Application of SWSAL in Semantic Annotation of RESTful Web Services

Francesco Pagliarecci, Luca Spalazzi, Gilberto Taccari

Università Politecnica delle Marche – Ancona, Italy
{pagliarecci | spalazzi | taccari}@dii.univpm.it

Abstract. Nowadays web services have become one of the main technologies in the development of web applications. According to that providers now offer an increasing number of capabilities as web services. Furthermore, in the recent years, such deployment trend has seen the success of REST architecture and, consequently, the proliferation of RESTful web services. This work focuses on the semantic description of RESTful web services. It shows how the SWSAL language, already used profitably to semantically annotate SOAP web services, can be used to semantically describe a web service compliant with the REST principles. The work establishes the bases for the application of SWSAL and its related researches to the RESTful web service area.

1 Introduction

Nowadays, service-oriented architecture is becoming the key element in the development of web and mobile applications. Furthermore cloud technologies are often deployed as services that can be used through HTTP methods interactions. As a consequence, there is a rich ecosystem of services that can be used to implement business processes tasks and/or develop applications. In the recent years, beyond "Big" Web Services [1], a new approach to web services deployment is spreading: HTTP-based web services and, in particular, RESTful web services are becoming the main technologies to offer web services to requesters [2].

Beside the increasing number of available web services, some works [3][4][5] pointed out the need of semantically describe web services in order to make ease the interaction with/among them. Indeed, the interoperability among heterogeneous services can be difficult or even impossible when there are different schemas and different names for the same concepts.

For these reasons, the goal of this paper is proposing an approach to semantically describe RESTful web services (or, in general, HTTP-based web services). Several past works try to do that: some of them use Resource Description Framework (RDF) [6] triples to describe the RESTful resources; other approaches use semantic annotations over interface description of web services. The approach explained in this paper shows how Semantic Web Service Annotation Language (SWSAL) [7], already used in [8] to annotate SOAP Web Services, can be profitably used also to semantically annotate RESTful web services, as well. This approach is based on the following technologies: Web Application Description Language (WADL) [9], used to describe HTTP-based applications; Web Service Modeling Language (WSML) [10], used to describe ontologies and SWSAL, used to link WADL elements with ontology concepts.

2 Related Work

Despite Fielding defined REST architecture in his PhD thesis dissertation in 2000 [11], only in the recent years researchers have tried to define a model to represent RESTful web services and semantically describe them. Furthermore, unlike SOAP web services for which it has been defined standards and languages to work with them, for the RESTful web services it has not been defined any standard to describe them: in many cases the only available description of a services is a human-readable web-page. The lack of standard let researchers and service providers free to propose their own models and/or description languages for RESTful web services. Proposed approaches can be divided in two main categories: on one hand there are those that modeling service interfaces and on the other hand there are those that describing the resources that compose the services.

hRESTS [12] and SA-REST [13] enrich the human-readable RESTful web services documentation with semantic annotation in order to make it machine-processable. While hRESTS uses micro-format to annotate the service and allows only input and output annotations, SA-REST uses Resource Description Framework in Attributes (RDFa) and allows a richer semantic description borrowing the tenets of Semantic Annotation for WSDL (SAWSDL) [14], a language to semantically annotate SOAP web services.

In [15] authors present the Semantic Bridge for Web Services (SBWS), a Java tool that wraps the WADL description file of a RESTful web service and creates a SPARQL [16] endpoint o interact with the web service resources. Such endpoint is responsible of the understanding of the query and the translation of web services returned data to RDF tuples. Indeed, in this work, data returned to the service requester are RDF tuples that refer to ontology. Thereof it is suitable for data web services.

In [17] the authors propose a SEREDASj, a semantic description language written in JSON used to define RESTful web services metadata and description. Metadata refer to information required to access the web service and navigate the resources while description refers to resources instances descriptions. Each described element is tied with a concept of an ontology in order to give it a semantic description. Furthermore it is possible to translate SEREDASj representations to RDF triples even if in some cases information can be lost.

A similar approach is used in [18], where the authors propose Resource Linking Language (ReLL), a language to describe a resource of a web service. It models a resource using three main elements: resource, link, and representation. It also permits to annotate elements using ontologies. After describing the resource, RESTler [19], a crawler, reads the ReLL file and generates RDF triples analyzing elements described by the file. RDF triples are stored in a Triplestore that can be accessed using SPARQL.

In [20] authors shown the EXPressing REstful Semantic Services (EXPRESS) approach. In this case the developer designs a service's ontology defining entities and relationships in an OWL file. Then the EXPRESS deployment engine creates a RESTful interface which fulfills with the ontology. Using this approach a service requester have to know only the ontology: knowing entities and relationships she/he can navigate the resources and manipulate them according the uniform interface of RESTful web services.

3 Web Services Interface Description

As said above, in recent years RESTful web services have received much attention from developers, services providers and scientific community. Despite of this interests, REST architecture still misses a standard language to describe resources in a resource-oriented architecture and often RESTful web services are described using natural language. Actually a few of languages are proposed or extended to describe these web services. In particular, the main two languages used to describe the interface of RESTful web services are WSDL 2.0 [21] and WADL [9]. Version 2.0 of WSDL extends the version 1.0 of the language and introduces elements and attributes to allow RESTful services endpoints definition. Even if this language can be used to describe the interface of a RESTful web service, it lacks the resource-oriented view of the service [22] and often it is misused producing ugly verbose results [23]. WADL is a XML-based description language for HTTP-based applications. Its main goal is to model resources, hence it fulfills well the ROA of RESTful web services. Beyond these languages, other formalisms have been proposed with the focus to describe RESTful (e.g. RESTful Interface Definition and Declaration Language (RIDDL)[23]), but they remains isolated solutions tied with research works and they are not wide accepted standard interface description languages.

In this work the chosen interface description language is WADL. Although some works assert that this language does not suit well the lean resource interface description, there are some reasons that suggest that it is sufficient for the scope of semantically annotation of RESTful web services. A first reason to use WADL is its resource-oriented nature. Furthermore the wealth of tags defined by that language allows annotating many elements of the service. Another reason regard the development frameworks and/or application servers that automatically generate WADL file for deployed services; for instance GlassFish+Jersey [24] application server creates a WADL file for each Java web service developed using JAX-RS API [25]. Furthermore there are tools that guide the user in the creation of WADL files (e.g. soapUI [26]) and simplify the editing of a description document for the service.

A WADL file is composed by the following elements.

- Grammar: includes external schemas for data-types. No schema is mandatory, but language specifications propose XML Schema and RelaxNG.

- Resource: defines a RESTful web service resources and give the URI to access it. A resource can contain methods elements. And, for each method, request and response elements can be defined. They refer to the input and output of the methods and complete the needed description for the scope of this work.

WADL allows the description of many other characteristics of a RESTful web service; to delve into these description elements and attributes the reader can refer to [9].

4 Ontology Representation

“Ontologies [in Computer Science] are explicit specifications of conceptualizations” [27]. As a consequence, they can be considered a set of concepts and relations between concepts. Even if there are several slightly different definitions of the notion of ontology, most authors agree that an ontology should be defined in a formal language, which, in practice, usually means a logic-based language suitable for automating reasoning. This work is based on Description Logics [28]. According to Description Logic jargon, within a given ontology we have a terminological component (called T-BOX) and an assertional component (called A-BOX).

The T-BOX contains concept definitions as well as generalization and aggregation relationships among them. A WSML file directly provides this part of the ontology. The T-BOX simply matches the set of the declared concepts and their structure (i.e. their attributes/roles).

On the other hand, the A-BOX contains assertion definitions of two different types: concept assertions and role assertions. Concept assertions have the form *a:C*. Such an expression tells us that the individual *a* is an instance of the concept *C*. For example, *joe:Person* states that Joe is a (instance of) Person. Role assertions specify the value that a certain role of an individual has. They have the form *a.R=b*. Intuitively, such an expression means that the value of the attribute *R* of the individual *a* is *b*, where *b* is another individual or a literal. For example, *joe.Mother = mary* states that Joe has a mother whose name is Mary. Mary is the identifier of another individual.

5 SWSAL as Semantic Annotation Language

SWSAL [7] is an annotation language that allows us to link data and operation definitions of the RESTful web service (WADL) with ontology elements (WSML). The language is based on XML. Annotations are necessary to give semantics to the exchanged data (e.g., what relations exist between the data given in input to the service and the data received as answers from the service), as well as to define the effects and outcomes of service executions (e.g., to distinguish successful service executions from the failures, and to describe the effects such executions). The annotation file is defined over XML syntax, so we are going to come up with an XML schema for annotation. Let us now analyze the main sections of annotation file.

In the *datatype_annotation* section there are several assertions that map WADL elements into WSML concepts. There is no direct mapping between data and concepts at structural level. In other words, it is perfectly legal to annotate an element with a concept whose attributes do not match with those of the element neither in number nor in type.

In the *declarations* section we simply list all the individuals used in the assertions. We make this assumption: individuals with the same name (*id*) refer to the same object. So, if we had a pair of assertions like “god.Exists = True” and “god.Exists = False” we would refer to the same instance ’god’, no matter if the A-BOX would be inconsistent.

The *procedural_annotation* section contains sets of assertions and it is always referred to an activity. An assertion is composed of a set of *concept_assertions*, *axiom_assertion* and *role_assertions*. Let us now delve into the structure of concept and role assertion. A concept assertion tells us that in a given activity, there exists an individual of a certain concept. The individual can be a brand new one, with a name chosen by the user. Role assertions express a relation between an attribute (role) of a certain individual and another individual. The left part is an individual previously declared. The role points to one of the attributes of the concept to which the individual belongs to. A WSML URI identifies the role. The right part of a role assertion may be another individual. Alternatively, the right part may be an individual directly declared in the ontology.

6 Annotating Semantically RESTful Web Services

According to [29], resources, methods, parameters and representations are the fundamental elements that characterize a RESTful web service. Their definitions give an adequate description of a service and provide enough information to properly interact with it. For this reason, the approach proposed here suggests to add semantic annotations to these elements in order to semantically describe the service. This section explains how such elements are described in a WADL document and how the SWSAL annotation language can be used to semantically annotate them. An example of the approach is available at http://www.pagliarecci.it/swsal4rest_foursquare.zip. In such example the Foursquare API[1], defined by the WADL file[2], is semantically annotated with concepts taken from the Venue ontology.

As said in the previous sections, the main components of a resource-oriented architecture are resources. A RESTful web service can be seen as a collection of resources referenced by URIs. Hence, annotating a resource with a concept of the domain ontology gives a semantic meaning to the resource: it declares the “type” of the resource. In the SWSAL file, it corresponds to a datatype annotation over the resource tag that identifies the resource.

Methods are the HTTP operations that can be called on the URI of a resource. They represent the uniform interface in the resource-oriented architecture as defined in [11]. Furthermore, for a pure RESTful web services, the semantic of these methods

[1] Foursquare API, https://developer.foursquare.com/, 2012.

[2] Foursquare WADL description file is taken from Apigee's github repository available at https://github.com/apigee.

is given: HTTP methods GET, POST, PUT and DELETE are the CRUD (Create, Read, Update, Delete) operations used to manipulate the resources. At first sight, this could suggest that these methods doesn't need any semantic description: requester and provider of the service apparently know their semantic, but in some cases it is convenient to annotate also these methods; this is the case of non-pure RESTful web services where some operations are expressed using the RPC style over the HTTP protocol and whenever it is needed to enrich the semantic meaning of a method. In these cases, it is possible to add a procedural annotation over the desired method tag defined by the WADL file.

Parameters and representations represent the exchanged data between requester and provider of the service. Representation is one of the principles of the REST architecture [11]: it consists of resource's state represented using the data format negotiated between the service requester and the service. Instead, parameters are mainly used with methods definitions. They can also be used within the description of resources and representations. Hence, parameters and representations both define exchanged data and they can be annotated with datatype annotations over, respectively, the param and representation tags.

Furthermore, in WADL language, a representation can also be externally defined using an external document, for example a XML Schema file. In this case, semantic annotations can be added to these external description files.

7 Discussion and Conclusions

This work shows how the SWSAL language can be used to semantically describe RESTful web services and, in general, HTTP-based web services. The proposed approach allows a web service developer or provider to semantically annotate the basic web service elements linking them with concepts taken from a domain ontology. This information is stored in an external SWSAL annotation file. This keeps unchanged any original documentation file has to be modified in contrast with [13] and [14].

Another advantage of this approach is the use of WADL. Despite some works assert that the WADL does not suitably describe RESTful web services, this work has shown how its expressiveness is enough for the description purpose. Furthermore, from a practical point of view, using frameworks and/or tools it is easy for a web service developer or provider to get or write a WADL file that describes a service.

Concluding, adding semantic description to RESTful web services can be straightforward and fast. Semantically described RESTful web services allow service users as well as machines to know what is the semantic of the resources that it uses and of the operations that it performs. Moreover it eases web services typical operations such as discovery, selection and composition.

Future works will concern the selection and composition of semantically annotated RESTful web services and their interaction with SOAP web services. Furthermore, it will be investigated the resource-oriented business process area and how semantically annotated RESTful web services could be used to implement business processes.

References

1. C. Pautasso, O. Zimmermann, and F. Leymann. Restful web services vs. "big"' web services: making the right architectural decision. In *Proceedings of the 17th international conference on World Wide Web*. ACM, New York, NY, USA, pp. 805-814, 2008
2. ProgrammableWeb, http://www.programmableweb.com/
3. E. W. Service, T. Sollazzo, S. H, S. Staab, M. Frank, and N. Stojanovic, “Semantic web service architecture–evolving web service standards toward the semantic web,” in In: Proceedings of the 15th International FLAIRS Conference, pp. 16–18, AAAI Press, 2002.
4. Kunal Verma, Amit Sheth, "Semantically Annotating a Web Service," IEEE Internet Computing, pp. 83-85, March/April, 2007
5. Lanthaler, M., Gutl, C., "Towards a RESTful service ecosystem," Digital Ecosystems and Technologies (DEST), 4th IEEE International Conference, 13-16 April 2010 pp.209-214
6. Resource Description Framework (RDF), RDF Working Group, February 2004, avaliable at http://www.w3.org/RDF/
7. Di Pietro, I.; Pagliarecci, F.; Spalazzi, L.; “Semantic Annotation for Web Service Processes in a Pervasive Computing”, Pervasive Computing: Innovations in Intelligent Multimedia and Applications, Springer—Verlag, Berlin, Germany, 2009
8. Di Pietro, I.; Pagliarecci, F.; Spalazzi, L.; "Model Checking Semantically Annotated Services," IEEE Transactions on Software Engineering, vol.38, no.3, pp.592-608, May-June 2012
9. Web Application Description Language (WADL), Marc Hadley and Sun Microsystems, Inc, available at, http://www.w3.org/Submission/wadl/
10. Web Service Modeling Language (WSML), J. de Bruijn and H. Lausen ed., W3C Member Submission, June 2005, available at http://www.w3.org/Submission/WSML/
11. R. Fielding, Architectural styles and the design of network-based software architectures. PhD thesis, University of California, 2000.
12. Kopecky, J.; Gomadam, K.; Vitvar, T.; , "hRESTS: An HTML Microformat for Describing RESTful Web Services," Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT '08. IEEE/WIC/ACM International Conference on, vol.1, no., pp.619-625, 9-12 Dec. 2008
13. Sheth, A.P.; Gomadam, K.; Lathem, J.; , "SA-REST: Semantically Interoperable and Easier-to-Use Services and Mashups," Internet Computing, IEEE, vol.11, no.6, pp.91-94, Nov.-Dec. 2007
14. Semantic Annotations for WSDL and XML Schema, Farrell J. And Lausen H., W3C Recommendation, August 2007, available at http://www.w3.org/2002/ws/sawsdl/
15. Robert Battle, Edward Benson, “Bridging the semantic Web and Web 2.0 with Representational State Transfer (REST)”, Web Semantics: Science, Services and Agents on the World Wide Web, Volume 6, Issue 1, February 2008, Pages 61-69
16. SPARQL Query Language for RDF, Prud'hommeaux E. and Seaborne A., W3C Recommendation, January 2008, available at http://www.w3.org/TR/rdf-sparql-query/
17. Lanthaler, M.; Gutl, C.; , "A semantic description language for RESTful Data Services to combat Semaphobia," Digital Ecosystems and Technologies Conference (DEST), Proceedings of the 5th IEEE International Conference, May 31 2011-June 3 2011, pp.47-53
18. R. Alarcon and E. Wilde, “Linking data from restful services,” in 3rd Inter-national Workshop on Linked Data on the Web, Raleigh, North Carolina, USA, 2010.
19. Rosa Alarcón and Erik Wilde. 2010. RESTler: crawling RESTful services. In *Proceedings of the 19th international conference on World wide web* (WWW '10). ACM, New York, NY, USA, 1051-1052.

20. Alowisheq, A.; Millard, D. E.; "EXPRESS: EXPressing REstful Semantic Services," Web Intelligence and Intelligent Agent Technologies, WI-IAT '09. IEEE/WIC/ACM International Joint Conferences, 15-18 Sept. 2009, vol.3, pp.453-456
21. Web Services Description Language (WSDL) 2.0 Chinnici R., Moreau J., Ryman A., Weerawarana S., W3C Recommendation, 2007, available at http://www.w3.org/TR/wsdl20/
22. Toshiro, T.; Makino, S.; Kawanaka, S.; Ueno, K.; Ferris, C.; Ryman, A. "Definition Languages for RESTful Web Services: WADL vs. WSDL 2.0" 2008.
23. Mangler, J.; Schikuta, E.; Witzany, C.; , "Quo vadis interface definition languages? Towards a interface definition language for RESTful services,"Service-Oriented Computing and Applications (SOCA), IEEE International Conference 14-15 Jan. 2009, pp.1-4
24. Jersey, http://jersey.java.net/
25. JAX-RS, http://jcp.org/en/jsr/detail?id=311
26. soapUI, http://www.soapui.org/
27. F. Giunchiglia and I. Zaihrayeu, "Lightweight ontologies," DIT, University of Trento, Tech. Rep. DIT-07-071, Oct. 01 2007.
28. F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, Eds., The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, 2003.
29. Silvia Schreier. Modeling RESTful applications. In *Proceedings of the Second International Workshop on RESTful Design* (WS-REST '11), ACM, New York, NY, USA, 15-21.