), our system recognizes 152 inputs (with duplications) of the initial list, of which 80 parameters can be annotated auto-

Table 6. Semantic Annotation (Service 1) following the hRESTS format

```
<div class="operation" id="op1">
<a href="#http://ws.geonames.org#countryInfo">http://ws.geonames.org/countryInfo</a>
</div>
<code class="address">
http://ws.geonames.org/countryInfo
</code>
...
<div class="method">
<h4 id="http://ws.geonames.org#Method 1"><span class="method">GET</span></h4>
<h6>request query parameters</h6>
...
<span class="input">
<strong>country</strong>
</span>
...
<span class="output">
<strong>countryCode</strong>
</span>
```

Table 7. Semantic annotation (Service 1) following the SA-REST format

<pre><div class="section domain-rel" lang="en" title="sarest:Service" > ... http://ws.geonames.org/countryInfo ... <code class="address"> http://ws.geonames.org/countryInfo </code> ... country ... <td> This is the input service </td> ... countryName ... <td> This is the output service </td></pre>
--

matically with the DBpedia (35 parameters) and GeoNames (45 parameters) ontologies.

Likewise, we have discovered with our evaluation that some other parameters are useless in terms of semantic annotation processes, due to lack of annotations or meaningful results. These parameters are called “special” ones. After the exhaustive analysis performed in this evaluation, we classify these parameters into two groups:

- **Navigation.** Those parameters referred to the navigation process through the RESTful service results, for example: link, userID, hits, total, pages, etc.
- **Unknown.** Those parameters described by only one letter (e.g., q, l, s, etc.), and hence not sufficiently descriptive for our automated approach to find any correspondence.

These parameters are not considered for this validation, concretely 155 navigation parameters and 72 unknown parameters, which were detected manually and a list of them is collected in this website (<http://www.oeg-upm.net/files/RESTfulAnnotationWeb/parameters/Parameters.ods>).

net/files/RESTfulAnnotationWeb/parameters/Parameters.ods’.

One aspect of our system is that we cannot always guarantee a successful annotation, because in some cases the system cannot find any correspondence between the service parameters and components of the DBpedia, GeoNames or WGS84 ontologies / vocabulary. This is common, for instance, when RESTful services contain the aforementioned parameters. In these cases the parameters should be shown to users for a manual description of them. Our system takes them out automatically from the service registration process⁵.

In summary, for 56 of the 60 initial geospatial RESTful services we have obtained correct input parameter associations, except for 4 cases where we could not find any correspondence.

With respect to the validation of output parameters⁶ (Table 9), our system recognizes 862 outputs (with duplications) that belong to the 56 services whose input parameters have been validated. This total of output parameters is divided into 315 whose correspondences can be found using the DBpedia (202 parameters) and

Table 8. Enriching initial parameters with external resources

	Attributes	Unique-Parameters	Additional Names ⁺	Parameters & Names	Matching with used Ontologies	Total Matchings
(a)	Parameters*	369	-	369	191	191
(b)	Alternative Spellings (Suggestions)	-	57	426	33	224
(c)	Synonyms	-	1,147	1,516	126	317
(d)	Suggestions + Synonyms	-	1,204	1,573	159	350

⁺*Additional names*: It enriches initial parameters with suggestions and synonyms of each initial parameter.

**Parameters*: Number of parameters without duplications.

GeoNames (113 parameters) ontologies, and 387 parameters (navigation (298) and unknown (89)) that are not considered for this validation, because they cannot be found.

While in the context of the input parameters we are interested in determining whether we can call the service or not, in the case of output parameters, we are interested in the precision and recall metrics of the annotation process. Hence, we have generated a gold standard with the studied services in order to assign manually the annotations that have to be produced for all output parameters of these services, and we have performed an evaluation of the results obtained from the system for the parameters that are found. Regarding the parameters that are found (Table 10), our system annotates 315 of them automatically, from which 315 parameters are annotated correctly according to the gold standard, while 160 parameters are not annotated by the system. This provides us with an average value for precision (the fraction of annotated parameters that are retrieved, that is,

$$(Right\ parameters) / (Annotated\ parameters)$$

equal to 0.85 and recall (the fraction of relevant annotated parameters that are retrieved, that is,

$$(Right\ parameters) / \{total\ parameters - Special\ parameters\}$$

equal to 0.56 for both metrics.

To the best of our knowledge, there are no available results from existing research works to compare our results against. Likewise, these preliminary results prove the feasibility of our system and highlight that its possible to carry out an assisted semantic annotation of RESTful services.

CONCLUSION AND FUTURE WORK

In this paper we have proposed an approach to support the semantic annotation of RESTful ser-

Table 9. Results of the input and output parameters

RESTful Service	Total Parameters	Annotated Parameters	Annotated Parameters (DBpedia)	Annotated Parameters (GeoNames)	Service Validation	
Input parameters	152	80	35	45	56✓	4✗
Output parameters	862	315	202	113	-	

Table 10. Output parameters metrics

RESTful Service	Total Parameters	Not found Parameters	Annotated	Annotated & not Annotated	Right Parameters	Precision	Recall
Output parameters	862	89	315	475	315	0.85	0.56

vices. This process is implemented in a system that takes into account the DBpedia ontology for general annotation, and GeoNames and the WGS84 vocabulary for geospatial specific results, as well as different external resources such as synonyms and suggestion services. We use combinations of these resources to discover meanings for each of the parameters of the RESTful services that a user may select and perform semantic annotations of them.

To illustrate our work and guide the explanations of the proposed semantic annotation process we have used two exemplary RESTful services related to the geospatial domain. Besides, we have presented some preliminary experimental results that prove the feasibility of our approach, at least in the geospatial domain, and show that it is possible to assist the semantic annotation of RESTful services, again at least in this domain.

Future work will focus on the development of a GUI to ease the introduction of existing services by users for their semantic annotation, probably incorporated in any existing RESTful semantic annotation tool/utility suite.

Furthermore, we also plan to make improvements to the proposed system through the analysis of instances retrieved in the matching process. In the same sense, we also aim at improving the SPARQL queries to DBpedia and other cross-domain semantic resources (such as, OpenCyc, <http://www.opencyc.org/>, UMBEL (Bergman & Giasson, 2008), etc.), to better explore this resource in the annotation process, and optimize the use of suggestion and synonyms services.

Finally, we will incorporate more specific domain ontologies in the semantic process for taking advantage of specific domain charac-

teristics. It is important to comment that the success of our approach is strongly based on the existence of domain ontologies that provide good coverage for the domain of the annotated services.

ACKNOWLEDGMENTS

This work has been supported by the R&D project *España Virtual* (CENIT2008-1030), funded by Centro Nacional de Información Geográfica and CDTI under the R&D programme Ingenio 2010, and the myBigData project (TIN2010-17060).

REFERENCES

- Adamczyk, P., Smith, P. H., Johnson, R. E., & Hafiz, M. (2011). REST and Web services: In theory and in practice. In Wilde, E., & Pautasso, C. (Eds.), *REST: From research to practice*. New York, NY: Springer. doi:10.1007/978-1-4419-8303-9_2
- Adida, B., Herman, I., & Sporny, M. (Eds.). (2012). *RDFa 1.1 primer*. Retrieved from <http://www.w3.org/TR/xhtml1-rdfa-primer/>
- Alarcon, R., & Wilde, E. (2010). Linking data from RESTful services. In *Proceedings of the Third Workshop on Linked Data on the Web*, Raleigh, NC.
- Alowisheq, A., Millard, D. E., & Tiropanis, T. (2009). EXPRESS: Expressing restful semantic services using domain ontologies. In *Proceedings of the 8th International Semantic Web Conference* (pp. 941-948).
- Altinel, M., Brown, P., Cline, S., Kartha, R., Louie, E., & Markl, V. ... Singh, A. (2007). Damia: A data mashup fabric for intranet applications. In *Proceedings of the 33rd International Conference on Very Large Data Base Endowment* (pp. 1370-1373).

- Ambite, J. L., Darbha, S., Goel, A., Knoblock, C. A., Lerman, K., Parundekar, R., & Russ, T. A. (2009). Automatically constructing semantic web services from online sources. In *Proceedings of the International Semantic Web Conference* (pp. 17-32).
- Auer, S., Bizer, C., Cyganiak, R., Erling, O., Idehen, K., & Kobilarov, G. ... Stadler, C. (2005). *The DBpedia ontology*. Retrieved from <http://wiki.dbpedia.org/Ontology>
- Battle, R., & Benson, E. (2008). Bridging the semantic web and web 2.0 with representational state transfer (REST). *Web Semantics*, 6, 61-69. doi:10.1016/j.websem.2007.11.002
- Bergman, M., & Giasson, F. (2008). *UMBEL*. Retrieved from <http://www.umbel.org>
- Braga, D., Ceri, S., Martinenghi, D., & Daniel, F. (2008). Mashing up search services. *IEEE Internet Computing*, 12(5), 16-23. doi:10.1109/MIC.2008.105
- Doan, A., Domingos, P., & Halevy, A. Y. (2001). Reconciling schemas of disparate data sources: A machine-learning approach. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (pp. 509-520).
- Doan, A., Domingos, P., & Halevy, A. Y. (2003). Learning to match the schemas of data sources: A multistrategy approach. *Machine Learning*, 50(3), 279-301. doi:10.1023/A:1021765902788
- Fensel, D., Fischer, F., Kopecký, J., Krummenacher, R., Lambert, D., & Vitvar, T. (2010). WSMO-Lite: Lightweight semantic descriptions for services on the Web. Retrieved from <http://www.w3.org/Submission/WSMO-Lite/>
- Fielding, R. (2000). *Architectural styles and the design of network-based software architectures* (Unpublished doctoral dissertation). University of California, Irvine, Irvine, CA.
- Filho, O. F. F., & Ferreira, M. A. G. V. (2009). Semantic Web services: A RESTful approach. In *Proceedings of the IADIS International Conference WWW/INTERNET*, Rome, Italy (pp. 169-180).
- García Rodríguez, M., Álvarez, J. M., Berrueta, D., & Polo, L. (2009). Declarative data grounding using a mapping language. *Communications of Systemics and Informatics World Network*, 6, 132-138.
- Geyer, C. (2007). *Service data objects (SDO)*. Retrieved from <http://www.oasis-open.org/sdo>
- Haas, H., & Brown, A. (Eds.). (2004). *Web services glossary*. Retrieved from <http://www.w3.org/TR/ws-gloss/>
- Hadley, M. (2009). *Web application description language*. Retrieved from <http://www.w3.org/Submission/wadl/>
- Heß, A., & Kushmerick, N. (2003). Learning to attach semantic metadata to web services. In D. Fensel, K. Sycara, & J. Mylopoulos (Eds.), *Proceedings of the International Semantic Web Conference* (LNCS 2870, pp. 258-273).
- Kopecký, J., Gomadam, K., & Vitvar, T. (2008). hRESTS: An HTML microformat for describing restful web services. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology* (pp. 619-625).
- Kopecký, J., Vitvar, T., Bournez, C., & Farrell, J. (2007). SAWSDL: Semantic annotations for WSDL and XML schema. *IEEE Internet Computing*, 11(6), 60-67. doi:10.1109/MIC.2007.134
- Lambert, D., & Domingue, J. (2008). Grounding semantic web services with rules. In *Proceedings of the Conference on Semantic Web Applications and Perspectives*, Rome, Italy.
- Lathem, J., Gomadam, K., & Sheth, A. P. (2007). SA-REST and (S)mashups: Adding semantics to restful services. In *Proceedings of the International Conference on Semantic Computing* (pp. 469-476).
- Lerman, K., Plangprasopchok, A., & Knoblock, C. A. (2007). Semantic labeling of online information sources. *International Journal on Semantic Web and Information Systems*, 3(3), 36-56. doi:10.4018/jswis.2007070102
- Maleshkova, M., Gridinoc, L., Pedrinaci, C., & Domingue, J. (2009). Supporting the semi-automatic acquisition of semantic restful service descriptions. In *Proceedings of the 6th European Semantic Web Conference*.
- Maleshkova, M., Kopecký, J., & Pedrinaci, C. (2009). Adapting SAWSDL for semantic annotations of restful services. In *Proceedings of the Confederated International Workshops and Posters on On the Move to Meaningful Internet Systems: ADI, CAMS, EI2N, ISDE, IWSSA, MONET, OnToContent, ODIS, ORM, OTM Academy, SWWS, SEMELS, Beyond SAWSDL, and COMBEK*, Vilamoura, Portugal (pp. 917-926).
- Maleshkova, M., Pedrinaci, C., & Domingue, J. (2009). Semantically annotating restful services with SWEET. In *Proceedings of the 8th International Semantic Web Conference*, Washington, DC.

Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., & McIlraith, S. ... Sycara, K. (2004). *OWL-S: Semantic markup for web services*. Retrieved from <http://www.w3.org/Submission/OWL-S/>

Martin, D., Paolucci, M., & Wagner, M. (2007). Towards semantic annotations of web services: OWL-S from the SAWSDL perspective. In *Proceedings of the OWL-S Experiences and Future Developments Workshop at the European Semantic Web Conference*, Innsbruck, Austria.

Paolucci, M., Wagner, M., & Martin, D. (2007). Grounding OWL-S in SAWSDL. In B. J. Krämer, K.-J. Lin, & P. Narasimhan (Eds.), *Proceedings of the 5th International Conference on Service-Oriented Computing* (LNCS 4749, pp. 416-421).

Pautasso, C., Zimmermann, O., & Leymann, F. (2008). Restful web services vs. "big" web services: Making the right architectural decision. In *Proceedings of the 17th International Conference on World Wide Web* (pp. 805-814).

Pedrinaci, C., Domingue, J., & Krummenacher, R. (2010). Services and the Web of data: An unexploited symbiosis. In *Proceedings of the AAAI Spring Symposium: Linked Data Meets Artificial Intelligence*.

Rahm, E., & Bernstein, P. (2001). On matching schemas automatically. *Very Large Data Base Journal*, 10(4).

Ranabahu, A., Parikh, P., Panahiazar, M., Sheth, A., & Logan-Klumpler, F. (2011). Kino: A generic document management system for biologists using SA-REST and faceted search. In *Proceedings of the Fifth IEEE International Conference on Semantic Computing*, Palo Alto, CA (pp. 205-208).

Resende, L. (2007). Handling heterogeneous data sources in a SOA environment with service data objects (SDO). In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (pp. 895-897).

Steinmetz, N., Lausen, H., & Brunner, M. (2009). Web service search on large scale. In *Proceedings of the 7th International Joint Conference on Service-Oriented Computing* (pp. 437-444).

ENDNOTES

- ¹ For presentation purposes in this paper we show XML responses.
- ² This classification is based on previous analysis of experimental results performed over different datasets.
- ³ These errors are caused by malformed XML files and therefore the SDO API cannot process these services.
- ⁴ A detailed analysis on these input parameters is available at: http://www.oeg-upm.net/files/RESTFulAnnotationWeb/inputs/inp_uts.ods
- ⁵ This was not described in the process described in section 3 since we did not consider it relevant for the description of the whole process.
- ⁶ A detailed analysis on these output parameters is available at: http://www.oeg-upm.net/files/RESTFulAnnotationWeb/outputs/out_puts.ods

Victor Saquicela is a PhD student in Artificial Intelligence at the Universidad Politécnica de Madrid. He has been a member of the Ontology Engineering Group since 2008 in the Artificial Intelligence Department of the Faculty of Informatics. He is an Engineer in Informatics (2002) from Universidad de Cuenca in Ecuador and Master in Software Engineering (2004) from Universidad Politécnica de Madrid in Spain. He has previously worked as programmer analyst and network administrator. His research activities are focused on Semantic Web, Data Integration, Geospatial Semantic Web and Linked Data.

Luis M. Vilches-Blázquez is a Post-Doc at Ontology Engineering Group of the Artificial Intelligence Department of the Computer Science School at Universidad Politécnica de Madrid (UPM). He holds an MSc on Geography from the University of Granada (Spain) and also holds a Master's degree in Geographical Information Systems (GIS) Project Management since 2004. Additionally, he was a grant holder at the National Geographic Institute of Spain, a research centre belonging to the Ministry of Civil Works (Ministerio de Fomento). His research activities are focused on geospatial ontologies, geographical databases, information integration and disambiguation, and Linked Data. He has participated at several European (Towntology, DIGMAP and DynCoopNet) and Spanish (Geobuddies, WEBn+1, Autores 3.0, and España Virtual, Ciudad2020) projects. Within this context, he has co-authored numerous publications in journals, books, book chapters, conferences and workshops, and is part of the program committee of several conferences and workshops. Finally, he is also a member of the AENOR CTN148 Geographical Information Committee, the corresponding Spanish Committee that participates in the Working Group of ISO 1950, and the ISO/TC 211 ad hoc group on Linked Data.

Oscar Corcho is an Associate Professor at UPM. Previously, he worked at the University of Manchester and iSOCO. He graduated in Computer Science (2000), he obtained his MSc in Software Engineering (2001) and his PhD in Artificial Intelligence (2004) at UPM. He worked as a Marie Curie research fellow at the University of Manchester and as an R&D manager at iSOCO. He has participated at several FP5, FP6 and FP7 projects and he published two books, over 50 international journal and conference/workshop papers, and reviews papers in many conferences and journals. He was part of the organisation committees of EKAW2002, ESWC2006, ESWC2008, ISWC2009 and EKAW2010, and will be PC chair at KCAP2011.

CALL FOR ARTICLES

International Journal on Semantic Web and Information Systems

An official publication of the Information Resources Management Association

The Editor-in-Chief of the *International Journal on Semantic Web and Information Systems* (IJSWIS) would like to invite you to consider submitting a manuscript for inclusion in this scholarly journal.

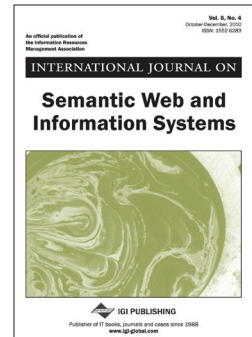
MISSION:

The *International Journal on Semantic Web and Information Systems* (IJSWIS) is an archival journal that publishes high quality original manuscripts in all aspects of Semantic Web that are relevant to computer science and information systems communities. IJSWIS is an open forum aiming to cultivate the Semantic Web vision within the information systems research community. The main focus is on information systems discipline and working towards the delivery of the main implications that the Semantic Web brings to information systems and the information/knowledge society.

COVERAGE/MAJOR TOPICS:

Topics to be discussed in the journal include (but are not limited to) the following:

- Beyond Semantic Web (e.g., extending meaning with perception and experience)
- Enterprise application integration
- From e-government to e-democracy
- Integration with other disciplines
- Intelligent systems
- Metadata-driven (bottom-up) versus ontology-driven (top-down) SW development
- New Semantic Web enabled business models
- New Semantic Web enabled information systems
- New Semantic Web enabled tools for the citizen/ learner/ organization/ business
- Ontologies, folksonomies, and associated knowledge representation issues
- Real world applications towards the development of the Knowledge society
- Semantic enabled business intelligence
- Semantic Web applications on the Web, enterprises, desktops, personal and mobile devices, e-science and e-government applications, and associated issues of provenance, trust, privacy, security, quality, scalability, and performance
- Semantic Web data management
- Semantic Web issues, challenges, and implications in each of the IS research streams
- Semantics and human computer interfaces including visualization and mashups
- Semantics in business processes and distributed computing and services
- Social Semantic Web and people Web
- Standards



ISSN 1552-6283
eISSN 1552-6291
Published quarterly

All submissions and inquiries should be directed to the attention of:

Tonya Davis or Amit Sheth
E-mail: ijswis@knoesis.org;
amit.sheth@wright.edu

Ideas for Special Theme Issues may be submitted to the Editor-in-Chief.

Please recommend this publication to your librarian. For a convenient easy-to-use library recommendation form, please visit:
<http://www.igi-global.com/ijswis>