An XML to WSML Adapter Implementation

Edward Kilgarriff, Brahmananda Sapkota, Laurentiu Vasiliu, David Aiken
Digital Enterprise Research Institute (DERI), National University Ireland, Galway
FirstName.LastName@deri.org

1. Introduction

This paper describes an implementation of an Adapter that converts XML to a Web Service Modeling Language (WSML) [1]. WSML is the language used to describe Web Service Modeling Ontology (WSMO) [2] concepts, related to Semantic Web services (SWS). SWS are web services that are semantically annotated. The semantic annotation is necessary to address various business logics in an appropriate manner, thus allowing complex business applications to be built and executed. The Web Service Execution Environment (WSMX) [3] is an execution environment for dynamic discovery, selection, mediation and invocation of semantic web services. WSMX is a reference implementation for WSMO.

2. The need for an XML to WSML adapter

When considering the scope of an adapter mediating two heterogeneous formats we must consider the process in which that adapter is involved. The process is initiated when a service requester (or originator) submits their goal (or desire) to WSMX in the form of an XML [4] message. The WSMX environment is responsible for matching the requester goal to the capabilities of web services registered to WSMX. WSMX then selects the most appropriate web service, mediates between the ontology's of service requester and provider, and finally, invokes the selected web service. However, WSMX can only understand its internal language WSML. Therefore, there is a need of some component to convert a requester's XML message to WSML message. We call this 'component' an Adapter.

An Adapter needs to be able to convert the XML message coming from the requester (for example the requester wishes to buy books), into a WSML message for the WSMX to be able to execute it. In this paper, we concentrate only on the message adaptation aspect. Other aspects, such as security (i.e., the message might have been digitally signed) are outside the scope of this paper.

3. Implementation

Adapters allow heterogeneous systems using their own message format to communicate with WSMX. The adapter that has been implemented transforms received XML files from a user into WSML files. After the Message Adapter receives a XML message, the XML content is validated then the document is parsed looking for specific constants or keywords (marked in CAPITAL italics, see Fig. 1) in order to decide what kind of XML file type has been received.

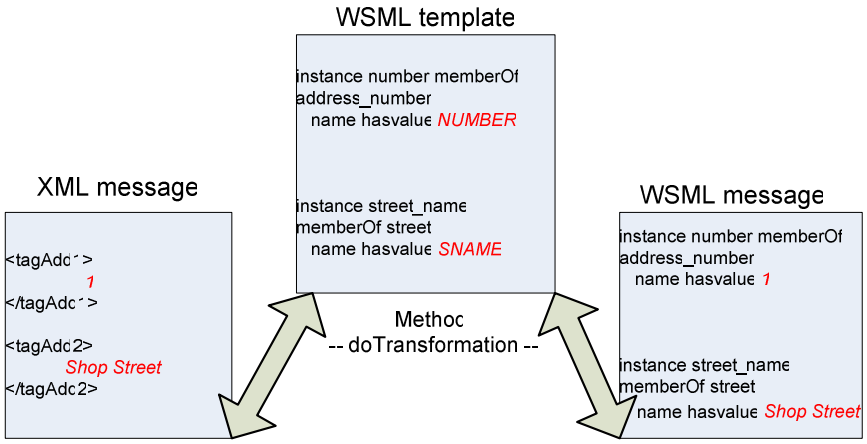

Fig.1. XML to WSML Transformation.

Once the received XML file type is identified, the XML is parsed and particular data extracted from it. This particular data provides the necessary parameters to call a method that implements the transformation from a class containing the transformation details. This transformation method performs the conversion from the XML file into the WSML file with the help of a WSML template file. The WSML template file is stored in the local database and when it is retrieved from the database, the transformation rules specific to the XML message are applied. The XML file type is identified by a number of 'if' statements which check the few home-grown XML files constructed especially for this implementation. How the adapter would handle 'any' type of XML message will be discussed in the future work.

The WSML template has several constants that are replaced with specific parameters extracted from the initially parsed XML files. Once the constants are replaced, the new created WSML file is sent from the Message Adapter to WSMX. WSMX will send the acknowledgement of receipt and an order number back to the originator application.

The technologies used in this implementation were; MySQL [5], a freeware database, Tomcat [6], a freeware Java Servlet, JSP [7] technologies used for implementing the GUI and a Java environment with an Apache-Ant [8] plug-in to publish the WSDL interfaces.

4. Conclusions and Future Work

The direction of work concerning this adapter should move towards the use of independent dynamic ontologies in order to provide a neutral message template from which a message in the target ontology language may be constructed. Decoupling ontologies from the WSML template aims to improve efficiency of the adaptor management process by using only one global template. This template may be used to create a message, tailored to the format of any particular ontology used in the adaptation process (i.e. from XML or another source language to WSML).

The aim of this design direction would be to create an adaptive resource to adjust the source information to meet our needs. The advantage of this is that instead of having 'N' message templates, there are 'N' independent, pluggable ontologies and one message template [9]. However, rather than mapping one message format to another, the conceptualised design allows for message decomposition and reconstruction in another format.

Work has already begun to facilitate the use of XSLT [10] technologies mapping all XML messages to one single XML format, then applying the mapping to the WSML template. This would reduce the need for separate WSML templates for different XML message formats. There will also be some effort in the direction of connecting the Message Adapter Framework into other WSMO compliant execution environments (e.g., IRS III [11]). Lastly, we will consider using other language adapters other than XML, for example, EDI [12], etc.

References

1. J. de Bruijn: WSML Specification (2004), available at: http://www.wsmo.org/2004/d16/
2. D. Roman, H. Lausen, U. Keller, E. Oren, C. Bussler, M. Kifer, D. Fensel: Web Service Modelling Ontology (2004), available at: http://www.wsmo.org/2004/d2/v1.0/
3. E. Cimpian, M. Moran, E. Oren, T. Vitvar, and M. Zaremba. Overview and Scope of WSMX. Technical report, WSMX Working Draft, http://www.wsmo.org/TR/d13/d13.0/v0.2/ , February 2005.
4. XML: http:// http://www.w3.org/XML/
5. MySQL: http://www.mysql.com/
6. Tomcat: http://java.sun.com/products/jsp/tomcat/
7. JSP: http://java.sun.com/products/jsp/
8. Apache-Ant : http://ant.apache.org/
9. H. Assal: Translation Methodology for Design Databases, UCLA, (1996). available at: http://www.calpoly.edu/~hassal/thesisintro.pdf
10. J. Clarke : XSL Transformation, W3C, (1999), available at: http://www.w3.org/TR/xslt
11. J. Domingue L. Cabral, F. Hakimpour, D. Sell, E. Motta: DEMO IRS III: A Platform and Infrastructure for Creating WSMO-based Semantic Web Services, available at: http://iswc2004.semanticweb.org/demos/45/paper.pdf
12. EDI: http://www.unece.org/trade/untdid/welcome.htm