

# WebML and Glue: an integrated discovery approach for the SWS Challenge

Marco Brambilla<sup>1</sup>, Irene Celino<sup>2</sup>, Stefano Ceri<sup>1</sup>, Dario Cerizza<sup>2</sup>,  
Emanuele Della Valle<sup>2</sup>, Federico Facca<sup>1</sup>, Andrea Turati<sup>2</sup>, Christina Tziviskou<sup>1</sup>

<sup>1</sup> Dipartimento di Elettronica e Informazione, Politecnico, 20133 Milano, Italy  
{mbrambil, ceri, facca, tzivisko}@elet.polimi.it

<sup>2</sup> CEFRIEL, 20133 Milano, Italy  
{celino, cerizza, dellavalle, turati}@cefriel.it

**Abstract.** In this paper we describe the improvements on our approach to the old discovery scenario of the SWS Challenge and our current solution for the new discovery and composition scenario. The mediation scenario is not included in this paper since there were no changes to it from the last edition of the SWS Challenge workshop in Athens. The solution proposed is based on the WebML design methodology and the Glue WSMO discovery engine.

## 1 Introduction

The solution proposed in this paper capitalizes on the experience we gained during the three workshops of the Semantic Web Service challenge 2006 [1] where we proposed one of the most complete solutions. In [2,3,4] we have presented a rather complete and convincing concept of a design support environment for Semantic Web applications, but our software solution requires a lot of improvements before encompassing all the needs which are set in the challenge. In this paper, we discuss how we intend to further extend our solution in preparation for the challenge's next "round", which is set for June 2007.

The challenge presents three scenarios: the *Mediation Scenario* integrates a legacy system with a system supporting the RosettaNet protocol, the *Discovery Scenario* performs order processing by automatically discovering a new supplier together with a shipper and a new *Discovery Scenario with Composition* that requires to select the best matching offers from different computer hardware providers and compose the best package according to a user buy goal. The mediation scenario is not changed since the last edition of the workshop and hence the complete and effective solution proposed in [4] was not changed and is not discussed in this paper.

Our solution, like in the previous workshops, is based on an original mix of Semantic Web and Software Engineering techniques: WSMO [5] as Semantic Web Service approach, Glue [6, 7] as Semantic Web Service Discovery engine, WebML [8, 9] as Web engineering model for designing and developing semantically rich Web applications implementing Service Oriented Architecture, and WebRatio [10] as WebML CASE tool. These background technologies are reviewed in Section 2.

Section 3 shows the general features of our solution for the discovery and invocation problems; then, Section 4 illustrates some improvements to the original discovery scenario and the in Section 5 we present how we address the new discovery and composition scenario. Finally Section 6 presents our long term vision and gives hints about our future research directions.

## 2 Background

**WebML/WebRatio WS edition.** WebML is a high-level notation for data- and process- centric Web applications. It allows specifying the conceptual modeling of Web applications built on top of a data schema used to describe the application data, and composed of one or more hypertexts used to publish the underlying data.

The WebML allows one to specify a data model describing the domain data structure as an Entity-Relationship (E-R) or, equivalently, a UML class diagram. Upon the same data model, it is possible to define different hypertexts (e.g., for different types of users or for different publishing devices), called *site views*. A site view is a graph of *pages*, allowing users from the corresponding group to perform their specific activities. Pages consist of connected *units*, representing at a conceptual level atomic pieces of homogeneous information to be published: the content that a unit displays is extracted from an entity, and selected by means of a *selector*, testing complex logical conditions over the unit's entity. Units within a Web site are often related to each other thru *links* carrying data from a unit to another, to allow the computation of the hypertext. WebML allows specifying also update *operations* on the underlying data (e.g., the creation, modification and deletion of instances of an entity, or the creation and deletion of instances of a relationship) or operations performing other actions (e.g. send an e-mail). To describe Web services interactions, WebML has been extended with Web service units, corresponding to the WSDL classes of Web service operations. *Request-response* and *response* operations are triggered when the user navigates one of their input links; from the context transferred by these links, a message is composed, and then sent to a remote service as a request. *Solicit* and *one-way* are instead triggered by the reception of a message. Indeed, these units represent the publishing of a Web service, which is exposed and can be invoked by third party applications. In the case of one-way, the WebML specification may dictate the way in which the response is built and sent to the invoker.

The language is *extensible*, allowing for the definition of customized operations and units. It has been implemented in the CASE tool WebRatio, a development environment for the visual specification of Web applications and the automatic generation of code for the J2EE platform. The design environment is equipped with a code generator that deploys the specified application and Web services.

**Glue WSMO discovery Engine** Glue is a WSMO compliant discovery engine that provides the basis for introducing discovery in a variety of applications that are easy to use for requesters, and that provides efficient pre-filtering of relevant services and accurate discovery of services that fulfill a given requester goal.

In conceiving Glue, the model for WSMO Web Service discovery was refined by making the role of mediation more explicit, by (i) using the notions of class of goals and class of Web Service descriptions; (ii) using ggMediators for automatically generating a set of goals semantically equivalent to the one expressed by the requester but expressed with a different form or using different ontologies (iii) making wgMediators the conceptual element responsible for evaluating the matching; (iv) using ooMediators for solving any terminological mismatch that can appear with different polarized ontologies for the domains; and (v) by redefining the discovery mechanism as a composite procedure where the discovery of the appropriate mediators and the discovery of the appropriate services is combined. Moreover in designing Glue the authors refined WSMX Discovery Engine architecture according to their refined WSMO discovery conceptual model both in terms of components and execution semantics.

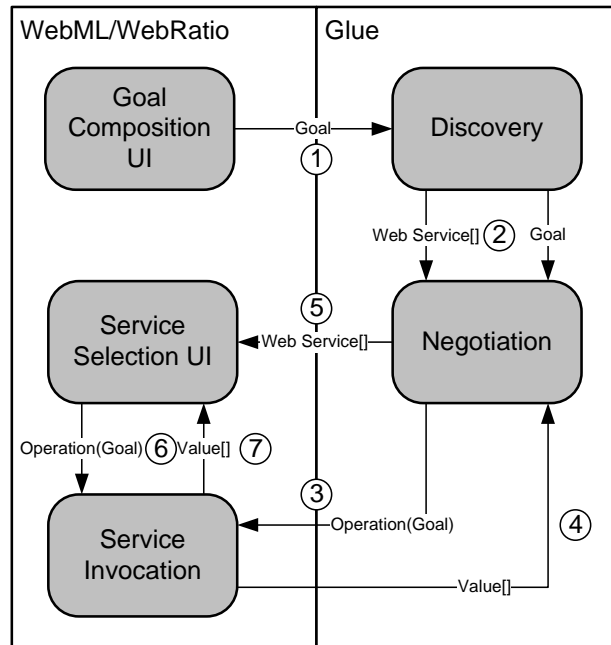
The Communication Manager exposes Glue functionalities to other services, by offering plain Web Services for publishing semantic Web Service Descriptions (WSD), submitting the goal for searching the repository for WSDs previously published and, finally, getting the results of the matching. The Constructors are responsible for translating the SOAP messages and goals. The Goal Translator translates the user goals into goals more close to the provider perspective. Finally, the Proof Generator gathers all the necessary information from the other components, invoking the internal reasoner and providing the discovered results to the client.

The Glue implementation uses internally F-logic and it is built around an open source F-logic inference engine called Flora-2 that runs over XSB, an open source implementation of tabled-prolog and deductive database system.

### **3 Summary of the overall approach to discovery**

In our general solution to the discovery scenarios, Glue and WebML shares the model of the Web Services and of the Goals to allow an integration of the two systems. In particular WebML/WebRatio is adopted to model and deploy a proper graphical interface to specify goals to be sent to the discovery, the same graphical interface is adopted to allow the final service selection by the user (even if it's possible to automatically select the best ranked service, we believe that in the SWS challenge scenarios the user should make the final choice). WebML/WebRatio is also in charge to provide the SOA layer to actually invoke the final service or to negotiate with it during the discovery. Glue is the core of the Discovery process: when a goal is sent, according to the available set of services and wgMediators, Glue select and rank the best matching services and if needed to filter in a better way the initial ranking, delegates to WebML/WebRatio the negotiation with the services. The interaction between WebML and Glue is showed in Figure 1.

Glue was extended for the third workshop so as to be able to invoke any external service in order to gather additional information on the service.

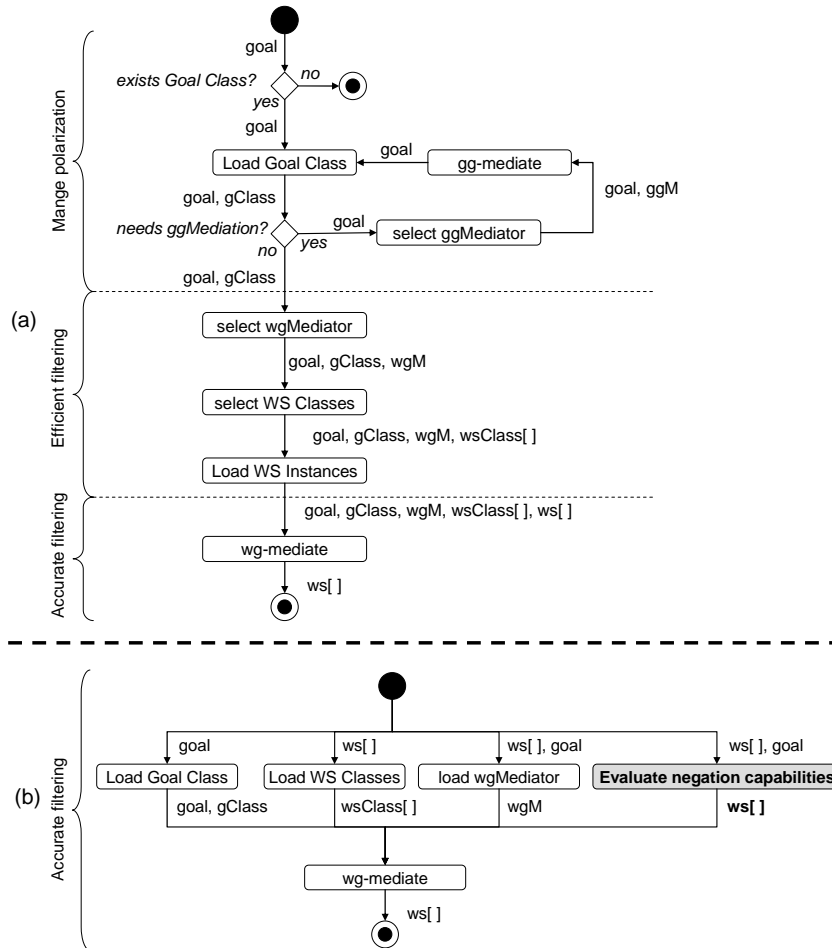


**Fig. 1.** The interfaces of current Glue implementation in terms of interfaces, components and interaction among them.

At conceptual level both Goal and Web Services are modelled making explicit differences among:

- “discovery” capabilities, which are static description of the service in terms of functional properties,
- “negotiation” capabilities, which are description of the service that need to be evaluated by invoking one or more operation of the service (for describing the choreography we rely on WebML), and
- “selection” capabilities, which are static or dynamic non functional descriptions.

Part of the Goal and Web Service model are shared between Glue and WebML to allow WebML to help in the Goal specification and to perform the actual invocation.



**Fig. 2.** The execution semantics of both the Discovery (a) and the Negotiation component (b) of Glue.

Figure 2 presents the execution semantics of Glue that is based on two core components: the Discovery and the Negotiation component. The execution semantics of the Discovery component is the same of the one presented at phase-II. The execution semantics of the Negotiation component is derived from the discovery by extending it with an activity for *evaluating negotiation capabilities* and by reusing activities that load WSMO element and evaluate the *wgMediator*.

## 4 Improvements to the Discovery Scenario

During the modeling of the ontologies concerning date-time, location, products and shipments, the development was kept to the minimum necessary for the scenarios. In the first version of the shipment discovery scenario, the example explaining the role of the delivery time included some references to different time zones. Even if neither the service instances nor the goal instances require the time zone to be expressed, we introduced them in order to become closer to a real scenario.

In order to deal with time zones, first of all we have modified the date-time ontology. In particular, we have added a reference to a time zone to the concept of `dateTime`, so that an instance of `dateTime` is now characterized by a date, a time and also a time zone. An example is `03/05/2007 14:58 GMT+1`. A time zone has a code and an offset. The former represents a standard time adopted by a specific region of the earth while the latter represents the difference (number of hours) between the local time and the standard time expressed by the code. For the previous example, the code is `GMT` and the offset is `+1`. The following code shows the piece of the date-time ontology in which we have introduced the modifications just discussed.

The time zones into the date-time ontology	
<pre>dateTime:instant[   date-&gt;date,   time-&gt;time,   timeZone-&gt;timeZone ].</pre>	<pre>timeZone[   zone-&gt;timeZoneAcronym,   offset-&gt;integer ].</pre>

By convention every time zone is far a definite number of hours from Coordinated Universal Time (UTC). For example, Greenwich Mean Time (GMT) corresponds to UTC, Central European Time (CET) is equivalent to UTC+1 and Pacific Standard Time (PST) is equivalent to UTC-8. In order to allow the comparison between two different `dateTime`, the difference between two time zones has to be considered. For this reason, after defining a set of time zone codes (GTM, UTC, CET, PST...), we have made explicit the distance between every time zone code and the Coordinated Universal Time (UTC), as shown in the following table.

Time zones and differences among them	
<pre>gmt:timeZoneAcronym. utc:timeZoneAcronym. wet:timeZoneAcronym. cet:timeZoneAcronym. eet:timeZoneAcronym. est:timeZoneAcronym. edt:timeZoneAcronym. cst:timeZoneAcronym. cdt:timeZoneAcronym. mst:timeZoneAcronym. mdt:timeZoneAcronym. pst:timeZoneAcronym. pdt:timeZoneAcronym.</pre>	<pre>hoursDifferenceFromUTC(utc, 0). hoursDifferenceFromUTC(gmt, 0). hoursDifferenceFromUTC(wet, 0). hoursDifferenceFromUTC(cet, 1). hoursDifferenceFromUTC(eet, 2). hoursDifferenceFromUTC(est, -5). hoursDifferenceFromUTC(edt, -4). hoursDifferenceFromUTC(cst, -6). hoursDifferenceFromUTC(cdt, -5). hoursDifferenceFromUTC(mst, -7). hoursDifferenceFromUTC(mdt, -6). hoursDifferenceFromUTC(pst, -8). hoursDifferenceFromUTC(pdt, -7).</pre>

Besides that, it has been necessary to modify all the rules involving a `dateTime` element in order to consider also the time zones. For example, the rule that calculates how many days are between two `dateTime` instances has been modified taking into account that the difference between the corresponding `timeZones` can influence the

final number of days. Obviously, a new rule for calculating the difference (number of hours) between two timeZones has been introduced: it takes into account the hours between two zones as well as the offsets.

Difference between two dateTime instances
<pre>//the difference in days between two different DateAndTime daysBetween(D1, D2, X) :-     D1:dateTime, D2:dateTime,     daysAfterChrist(D1.date,DAC_D1),     daysAfterChrist(D2.date,DAC_D2),     hoursBetweenTimeZones(D1.timeZone,D2.timeZone,HBTZ),     (         (D1.time.hourOfDay + HBTZ &gt;= 24 , X is DAC_D1 - DAC_D2 + 1)         ;         (D1.time.hourOfDay + HBTZ &lt; 24 , D1.time.hourOfDay + HBTZ &gt;= 0, X is DAC_D1 - DAC_D2)         ;         (D1.time.hourOfDay + HBTZ &lt; 0 , X is DAC_D1 - DAC_D2 - 1)     ).  hoursBetweenTimeZones(X,Y,H) :-     X:timeZone, Y:timeZone,     hoursDifferenceFromUTC(X.zone, HX),     hoursDifferenceFromUTC(Y.zone, HY),     H is (HX - HY) + (X.offset - Y.offset).</pre>

At the end, we have modified the goal instances by introducing a timeZone element in every dateTime. With these modifications, a final user can express all the dateTimes in the preferred local time. For example, now a user that is in England can request a shipment from a location in Italy to a location in the USA, by expressing all the dateTime in the local time zones of the involved location. In the formal goal instance corresponding to this example, the timeZone of the currentDateTime would be equal to UTC (the England time zone), the timeZones of the requestedPickup-DateTimeInterval would be set to the Italian time zone (GMT+1) while the timeZone of the requestedDeliveryDateTime could be set to CDT-2.

## 5 Addressing the new Discovery with Composition Scenario

The new scenario involves several products with different characteristics. We modeled this situation by defining new concepts (and the corresponding properties) into the products ontology.

In particular, a general product is characterized by a global identifier (GTIN) and a price. A notebook is a special product having a monitor, a processor and many other features. Besides all these concepts that represent classes belonging to the ontology, there are the instances, which represent the real notebooks that users can buy.

The concepts defined in the product ontology have been used both in the web service descriptions and goals. First of all, we defined a new class of web service that represent the sellers and basically states that a service sells a set of products. The instances of this class can list the real products with their characteristics. Furthermore, we defined a new class for the goal of purchase, which basically states that a user asks for a product with particular features.

Glue is based on the mediator centric methodology. Mediators assume a central role in the discovery process, especially the wgMediators. In order to deal with the

new scenario, we introduced new rules in the wgMediator responsible for picking out the web services that are able to sell the product required in a goal. For example, one of these rules is responsible for checking whether a web service offers at least one product of which features satisfy the preferences that have been expressed in the goal.

With the simple modifications described above, which are always necessary whenever a new domain is introduced, we are able to directly solve Goal A1. With our approach it was easy to solve Goal A2 too. We noticed that Goal A2 can be rephrased in two different goals, which can be executed separately and the final result is made up of the two different result sets. For this reason, we introduced another goal class that can be viewed as a “composite” goal because it connects two “simple” purchase goal classes.

In the new scenario, a user can express preferences about, for example, the price or some features of the product he wants to buy. For example, he can state the maximum price that he can pay, the minimum amount of RAM, and so on.

Goal B1 involves a particular kind of preference: “the price is most important to me, get me the cheapest offer if the other requirements are met”. This statement requires that we are able to sort the results in ascending order with respect to the price and then take the first element. The sorting preferences are applied in WebML according to the user selection.

## 6 Conclusions

In this paper we present how we address the last changes to the Discovery Scenario and the new Discovery with Composition Scenario. While we almost completely cover the old scenario, the new one has still many point that we cannot address with our actual technology since they requires some changes in the discovery process. In future editions we plan to explore also these new requirements so as to achieve a better coverage of the Discovery with Composition Scenario and other possible scenarios.

## 7 References

1. Semantic Web Service Challenge: <http://www.sws-challenge.org>, 2006.
2. Brambilla, M., Ceri, S., Cerizza, D., Della Valle, E., Facca, F. M., Fraternali, P., Tziviskou, C.: Web Modeling-based Approach to Automating Web Services Mediation, Choreography and Discovery. In Semantic Web Service Challenge 2006 Phase I, Stanford University, Palo Alto, CA, March 2006. Available at: [http://swschallenge.org/wiki/index.php/Workshop\\_Stanford](http://swschallenge.org/wiki/index.php/Workshop_Stanford).
3. Brambilla, M., Ceri, S., Cerizza, D., Della Valle, E., Facca, F. M., Fraternali, P., Tziviskou, C.: Web Modeling-based Coping with Requirements Changes: SWS-challenge phase II. In Semantic Web Service Challenge 2006 Phase II, Budva, Montenegro, June 2006. Available at: [http://sws-challenge.org/wiki/index.php/Workshop\\_Budva](http://sws-challenge.org/wiki/index.php/Workshop_Budva).

4. Brambilla, M., Ceri, S., Cerizza, D., Della Valle, E., Facca, F. M., Fraternali, P., Tziviskou, C.: Improvements and Future Perspectives on Web Engineering Methods for Automating Web Services Mediation, Choreography and Discovery. In Semantic Web Service Challenge 2006 Phase III, Athens, GA, November 2006. Available at: [http://sws-challenge.org/wiki/index.php/Workshop\\_Athens](http://sws-challenge.org/wiki/index.php/Workshop_Athens).
5. Roman, D., Keller, U., Lausen, H., de Bruijn, J., Lara, R., Stollberg, M., Polleres, A., Feier, C., Bussler, C., Fensel, D.: Web Service Modeling Ontology. In Applied Ontology, 1(1): 77–106, 2005.
6. Della Valle, E., Cerizza, D.: Cocoon glue: a prototype of WSMO discovery engine for the healthcare field. In Proceedings of 2nd WSMO Implementation Workshop WIW'2005, 2005.
7. Della Valle, E., Cerizza, D.: The mediators centric approach to automatic web service discovery of Glue. In Hepp, M., Polleres, A., van Harmelen, F., Genesereth, M.R., editors, Proceedings of the First International Workshop on Mediation in Semantic Web Services (MEDIATE 2005), Amsterdam, The Netherlands, December 12, 2005, CEUR-Workshop Proceedings, 8, 35–50.
8. Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., Matera, M.: Designing Data-Intensive Web Applications. Morgan-Kaufmann, December 2002.
9. WebML.org: <http://www.webml.org>, 2006.
10. Webratio: <http://www.webratio.com>, 2006.
11. Brambilla, M., Celino, I., Ceri, S., Cerizza, D., Della Valle, E., Facca, F. M.: A Software Engineering Approach to Design and Development of Semantic Web Service Applications. In proceedings of 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA.
12. Popa, L., Hernández, M.A., Velegrakis, Y., Miller, R. J.: Mapping XML and Relational Schemas with CLIO. In System Demonstration, IEEE Data Engineering Conference, 2002.
13. Brambilla, M., Ceri, S., Comai, S., Tziviskou, C.: A Visual Data Mapping Tool for Software Components Interactions in Service-Oriented Architectures. In IASTED Conf. on Software Engineering 2006, 33–38.