Mapping a business process model to a semantic web service model

Li Guo, Yun-Heh Chen-Burger and Dave Roberston
CISA, Informatics, The university of
Edinburgh, United Kingdom
L.Guo@sms.ed.ac.uk, {Jessicac,dr}@inf.ed.ac.uk

1. INTRODUCTION

As business grows more complex, it is difficult to find specific web services to meet our business requirements. The situation becomes even more complicated when there is no single web service which could satisfy all of the requirements but only a combination of several web services can. In order to tackle such problems, various standards for coordinating the web services have been developed by different organisations. Such examples are BPEL4WS [1], WS-Coordination [2], WS-CAF [3] and OWL-S [4]. As OWL-S is growing into the de facto standard, it is the target of our paper.

The purpose of OWL-S is to support effective automation of various web services related activities including service discovery, composition, execution and monitoring. For our work, we are interested in automated web services composition for a business process model. OWL-S process ontology provides a standard language for describing the composition of web services. Thus we can treat composed web services as a process model. However, since OWL-S is an XML based syntax description which is used for automatic agent processing, it is difficult to read and understand for human reader, especially for those with little XML knowledge but from a process modelling background, for example, businessmen. Therefore it can not be easily and directly used for process modelling tasks.

The Fundamental Business Process Modelling Language[5] is a diagram based business process modelling language which merges IDEF3[7], RAD[8] and PSL[9]. It has a precise operational semantics and describes business processes in a conventional logic. Because it is diagram based, it can be easily accepted by processes modelers who normally are familiar with such diagrams. However, its problem is that it does not provide a direct specification for web services based agents. In addition, the business process models are used to describe the requirements of the enterprises which are more stable and seldom change. Thus once the business requirements of an enterprises are determined and described by a business process modelling language, it is totally not necessary to re-write them in another one only because of the new technical features. So a "gap" exists between existed diagram based business process modelling effort and web service based agent development. The aim of this paper is to bridge the gap via conceptual mapping between two the different process modelling languages, FBPML and OWL-S.

2. MAPPING THE FBPML MODEL TO THE OWL-S PROCESS MODEL

From the introduction in section 1, we can see there is no obvious one-to-one mapping between individual terms in FBPML and OWL-S process models, because OWL-S uses different methods to represent process decomposition. Therefore, when we compare the two methods, we treat the whole FBPML process model as a specification that may be decomposed into the corresponding OWL-S process models with some decomposition decisions depending on other parts of the model. Nevertheless, we start our mapping by comparing common components.

We know that common process components of FBPML activity are: "Precondition", "Trigger", "Action" and "Postcondition". In an OWL-S process model, the basic properties of process are: "Input", "Output", "Precondition" and "Effect".The OWL-S ontology "Precondition" corresponds to the combination of FBPML notations "Precondition" and "Trigger". "Trigger" is a specialised pre-condition that realises important (business) policies and strategies. Since OWL-S doesn't has the concept of "Trigger", we have to combine "Precondition" and "Trigger" together in FBPML as a "Precondition" for an OWL-S process. An "Action" describes what operations a activity might do when it executes in FBPML. It provides a script for the implementation of a process model. However, OWL-S does not specify how a process model may be implemented. Instead, it specifies the "Input", "Output" and "Effect" of a process.

An ”Effect” defined in an OWL-S specification is a placeholder for future work, it is intended to indicate the effects caused by the execution of a OWL-S process. Therefore, it can’t be determined if ”Effect” is equivalent to ”Postcondition”, as a FBPML ”Postcondition” represents not only the effects caused by the execution of an activity but also those conditions that must hold true after execution of an activity. Input and output parameters of a FBPML activity are described using FBPML-DL [6] that provides data constructs used in a FBPML activity. Although, there is not a separate place holder to store those parameters, they may be extracted from the ”Precondition”, ”Trigger”, ”Action” and ”Postcondition” of a FBPML activity.

The following sections show how a particular FBPML model may be mapped to a OWL-S process model. From the notation definition, we can treat FBPML primary activity as atomic process in OWL-S Process Model, since they are both undecomposable and both represent single step execution. After translation, the XML based syntax for an OWL-S atomic process is:

```
<process:AtomicProcess rdf:ID=”A”>
  <process:hasInput>
    <process:Input rdf:ID=”...”>
      <process:parameterType rdf:resource=”...”/>
    </process:Input>
  </process:hasInput>
  <process:hasOutput>
    <process:Output rdf:ID=”...”>
      <process:parameterType rdf:ID=”...”/>
    </process:Output>
  </process:hasOutput>
</process:AtomicProcess>
```

The prefix ”process” in the above XML code is shorthand for the namespace ”http://www.daml.org/services/owl-s/1.0/Process.owl”. The properties ”input” and ”output” are extracted from the ”Precondition” , ”Trigger” and ”Postcondition” of FBPML primary activity ”A”. Since for all the activities in FBPML model, the translated OWL-S processes are almost same except the control constructs used, we will not re-write the XML based syntax in the following translations.

We can use the control construct ”Sequence” from an OWL-S process model to compose several activities connected by ”Precedence Links” in a FBPML model. ”And_Split-And_Joint combination” from FBPML describes a set of concurrent processes and has direct correspondence to the control construct ”Split” in the OWL-S process model. Combination of ”AND_Split” and ”OR_Joint” means that the activities between the two junctions must all start after the process ahead of ”AND_Split” finishes and if one of these activities finishes, the process after ”Or_Joint” can start. In OWL-S control constructs, ”Split+Join” is used for this purpose. An ”OR-OR” junctions combination allows very flexible execution order. ”Choice” control construct from OWL-S also does not specify the execution order and number of its components also. An ”OR-AND” combination says that any number of activities between the two junctions can start after the process before ”OR_Split” finishes, but all must be finished for the process after the ”OR_Joint” to start. The ”Unordered” control construct in the OWL-S process model is used for the same purpose. It is very common for us to see loops in a business process model. For models drawn in FBPML, a loop starts with a single activity and ends with an ”OR_Split”. The OWL-S process ontology also has mechanisms to control loops, which are ”Iteration” and ”Repeat-While”. ”Repeat-While” seems to be more commonly used. In a FBPML model, the activity that is next to ”OR_Split” and leads to the starting activity of the loop can be translated as the ”WhileProcess” of ”Repeat-While” control construct. If one of the branches of the ” OR_Split” leads to starting activity of the loop directly then we use the ”WhileCondition” in ”Repeat-While”. For the loop above with ”WhileCondition”, we only have to change the property ”process:WhileProcess” to ”process:WhileCondition” and the value of ”rdf:resource” to pre-defined ”WhileCondition”.

In order to show how composition and decomposition work with FBPML and OWL-S process model, we supply an example model in Figure 1 drawn in FBPML style. For simplicity, a cut-down FBPML notation is used, in which details that are not essential are omitted.

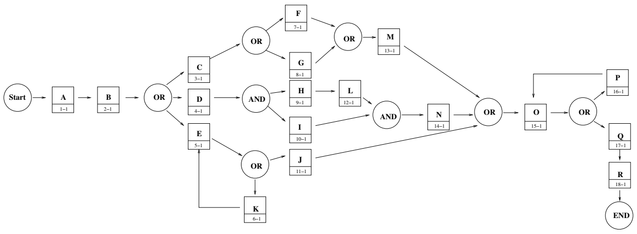

Figure 1. A complicated FBPML example model

Based on the initial mapping provided, we may decompose the example model step by step. The example model starts with a ”start” and ends with an ”end”, so it can be translated into a sequence process in OWL-S, all the activities between ”start” and ”end” are composite components of the sequence process. The out-most ”OR-OR” combination and the loop starting with ”O” are treated as 2 single composite activities with new names ”C1” and ”L1”. After getting the highest level composite process, we need to decompose the sub-processes in it, i.e ”C1” and ”L1”. The way we decompose each sub-process is exactly same as the way we use to create the sequence process, although different control constructs are used appropriately. The results of model decomposition that can be expressed by OWL-S directly are shown in figure 2 below:

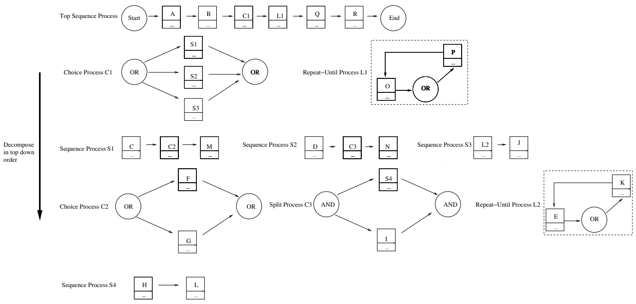

Figure 2. Decomposed FBPML models

A system has been developed for the automatic translation. We will not discuss that because of the limited space of the paper.

3. MAPPING PROBLEMS

3.1. COMPLEX CONTROL PROBLEMS

In the previous sections, we have shown a conceptual mapping from FBPML to OWL-S process models. We have implemented a translator to support the automatic translation based on our mapping algorithm. However, all the work we have done is based on models which can be hierarchically decomposed. The model shown in Figure 1 is a typical example. Many models do not decompose hierarchically and these cause mapping problems.

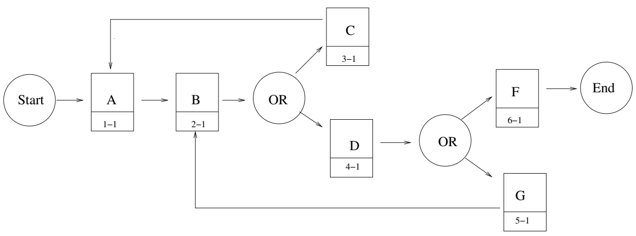

Figure 3. FBPML model with partly overlapped processes

When applying the decomposition, we find 2 loops in the above example, namely loop1 that includes activities ”A”, ”B” and ”C”as components and loop2 that is composed of ”B”, ”D” and ”G”. Each of them corresponds to a ”Repeat-While” control construct in the OWL-S process ontology. However, when we try to translate a model that includes these two ”Repeat-While” processes, we encounter a problem. What is the relationship between them? At a first glance, loop1 is ahead of loop2, so perhaps they may be translated into a sequence process? If we do this, the content of the sequence process will be (”A”,”B”,”C”) , (”B”,”D”,”G”). We can see process ”B” appears separately in 2 different OWL-S sub-processes which is not semantically equivalent to the original FBPML model unless extra control is added to the translated OWL-S model. Since the loops are only partially overlapped, none of the control constructs from the OWL-S process model can be used in this case. The reason causes this problem is that OWL-L process model uses programming-liked control constructs as its basic building blocks which are inadequate for all the modelling issues. This is not exclusively the OWL-S's problem, it is a common problem for BPEL4WS also.

To solve the problem, one propose is to re-draw the FBPML model to allow it to be decomposed in an OWL-S acceptable way without affecting its meaning. But this causes another problem: that is how could we guarantee that the re-drawn model is equivalent to the previous intended model? Even if we can do that, it obviously causes extra effort on modelling. Furthermore, there is also a operational problem after the model is separated into 2 sub-models: (”A”,”B”,”C”) and (”B”,”D”,”G”). If we have to choose one of the two models when running a service and keep with it, then we cannot really choose between ”C” and ”D” if the two loops are represented separately, since the two models cannot communicate with each other. This is a synchronisation problem caused by the OWL-S process model. OWL-S does specify how to control several processes' synchronisations within a single composite process, but says nothing about the interaction of several processes from different OWL-S process models. With the OWL-S process ontology, it is impossible to express such an interaction, since there is no concept of variables in OWL-S for information passing and sharing.

Another possible solution for this problem is to change the way that OWL-S is used to construct the model. Use logic message flow to construct the process model. For example, we can express sequence (A,B,C,D) by using link(A,B), link(B,C), link(C,D). This approach causes extra efforts at implementation phase since we have to identify the relations between activities solely by our own.

3.2. CONTEXT AWARENESS PROBLEM

In our conceptual mapping, we have nothing in the OWL-S process model corresponding to the notion of ”Role” as it is in FBPML. The notation ”Role” should not simply be understood as a text annotation used to indicate the performer of processes but in a intrinsic part of an organization and may be a part of an ontology that defines the processes. With ”Role”, we can group processes from different domains, and also a process can be shared by different roles. Furthermore, we can use appropriate processes to replace the unavailable processes in process model, if they are performed by the same role. It seems that we could also use alternatives in OWL-S by using SameClassOf to find synonyms and SubClassOf to identify homonyms. However, alternatives defined in OWL-S could not be used across different domains/service models easily without being known in advance. A further problem is, how we could know what kinds of classes are already available in an OWL-S application domains when we want to add a new class into such domains and want to define the same classes of it? With FBPML we could simply add the role to the new process to classify it. In addition, role, process and ontology can be used to help searching, process management, error detection, and model reuse. An example illustrating the interaction between enterprises is given below:

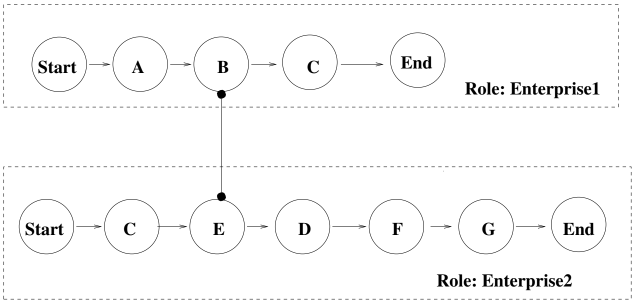

Figure 4. Interaction between different enterprises.

4. CONCLUSIONS AND FUTURE WORK

We have described our attempts to map the concepts of a business process modelling language, FBPML, to a semantic web services language,OWL-S . The mapping between languages is partial because of differences in control assumptions and process decompositions. There is also an issue of how to visualise OWL-S representations in the style used in FBPML. Since both business process and semantic web service languages continue to evolve, there is an opportunity to enhance both strands by strengthening the mapping in areas that we have highlighted. The contribution of such efforts is to support automated deployment on semantic webs of workflows that can be described in conventional business modelling terms.

Future work could address mapping problems identified in our paper, formalising FBPML model automatically and making our system more complete. We also will provide the ability of not only translating business process models to OWL-S process models, but also of generating service profiles automatically according to the OWL-S Process Models translated etc.

References

[1] Tony Andrews, Francisco Curbera, Hitesh Dholakia, Yaron Goland, Johannes Klein, Frank Leymann, Kevin Liu, Dieter Roller, Doug Smith, Satish Thatte, Ivana Trickovic, Sanjiva Weerawarana *Business Process Execution Language for Web Services Version 1.1* http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/

[2] Doug Bunting, Martin Chapman, Oisin Hurley, Mark Little, Jeff Mischkinsky, Eric Newscomer, Jim Webber, Keith Swenson *Web Services Coordination Framwork* http://www-106.ibm.com/developerworks/library/ws-coor/

[3] Doug Bunting, Martin Chapman, Oisin Hurley, Mark Little, Jeff Mischkinsky, Eric Newscomer, Jim Webber, Keith Swenson *Web Services Composite Application Framework.* http://developers.sun.com/techtopics/webservices /wscaf/

[4] David Martin, Mark Burstein, Grit Denker, Jerry Hobbs, Lalana Kagal, Ora Lassila, Drew McDermott, Sheila McIlraith, Massimo Paolucci, Bijan Parsia, Terry Payne, Marta Sabou, Evren Sirin, Monika Solanki, Naveen Srinivasan, Katia Sycara. *OWL-S: Semantic Markup for Web Services.* http://www.daml-s.org/owl-s/1.0/.

[5] Yun-Heh Chen-Burger, Austin Tate, and Dave Robertson, Enterprise Modelling: A Declarative Approach for FBPML, European Conference of Artificial Intelligence, Knowledge Management and Organisational Memories Workshop.

[6] Yun-Heh Chen-Burger, *Informal Semantics for the FBPML Data Language*, Informatics Report Series: EDI-INF-RR-0154, School of Informatics, the University of Edinburgh, October 2002.

[7] *Information Integration for Concurrent Engineering (IICE) IDEF3 Process Description Capture Method Report,* Knowledge Based Systems Inc. (KBSI), September 1995.

[8] Martyn A. Ould, *Business processes: Modelling and Analysis for Reengineering and Improvement,* John Wiley and Sons, 1995.

[9] Craig Schlenoff, Amy Knutila, and Steven Ray, ’Proceeding of the process specification language(psl) roundtable’, *NISTIR 6081, National Institute fo Standards and Technology, Gaithersburg,MD,*(1997). http://www.nist.gov/psl/.