The Semantic Web Services Tetrahedron: Achieving Integration with Semantic Web Services*

Juan Miguel Gómez[1], Mariano Rico[2], Francisco García-Sánchez[3], and César J. Acuña[4]

[1] DERI Ireland, National University of Ireland, Galway, Ireland
juan.gomez@deri.org
[2] Universidad Autónoma de Madrid
Mariano.rico@uam.es
[3] Universidad de Murcia
frgarcia@um.es
[4] Kybele Research Group, Universidad Rey Juan Carlos
cesar.acuna@urjc.es

Abstract. Web Engineering is going through several major changes. New promising application fields related to the Web such as the Semantic Web and Web Services, and its combination, Semantic Web Services, are transforming the Web from a mere repository of information into a new vehicle for business transactions and information exchange. However, the lack of integration among different Web Applications such as Web Portals and web services is hindering the potential leverage of the Web. In this paper, we present a novel approach for integration in Web engineering based on Semantic Web Services. Our approach enables the combination in a Tetrahedron of a fully-fledged architecture consisting of three layers. An application integration layer, a second layer based on goal-oriented and intelligent user-interaction and finally, a common middleware platform for Semantic Web Services. We describe the implementation of our architecture and its benefits by achieving full integration all over Web applications.

1 Introduction

The Web is changing from a mere repository of information to a new vehicle for business transactions and information exchange. Large organizations are increasingly relying on Web Services technologies for large-scale software development and sharing or exposing of services within an organization. Web Services are loosely coupled software components published, located and invoked across the web. The growing number of Web Services available on the Web raises a new and challenging problem, the integration among heterogeneous applications.

* This work is founded by the Ministry of Science and Technology of Spain under the project DAWIS (TIC2002-04050-C02-01) and Arcadia (TIC2002-1948); by the European Commission under the projects DIP, Knowledge Web, Ontoweb, SEKT, and SWWS; by Science Foundation Ireland under the DERI-Lion project; and by the Austrian government under the CoOperate programme. We also thanks FUNDESOCO for it support through the project FDS-2004-001-01

The Semantic Web is about adding machine-understandable and machine-processable metadata to Web resources through its key-enabling technology: ontologies. Ontologies are a formal, explicit and shared specification of a conceptualization. The breakthrough of adding semantics to Web Services leads to the Semantic Web Services paradigm. Several Semantic Web Services initiatives such as OWL-S [7], which offers semantic markup based on OWL [2] or the newly established WSMO [9] have recently gained momentum.

However, the problem of integration remains since, to our knowledge, nobody has attempted to define architecture and implementation to integrate several Semantic Web Services from a rich user-interaction perspective i.e. by enabling specifying user request with natural language and presenting the results properly. In this paper, we present a novel approach to address those challenges and solve the integration problem regarding to searching, finding, interacting and integrating Semantic Web Services. Our contribution is an overall solution based on a Tetrahedron composing a fully-fledged architecture consisting of three layers. The presentation layer deals with intelligent user human-interaction through the Jacinta [8] and GODO [3] applications. The intermediate layer is based on WSMX [6] a common middleware platform for Semantic Web Services and finally, an integration layer composed of Web Services and Web Portal wrapper applications.

The remainder of the paper is organized as follows. Section 2 describes the common middleware platform for Semantic Web services. In Section 3 and Section 4, Jacinta and GODO, the components of the presentation layer are introduced. Section 5 presents the integration layer regarding Web Portals and finally, the whole architecture is depicted in Section 6. Conclusions and Related Work are discussed in Section 7.

2 WSMX: Middleware for Semantic Web Services

WSMX is the reference implementation for WSMO, aiming to provide both a test bed for WSMO and to demonstrate the viability of using Semantic Web Services as a means to achieve dynamic inter-operation of Semantic Web Services.

The current architecture of WSMX provides dynamic discovery, mediation, selection and invocation and a simple but complete implementation of these components. In short, its functionality could be summarized as performing discovery, mediation selection and invocation of several Web Services on receiving a user goal specified in WSML [1], the underlying formal language of WSMO. The user goal is matched with the description of a Web Service, then the Web Service is invoked and the result is returned. Presently the WSMX architecture relies on a set of loosely-coupled main components. The core component is the WSMX manager, an event-based driven coordinator managing the other components. The Matchmaker component attempts to match the user goal to a Capability of a Web Service and the Mediator component bridges the gap between different ontologies in the matchmaking process. The Message Parser and Compiler components deal with the understanding and interpretation of WSML messages. Various repositories store descriptions and ontologies of Web Services. Transforming message formats is a task for the Message Adapters and finally, the Invoker component performs the required SOAP request (invocation) to the chosen Web Service.

3 Jacinta: User-Interaction for Semantic Web Services

In principle, the Semantic Web could be viewed as a "web for agents, not for humans", but this is only the case of the B2B aspect. The B2C aspect has to deal with a human user. As a first step in this direction, we proposed the creation of a simple, specialized type of agent, to which we will refer as Jacinta, who mediates between traditional Web Services and end users. This mediator agent is invoked by human users who interact with it using a traditional web browser, so that the mediator (Jacinta) invokes traditional Web Services on behalf of the user and show the results in the most appropriated way. Jacinta can be seen as a Semantic Agent (SA) specialized in interaction with human users. The main objective of Jacinta is not the creation of Semantic Web Services (SWS) or SA but the development of techniques for interacting with the end user. Once SA-SWS will become a reality, as it is case of WSMX (see section 2), Jacinta can be used as a module for Human-Agent interaction. Firstly, dealing with the physical features of the interaction user device such as PDA, mobile phones or other small devices. Secondly, delegating a more sophisticated aesthetic to specialized enterprises so that human user can choose the most appropriated or fashionable.

4 GODO: Goal-Oriented Discovery for Semantic Web Services

As mentioned in Section 2, WSMX [6] enables automatic discovery, selection, mediation, invocation and interoperation of the Semantic Web Services. It accepts *goals* in a specified format as input and returns the Web Services invocation results. GODO [3] can be seen as a software agent that aims at helping users to send their goals to WSMX. It can either accept a text in natural language or define a set of rules to determine the user wishes. Once GODO has inferred what the user wants, it creates the goals WSMX have to execute in order to achieve it.

GODO is composed of six main components. The most important is the *Control Manager*. It supervises the whole process and acts as an intermediary among the other components. The *Language Analyzer* is in charge of understanding the text in natural language with the users' whishes. Another important component is the *Rules Loader*, which loads the rules generated by the rules-based system. It helps users to express exactly what they want. The *Goal Loader* looks for goals either in the WSMO goal repository or in an internal repository with goals templates. These goals will be compared with the users' whishes by the *Goal Matcher*. The goals accepted are then composed by the *Control Manager*. Finally, the *Goal Sender* sends individually the goals to WSMX.

5 WP2WS: Using Web Portals as Web Services

There is still, a huge amount of information and functionality available in the Web in the form of traditional Web Portals. Nowadays lot of research and industrial efforts are focused in web services integration, but they mostly ignore the information and services offered by those traditional Web Portals. That is, only a few Web Portals offer their information and functionality in the form of Web Services (i.e. amazon.com). Achieving full web integration requires taking into account those Web Portals.

WP2WS (Web Portals to Web Services) is a tool which allows the semi-automatic generation of web extraction modules or wrappers. On the one hand, WP2WS generates the modules to automatically extract the Web Portals information and the modules to access the Web Portal services. On the other hand, it generates the WSDL Web Services description to allow the web wrapper to be invoked as a Web Service.

WP2WS follows an ontology-based approach. It uses domain-specific ontologies in conjunction with some heuristics to locate constants present in the Web Portal and constructs objects with them. WP2WS is a semi-automatic tool because while the tool is running, it requests the user for feedback to accurately construct the proper object to data extraction or to functionality access. The user also decides which of the detected data and functionality is going to be available as Web Services. Depending on the selected data and functionalities, WP2WS generates the modules in charge to process the Web Portal in order to use the offered functionality or data.

6 The Tetrahedron Approach: Achieving Integration with Semantic Web Services

The main objective of this work is to integrate and articulate in a fully-fledged architecture the described applications in order to solve the problem of integration of Web Services over the Internet. Our architecture uses WSMX as its middleware technology. It partitions into three well defined layers: presentation, intermediate and integration layer. In Figure 1 the layer model of the integrated architecture is shown.

The first layer in the figure is the presentation layer constituted by GODO and Jacinta. As middleware and facilitator for the integration we find WSMX in the second layer. Finally, WP2WS is sited in the bottom layer, as it has to provide access to the data and functionality provided by the portals and a number of Web Services distributed over the Internet.

For the sake of clarity, we will detail how the user interacts with the architecture and how it solves the user problems regarding integration. The process takes place as follows:

1. Users access GODO. Their have two possibilities for the input. They can either write a sentence in natural language or indicate what they exactly want through a set of rules.
2. It is the GODO task to analyze that input and to determine the goals it has to send to WSMX in order to achieve the user wishes.
3. WSMX receives the goals, process it, and infers the Web Services it has to invoke.
4. Some of the Web Services invoked will be the ones proportioned by WP2WS. They enable us to access to the information contained in several portals distributed on the Internet and also to their functionality.
5. WSMX receives the responses from the Web Services and send them directly to Jacinta.
6. Finally, Jacinta determines the better way this information can be shown to the users, formats it and presents it to the user.

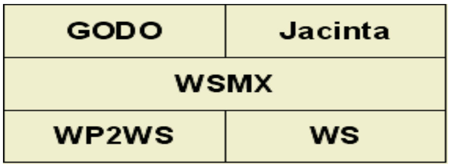

Fig. 1. Our fully-fledged layered architecture

The use of the Tetrahedron metaphor is justified by the interdependence of those applications in terms of the common middleware platform (the common vortex) on top of the three other applications.

7 Conclusions and Related Work

As the use of Web Services grows, the problem for searching, interacting and integrating relevant services will get more acute. In this paper, we proposed a layered architecture based on a Tetrahedron and an effective implementation to integrate Semantic Web Services. While this paper focuses mainly on Semantic Web Services concerns about integration, there is a whole plethora of integration problems in many other domains. The forthcomings of our approach are mainly two: namely the ease of user interaction and application integration. Ease of user interaction, since he can specify in natural language what he is expected to achieve and without worrying about heterogeneity problems and the input and results for interaction are handled in an intelligent and user-friendly manner. Application integration is achieved by means of mediation and interaction with loosely-coupled distributed software components.

There are several other approaches to the ones presented in this paper. In [5], the author points out how the ontology languages of the Semantic Web can lead to more powerful agent-based approaches for using services offered on the Web. The importance of the use of ontologies in agent-to-agent communication has also been highlighted. In [4], the authors outline their experiences of building semantically rich Web Services based on the integration of OWL-S [7] based Web Services and an agent communication language (it is done to separate the domain-specific and the domain-independent aspect of communication).

References

1. Bruijn, J.,Foxvog, D.,Fensel, D.: WSML-Core, Working Draft, 2004..
2. Dean, M. Schreiber, G.editors. *OWL Web Ontology Language Reference*. 2004. W3C Recommendation 10 February 2004.
3. Gómez, J.M, Rico, M., García-Sánchez, F., Martínez-Béjar, R., Bussler, C. (2004) GODO: Goal-driven orchestration for Semantic Web Services. Proceedings of the Workshop on Web Services Modeling Ontology Implementations (WIW 2004). Frankfurt, Alemania, September 29-30, 2004.
4. Gibbins, N., Harris, S., Shadbolt, N. (2003). Agent-based Semantic Web Services. In Proc. of the 12th Int. World Wide Web Conference, May 2003.
5. Hendler, J. (2001). Agents and the Semantic Web. IEEE Intelligent Systems, 16(2): 30-37, March/April 2001.

6. Oren, E., Zaremba, M., Moran M.: Overview and Scope of WSMX. WSMX Working Draft, 2004. Available from http://www.wsmo.org/2004/d13/d13.0/v0.1/20040611/.
7. OWL-S: Semantic Markup for Web Services. http://www.daml.org/owls
8. Rico, M., Castells, P. Jacinta: a mediator agent for human-agent interaction in semantic web services. In Proc. The Semantic Web - International Semantic Web Conference (ISWC) 2004. Selected Posters., Hiroshima, Japan, November 2004.
9. Roman, D. Lausen, H., Keller, U.: Web Service Modeling Ontology Standard. WSMO Working Draft v02, 2004. Available from http://www.wsmo.org/2004/d2/v02/20040306/