From BPEL4WS Process Model to Full OWL-S Ontology

Muhammad Ahtisham Aslam
Betriebliche Informationssysteme
University of Leipzig
Germany
+49 (341) 97-32323
aslam@informatik.uni-leipzig.de

Sören Auer
Betriebliche Informationssysteme
University of Leipzig
Germany
+1 (215) 898-2716
auer@seas.upenn.edu

Jun Shen
School of IT and CS
University of Wollongong
Australia
+61-2-42213873
jshen@uow.edu.au

ABSTRACT

BPEL4WS is one of the most utilized business process development languages. It can be used to develop executable business processes as a combination of Web Services interactions in a specific sequence called process flow. But still BPEL4WS lacks sufficient representation of business process semantics required for business processes automation. On the other hand OWL-S (OWL for Web Services) is designed to present such kind of semantic information. There exists similarity in the conceptual model of OWL-S and BPEL4WS that can be used to overcome this lack of semantics in BEPL4WS by mapping the BPEL4WS process model to the OWL-S ontology. The mapped OWL-S service can be dynamically discovered, composed and invoked on the basis of matching semantics. Such a process of mapping syntax based Web Services composition in the form of BPEL process model to Semantic Web Services composition in the form of OWL-S composite service can also enable automation of BPEL processes as OWL-S services by applying AI planning techniques. In this paper we present a mapping strategy and a mapping tool that can be used to map BPEL processes to the OWL-S suite of ontologies.

Keywords

Web Services, Semantic Web Services, Business Process Execution Language for Web Services.

1. INTRODUCTION

Different workflow languages specially Business Process Execution Language for Web Services (BPEL4WS) [1] use Web Services in a more meaningful way by combining Web Services functionality in a specific sequence to perform a certain task. Even though BPEL has good process modeling capabilities, its semantic limitations are a hurdle in business process automation. OWL-S (OWL ontology for Web Services) [2], aims to make Web Services descriptions more computer-interpretable, thus enabling automation of a variety of tasks including Web Service discovery, invocation, and composition. Therefore, mapping a BPEL process to an OWL-S service can help to automate business processes on the basis of semantic information provided by the OWL-S ontology.

Our work (an improvement and extension to [3]) presents a mapping strategy and its prototypical implementation as mapping tool[1] (BPEL4WS2OWL-S) that can be used to map BEPL4WS processes to the complete OWL-S suite of ontologies. The whole mapping process uses the OWL-S API [4] on its backend for writing the OWL-S ontology for resulting OWL-S Service.

2. Mapping Specifications

BPEL has two kinds of activities, primitive activities and structured activities. BPEL primitive activities are mapped to the OWL-S *Perform* control construct to perform the relevant atomic process. Also, if a primitive activity is an input/output (I/O) activity (working as BPEL process interface) then this activity is used to create the *Profile* of the resulting OWL-S service. BPEL structured activities are mapped to relevant OWL-S control constructs as shown in figure 1.

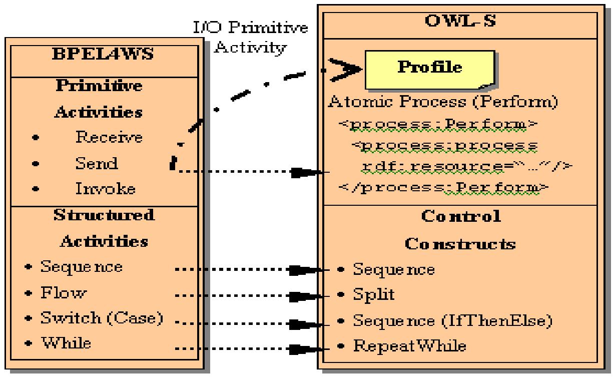


Fig.1. Overview of mapping specifications.

- **BPEL Process Mapping to OWL-S Process Model:** BPEL executable processes are mapped to OWL-S atomic and composite processes within the *Process Model* ontology. Also, to keep the mapping complexities within limitations synchronization between process components is not supported in this version.
- **Atomic Processes:** Operations supported by partner services (WSDL services) can be used to perform some specific task in a single step. Like an operation supported by a Web Service, an atomic process in OWL-S is a process that can perform some action in a single step. Therefore partner Web

Demos and Posters of the 3rd European Semantic Web Conference (ESWC 2006), Budva, Montenegro, 11th – 14th June, 2006.

[1] http://bpel4ws2owls.sourceforge.net/

Services (WSDL Services) are parsed and an atomic process (with *Profile, Process Model and Grounding*) is created for each supported operation.

- **Primitive Activities and Atomic Processes:** The *Perform* control construct in OWL-S is used to perform an atomic process, while BPEL has primitive activities (e.g. *Receive, Invoke and Reply*) that can be used to perform some specific operation in a single step. Therefore, BPEL primitive activities are mapped to OWL-S *Perform* control construct to perform the relevant atomic process.

- **Structured Activities:** BPEL structured activities are mapped to OWL-S control constructs within an OWL-S composite process. Fig.1 shows the mapping of BPEL structured activities to OWL-S control constructs.

- **Data Flow:** Mapping of assignment activity that is used between two primitive activities, results in the creation of data flow between the corresponding atomic processes.

- **Profile:** A BPEL process can have one or more primitive activities, which act as an interface to communicate with the BPEL process. Therefore, among these primitive activities options, the input message of the first *Receive* primitive activity receiving a message from the outer world is defined as an input for the OWL-S composite process. If a *Receive* activity has corresponding *Reply* activity then the message variable of this *Reply* activity is used to set the output of the OWL-S composite process. If a *Receive* activity has no corresponding *Reply* activity then, the first primitive activity (e.g. the first *Invoke* activity sending some message to outer world) is taken as an output activity to define the output of the OWL-S composite process. Also, a primitive activity is declared as an Input/Output (I/O) activity if the BPEL's corresponding WSDL file supports its port type and operation. These input and output messages are used to create the *Profile* of the resulting OWL-S service. This *Profile* is used to present the semantically enriched service capabilities by annotating input and output parameters of *Profile* with ontological concepts.

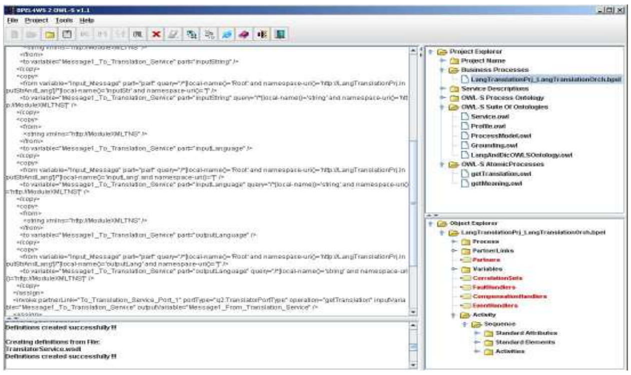
- **Grounding:** The grounding of the mapped OWL-S service specifies the location of the grounding of each atomic process (created during the mapping as discussed above). Of course, the mapping is not able to define the XSL transformation for complex messages. Web Services Description Language (WSDL) service, being an XML format for describing network services is described in the grounding of each atomic process to have access to the original implementation of the WSDL service.

3. User Interface

The BPEL4WS2OWL-S mapping tool provides an easy to use interface (fig.2) employing menus and buttons to perform the mapping process. The mapping process includes creating a new project, adding input BPEL and WSDL files, validating the input files, building the project (to create object view of input files) and finally mapping the project. The resulting OWL-S ontology files can be viewed in the project explorer (upper right window) and the contents of these files can be seen in upper left window of the tool. The lower left window acts as an output window to see the output of different mapping actions. The lower right window is an object explorer, which gives an object view of the input files.

4. Conclusion and Future Work

OWL-S is not as mature as BPEL. For example, equivalents of BPEL activities like *Assignment*, *Fault Handler*, *terminate* etc. are not available in OWL-S for direct mapping from BPEL to OWL-S. Also, users need to annotate the *Profile* of the resulting OWL-S service with their domain ontologies. Therefore, manual changes are required in the areas where the mapping is only partially supported or information needs to be added by the user. Such manual changes are a time consuming and complex task and require a user to be an expert of OWL-S. So at this stage our BPEL4WS2OWL-S tool needs constant updates with the upcoming versions of the related technologies. Furthermore, a tool is needed that can be used to develop domain ontologies and an editor which helps in editing the resulting OWL-S ontology with these domain ontologies more easily, ideally in a visual environment. Protégé with its plugin, OWL-S Editor, is an ideal basis to achieve this goal. Hence, we are working to improve our tool, and to make it available as a BPEL4WS2OWL-S import plug-in for Protégé with OWL-S Editor, so that the mapped OWL-S services can be directly imported in the OWL-S Editor tab and can thus be edited in a visual environment.



```
        <string xmlns="http://ModuleXMLTNS" />
      </from>
      <to variable="Message1_To_Translation_Service" part="inputString" />
    </copy>
    <copy>
      <from variable="Input_Message" part="part" query="/*[local-name()='Root' and namespace-uri()='http://LangTranslationPrj.in
putStrAndLang']/*[local-name()='inputStr' and namespace-uri()='']" />
      <to variable="Message1_To_Translation_Service" part="inputString" query="/*[local-name()='string' and namespace-uri()='htt
p://ModuleXMLTNS']" />
    </copy>
    <copy>
      <from>
        <string xmlns="http://ModuleXMLTNS" />
      </from>
      <to variable="Message1_To_Translation_Service" part="inputLanguage" />
    </copy>
    <copy>
      <from variable="Input_Message" part="part" query="/*[local-name()='Root' and namespace-uri()='http://LangTranslationPrj.in
putStrAndLang']/*[local-name()='inputLang' and namespace-uri()='']" />
      <to variable="Message1_To_Translation_Service" part="inputLanguage" query="/*[local-name()='string' and namespace-uri()
='http://ModuleXMLTNS']" />
    </copy>
    <copy>
      <from>
        <string xmlns="http://ModuleXMLTNS" />
      </from>
      <to variable="Message1_To_Translation_Service" part="outputLanguage" />
    </copy>
    <copy>
      <from variable="Input_Message" part="part" query="/*[local-name()='Root' and namespace-uri()='http://LangTranslationPrj.in
putStrAndLang']/*[local-name()='outputLang' and namespace-uri()='']" />
      <to variable="Message1_To_Translation_Service" part="outputLanguage" query="/*[local-name()='string' and namespace-uri
()='http://ModuleXMLTNS']" />
    </copy>
  </assign>
  <invoke partnerLink="To_Translation_Service_Port_1" portType="q2:TranslatorPortType" operation="getTranslation" inputVaria
ble="Message1_To_Translation_Service" outputVariable="Message1_From_Translation_Service" />
  <assign>
```

```
Definitions created successfully !!

Creating definitions from File:
TranslatorService.wsdl
Definitions created successfully !!
```

Fig.2. Overview of BPEL4WS2OWL-S mapping tool interface.

5. REFERNECES

[1] Business Process Execution Language for Web Services Version 1.1. 5th May 2003. [online] Available ftp://www6.software.ibm.com/software/developer/library/ws-bpel.pdf.

[2] OWL-S: Semantic Markup for Web Services. [online] Available http://www.daml.org/services/owl-s/1.1/overview/.

[3] J. Shen, Y. Yang, C. Zhu and C. Wan. "From BPEL4WS to OWL-S: Integrating E-Business Process Descriptions", *Proc. of 2nd IEEE International Conference on Services Computing (SCC 2005)*, pp.181-188, Orlando, USA, July 2005.

[4] OWL-S API : [online] Available http://www.mindswap.org/2004/owl-s/api/