

Reengineering Patterns for Ontologizing the Business Processes

Violeta Damjanovic
(Salzburg Research, Austria
violeta.damjanovic@salzburgresearch.at)

Abstract: This paper discusses how to resolve knowledge gap between the business process models and the ontological models as a way to semantic interoperability among both enterprise engineering and knowledge engineering. The complexity of such a problem is reflected in the range from improving semantic exchange of the knowledge between ontologies and business processes by involving process theory to designing the reengineering patterns that can help in articulating knowledge about the business processes. In this work we have additionally employed the DDPO (DOLCE Descriptions and Situations (D&S) Plan and Tasks Ontology) module of the DOLCE foundational ontology to serve as a formal framework for ontological description of business processes. As a result of reengineering different models into each other, two reengineering patterns that we have called WSDL-DDPO and BPEL-DDPO patterns have been noticed. The generic role of these patterns have been seen in facilitating ontological description of business processes and their possible (semi)automatic execution by the workflow engine.

Keywords: Ontology Design Patterns, Semantic Web, Business Processes, BPEL, WSDL

Categories: D.2.12, D.3.3, A.1, M.1, M.8,

1 Introduction

Nowadays, the enterprise engineering connects two fields: (a) *management of enterprises*, which encompass the knowledge and principles related to an enterprise (life cycles, business operations, activities, products), and (b) *software engineering*, which deals with the modelling and integration of business processes [Vernadat, 1996]. Up to now, a variety of the enterprise engineering methods have been used such as the following: Integrated DEFinition methodology (IDEF), Petri nets, Unified Enterprise Modelling Language (UEML), Enterprise Function Diagram (EFD). Each of these methods and methodologies only partially close the gap between the knowledge about processes associated with an enterprise, on the one hand, and reusable and (semi-)automatically executable business processes, on the other hand. As a promising solution in achieving interoperability between the real enterprise needs and the business process models, we already pointed out the importance of involving process theory in [Damjanovic, 2008].

This paper describes the focus of the ImportNET¹ project, which lies in the intersection of three mechatronic domains such as mechanical engineering, electrical

¹ The ImportNET project is co-funded by the European Commission within the Sixth Framework Programme under Contract 033610, in the area of ICT for Networked Businesses

engineering, and software engineering [Damjanovic et al., 2008]. For each of these engineering domains there exists ontological model and the exchange and usage of knowledge between them should be pragmatically interpreted. Here, we report on the results of our efforts in integrating diverse knowledge models, and at the same time - connecting the knowledge models and the business process models. We call such a design process - ontologizing the business processes and their encoded pragmatic structures - Reengineering Ontology Design Patterns (ReODP) for the business processes.

The approach taken by ImportNET project is to move from a cognitive model to a computational model [Damjanovic et al., 2008]. For this purpose, we have used the DOLCE foundational ontology (upper ontology), which “aims at capturing the ontological categories underlying natural language and human commonsense” [Pisanelli et al., 2002]. The DOLCE was originally developed in the EU WonderWeb project [Masolo et al., 2004] and extended in the METOKIS² project by adding an ontology module called D&S (Descriptions and Situations). The D&S module includes a representation language for tasks and processes [Gangemi et al., 2004], and it serves as a base for the DDPO (DOLCE D&S Plan and Tasks Ontology) ontology module, which we found to be necessary for modelling mechatronic tasks and processes. More specifically, we have used the DDPO module to help us getting a good understanding of business process models at the different levels of abstraction and to provide implicit rules for the facts that explain behaviour and structure of both executable and abstract business processes [Masolo et al., 2004][Damjanovic, 2008].

The introduction section gives a brief view of the research problem and questions tackled in this paper. In Section 2 related work in ontology design patterns is presented. Section 3 describes our main motivation for aligning business processes into the Semantic Web ontologies. We have explained the syntax and semantics of both the BPEL (Business Process Execution Language) and the WSDL (Web Service Description Language) that are used in designing business process models, as well as the DDPO module of the DOLCE foundational ontology that is used in formulating ontological models. In addition, an excerpt of a comparison analysis between the DDPO tasks and the BPEL activities is briefly reported. Section 4 describes the major research question addressed by the paper: how to ontologically support knowledge that describes business processes and allows (semi-)automatic execution by the workflow engine. Hence, we propose two ReODPs called the WSDL-DDPO and the BPEL-DDPO patterns for ontologizing the business processes. In Section 5, some brief conclusion remarks are drawn.

2 Related Work

The notion of *design patterns* is well-known software engineering paradigm that describes proven and documented solutions to a known modelling problem that repeatedly appears when designing different software systems [Aranguren, 2005]. Analogous to software engineering’ design patterns, Ontology Design Patterns (ODP) bring similar advantages to the ontology engineers.

² <http://metokis.salzburgresearch.at/>

The notion of ODP is introduced in 1999 for a particular problem domain in biology [Reich, 1999]. Then, semantic patterns as a language independent description of a certain concepts, relations, and/or axioms are described in [Staab et al., 2001], whereas in [Devedzic, 2002] main similarities between traditional design patterns in software engineering and ontologies are emphasized. The knowledge patterns as conceptual patterns that are “morphed” into a given knowledge base by a set of mapping axioms are represented in [Clark et al., 2003], whereas both [Svatek, 2004] and [Gangemi, 2005] are more focused on designing patterns for Semantic Web ontologies that are now called - ODP.

Today, there are two communities around ODP: the one located at the University of Manchester, covered by the Gene Ontology Next Generation (GONG) project³ in which ODPs are promoted to enable migrating current bio-ontologies to a richer and more rigorous level; another one is represented by the ODP community⁴, also known as *ontologydesignpatterns.org*, started under the NeOn project⁵ that investigates both development lifecycle and evolution lifecycle of networked ontologies. Both communities collect ODPs through online, public catalogs, support their evaluation, improvements and reuse.

3 Towards Semantic Business Processes based on ODP

In ImportNET project, the design and development of ontologies was based on DOLCE foundational ontology. More specifically, we employ the DDPO ontology module to be the main working paradigm [Damjanovic et al., 2008] and to provide semantically meaningful usage of knowledge that describes tasks, description and situation in which business processes would be applied [Damjanovic, 2008]. We suggest reading [Gangemi et al., 2004] and [Masolo et al., 2004] to gain deeper insight into the DDPO formal characterization.

Moreover, the main motivation of this research was to support the ontological description of the business processes and to facilitate (semi)automatic execution of ontology-supported business processes through the proposed ReODPs. As the process of creating an executable business process requires both the WSDL description of all business operations and the BPEL description of business processes that orchestrate Web services, we define ReODPs for both the WSDL and the BPEL.

The WSDL is an XML (eXtensible Markup Language) set of definitions for describing operations and messages that constitute a network endpoint [W3C WSDL, 2001]. Services (abstract endpoints) are defined using the following elements: *types*, *message*, *portType* (a set of abstract operations and messages), *binding* (e.g. SOAP binding), *port* (specifies an address for a binding), *service* (aggregates a set of related ports), *operation*.

The BPEL activities consist of the following [Baretto et al., 2007]:

-the BPEL activities, such as: *assign*, *compensate*, *empty*, *wait*, *throw*, *terminate*;

³ http://www.gong.manchester.ac.uk/?page_id=7

⁴ <http://ontologydesignpatterns.org/wiki/Odp:About>

⁵ <http://www.neon-project.org/web-content/>

- the BPEL partner activities: receive, reply, invoke;
- the BPEL structure activities: pick, sequence, flow, while, switch.

3.1 A Comparison Analysis between the DDPO Tasks and the BPEL Activities

The first step in constructing the proposed ReODP to support ontologizing of the business processes and handling them into the executable business processes is based on the syntactic comparison of the BPEL activities and the DDPO tasks. Table 1 illustrates the result of a comparison of the BPEL receive activity and the DDPO hybrid task.

Table 1: A syntactic comparison of the BPEL activity and the DDPO task

| Syntax of the BPEL activity [Barreto et al., 2007] | Syntax of the DDPO task [Masolo et al., 2004] |
|--|---|
| <pre><receive partnerLink="nc" port Type="qn" operation="nc" variable= "nc"? createInstance = "yes no"? standard-attr> standard-elem <correlations>? <correlation set="nc" initalize=" yes no"? +</correlations> </receive></pre> | <pre>HybridTask(x) =_{df} ComplexTask(x) ∧ (∃y, z. Component (x, y) ∧ Component(x, z) ∧ y ≠ z ∧ Co ntrolTask(y) ∧ ActionTask(z)</pre> |

The connection points between the BPEL receive activity and the DDPO hybrid task can be explained as follows: the BPEL receive partner activity allows receiving messages from an external partner [Barreto et al, 2007]. Therefore, the receive activity specifies the partner link and operation of the partners. Additionally, it specifies a variable (or a set of variables) that holds the requested data that will be received from the partner. The requested data is specified by source and it comes either from messages exchanged with a partner, or it is intermediate data that is private to the process [Barreto et al, 2007]. The source of the requested data is specified by the link name that can have a transition condition, which offer a mechanism to split the control flow based on the required conditions. Therefore, we connect the BPEL receive activity with the DDPO hybrid task, which is a complex task that has at least one control task and one action task as components. The way of connecting the BPEL receive activity to the DDPO hybrid task is shown in the next Section.

4 The Reengineering Patterns: WSDL-DDPO and BPEL-DDPO

We are interested to enable ontology (re)engineering of an existing domain, which is DDPO-based, to fully support the knowledge stack that is required for executing the business processes by the workflow engine.

Following the syntax and the semantics of both the WSDL and the BPEL business process definition, we have noticed the collections of the WSDL and the BPEL business process elements that can be connected with the DDPO concepts. Thus, we have specified two ReODPs called the WSDL-DDPO and the BPEL-DDPO patterns. As the process of modelling the transition conditions (rules) is application specific, we conclude that the WSDL-DDPO and the BPEL-DDPO patterns enable reuse of knowledge, whereas the transition rules of business processes must be defined separately (and manually). Hence, we conclude that the BPEL-DDPO pattern that defines the transition rules is only semi-automatically applicable.

Table 2: The WSDL-DDPO pattern

```

edns : description
--- wsdlDefinition : wsdlTargetNamespace (>0)
--- wsdlMessage : wsdlName (=1)
      :  $\exists$ dol:proper_part some wsdlPartOfMessage
--- wsdlPartOfMessage: wsdlName (=1)
      : wsdlType (=1)
      : wsdlElement (optional)
      :  $\exists$ dol:proper_part some wsdlPortType
--- wsdlPortType : wsdlName (=1)
      :  $\exists$ dol:proper_part some wsdlOperation
--- wsdlOperation : wsdlName (=1)
      : wsdlInputMessage (=1)
      : wsdlOutputMessage (=1)
      : wsdlFaultName (optional)
      : wsdlFaultMessage (optional)
      :  $\exists$ edns:d_uses some edns:task
--- wsdlBinding : wsdlName (=1)
      : wsdlType (=1)
      : wsdlSoapBinding (=1)
      :  $\exists$ edns:d_uses some wsdlOperation
--- wsdlService : wsdlName (=1)
      : wsdlDocumentation (=1)
      :  $\exists$ edns:d_uses some wsdlPort
--- wsdlPort : wsdlName (=1)
      :  $\exists$ edns:d_uses some wsdlBinding
      : wsdlSoapLocation (=1)
--- plnkPartnerLinkType: wsdlName (=1)
      :  $\exists$ dol:proper_part some plnkPartnerRole
--- plnkPartnerRole: wsdlName (=1)
      :  $\exists$ edns:d_uses some wsdlPortType
--- wsdlSoapBinding: soapStyle(rpc I document)
      :  $\exists$ edns:d_uses some soapTransport

```

4.1 The WSDL-DDPO Pattern

The proposed WSDL-DDPO pattern is shown in Table 2 and described as follows: a `wsdlBinding` element has two attributes: a `wsdlName` that defines the name of

the binding and a `wSDLType` that points to the port of the binding. Additionally, the `wSDLBinding` element has a `wSDLSoapBinding` and uses certain `wSDLOperation`. The `wSDLSoapBinding` element has a `soapStyle` attribute and uses some `soapTransport` attribute. The `a_d_uses` and `a_proper_part` relations represent connections between the WSDL elements and the DOLCE concepts (`dol:` stands for DOLCE concepts, whereas `edns:` is used in the DDPO).

4.2 The BPEL-DDPO Pattern

The BPEL-DDPO pattern is shown in Table 4. The very essence of this pattern consists in considering a `bpelSource` element, which has a `bpelLinkName` attribute and can have a transition condition associated with the status of the link (true or false). The transition conditions are application dependent and influence on the BPEL-DDPO patterns to be semi-automatically applicable.

An example of the transition conditions is given in Table 3.

Table 3: An example of the transition conditions

```
<bpel:sources>
  <bpel:source linkName="target-to-request">
    <bpel:transitionCondition>$decision.tRequest = 'yes'
and $decision.changes = 'no'</bpel:transitionCondition>
  </bpel:source>
  <bpel:source linkName="target-to-assign">
    <bpel:transitionCondition>$decision.tRequest = 'no'
and $decision.changes = 'no'</bpel:transitionCondition>
  </bpel:source>
</bpel:sources>
```

Table 4: The BPEL-DDPO pattern

```
edns:description
---  bpelProcess      : bpelName (=1)
                        : bpelTargetNamespace (>0)
                        : bpelImportLocation (=1)
                        : bpelImportNamespace (=1)
                        : ∃ dol:proper_part some bpelPartnerLink
                        : ∃ dol:proper_part some bpelVariable
                        : ∃ dol:proper_part some bpelFlow
---  bpelPartnerLink : bpelName (=1)
                        : bpelPartnerLinkType
---  bpelMyRoleLink  : bpelMyRole
---  bpelPartnerRoleLink : bpelPartnerRole
---  bpelVariable     : bpelName (=1)
                        : bpelMessageType
---  bpelCatch        : bpelFaultName (=1)
                        : bpelFaultMessageType
```

```

      : bpelFaultVariable
      :  $\exists$  dol:proper_part some bpelFaultReply
---  bpelFaultReply: bpelFaultName
      : bpelReplyOperation
      : bpelPartnerLinkReference
      : bpelVariableReference
---  bpelFlow      : bpelLink(>0)
      :  $\exists$  dol:proper_part some bpelReceive
      :  $\exists$  dol:proper_part some bpelSource
      :  $\exists$  dol:proper_part some bpelReply
      :  $\exists$  dol:proper_part some bpelInvoke
---  bpelReceive  : bpelName(=1)
      : bpelOperation
      : bpelPartnerLinkReference
      :  $\exists$  edns:d_uses some wsdlPortType
      : bpelVariableReference
      :  $\exists$  dol:proper_part some bpelSource
---  bpelSource   : bpelLinkName(=1)
                        => bpelTransitionRule
---  bpelReply    : bpelName(=1)
      : bpelOperation
      : bpelPartnerLinkReference
      : bpelVariableReference
      :  $\exists$  edns:d_uses some wsdlPortType
      : bpelTarget(>0)
---  bpelInvoke   : bpelName(=1)
      : bpelOperation
      : bpelPartnerLinkReference
      :  $\exists$  edns:d_uses some wsdlPortType
      : bpelInputVariable(restrictedString)
      : bpelOutputVariable(restrictedString)
      : bpelTarget(>0)
      :  $\exists$  edns:d_uses some bpelSource
---  bpelAssign   : bpelName(=1)
      :  $\exists$  dol:proper_part some bpelSourceTargetPair
      :  $\exists$  edns:d_uses some bpelCopyTo
---  bpelCopyTo   : bpelName(=1)
      : bpelVariableReference
---  bpelSourceTargetPair: bpelTarget(=1)
      :  $\exists$  edns:d_uses exactly 1
      :  $\exists$  edns:d_uses some bpelSource

```

5 Conclusion

The process of creating an executable business process requires both the BPEL process description and the WSDL description of all business operations that will be invoked to carry the activities of the BPEL process. Thus, in this paper we have proposed two patterns called the WSDL-DDPO and the BPEL-DDPO to connect the

knowledge from the business process models (BPEL and WSDL) with the ontological models (DOLCE foundational ontology).

As the transition conditions in the BPEL are application dependent, they must be modelled separately, which causes that the BPEL-DDPO pattern cannot be automatically applied and executed by the workflow engine. Hence, our further work will investigate the new possibilities in modelling the transition conditions as a part of the proposed BPEL-DDPO pattern.

Acknowledgements

This work has been supported by the ImportNET project (IST-2006-033610). Many thanks to Merlin Holzapfel for his great contribution in the definition of the ReODPs proposed in this paper.

References

- [Aranguren, 2005] Aranguren, M.E.: "Ontology Design Patterns for the Formalization of Biological Ontologies"; MSc Thesis at the University of Manchester, Department of Computer Science (2005).
- [Barreto et al., 2007] Barreto, C., et al.: "Web Service Execution Language Version 2.0 – Primer" OASIS (2007).
- [Damjanovic, 2008] Damjanovic, V.: "DOLCE and Pi-Calculus Rendezvous Semantics for Business Processes"; Proc. of the 1st International Workshop on Knowledge Reuse and Re-engineering over the Semantic Web (KRRSW 2008), Tenerife (2008).
- [Damjanovic et al., 2007] Damjanovic, V., Behrendt, W., Plössnig, M., Holzapfel, M.: "Developing Ontologies for Collaborative Engineering in Mechatronics"; Proc. of the 4th ESWC 2007, Austria (2007).
- [Devedzic, 2002] Devedzic, V.: "Understanding Ontological Engineering"; Communications of the ACM, Vol.45, No.4, (2002) 136-144.
- [Clark et al., 2003] Clark, P., Thompson, J., Porter, B.: "Knowledge Patterns"; Springer. International Handbooks on Information Systems (2003).
- [Gangemi et al., 2004] Gangemi, A., Borgo, S., Catenacci, C., Lehmann, J.: "Task Taxonomies for Knowledge Content"; METOKIS Deliverable D07 (2004).
- [Gangemi, 2005] Gangemi, A.: "Ontology Design Patterns for Semantic Web Content"; LNCS 1729, ISWC 2005, (2005) 262-276.
- [Masolo et al., 2004] Masolo, C., Borgo, S., Gangemi, A., Guarino, N.: "Ontology Library"; WonderWeb Deliverable D18 (2004). Online available: <http://wonderweb.semanticweb.org/deliverables/documents/D18.pdf>
- [Pisanelli et al., 2002] Pisanelli, D., Gangemi, A., Steve, G.: "An Ontology of Descriptions and Situations for Lyee's hypothetical World"; CNR-ISTC, Rome, Italy (2002).
- [Reich, 1999] Reich, J.R.: "Ontological Design Patterns for the Integration of Molecular Biological Information"; Proc. of the GCB'99, Germany (1999).
- [Staab et al., 2001] Staab, S., Erdmann, M., Maedche, A.: "Engineering Ontologies Using Semantic Patterns"; Proc. of the IJCAI '01, USA (2001).

- [Svatek, 2004] Svatek, V.: “Design Patterns for Semantic Web Ontologies: Motivation and Discussion”; Proc. of the 7th Conference on Business Information Systems (2004).
- [Vernadat, 1996] Varnadat, F.B.: “Enterprise Modelling and Integration: Principles and Applications”; Chapman & Hall, London (1996).
- [W3C WSDL, 2001] Web Service Description Language (WSDL) 1.1, W3C Note, 2001. Online available: <http://www.w3.org/TR/wsdl>