Towards an XPDL compliant process ontology

Armin Haller, Walid Gaaloul and Mateusz Marmolowski
Digital Enterprise Research Institute
Galway, Ireland
firstname.lastname@deri.org

Abstract

This paper presents initial results on oXPDL, a process interchange ontology based on the standardised XML Process Definition Language (XPDL). XPDL was introduced to allow process model exchange between information systems, most of which are based on proprietary workflow models. Our process interchange ontology oXPDL explicitly models the complete semantics of XPDL process models in a standard Web ontology language. The oXPDL process models may be used for integrated process analysis by querying and reasoning over multiple models and their instances in combination with business rules described in background ontologies, each of which may originate from different information systems.

1 Introduction

The majority of enterprise information systems are process-aware, using some mechanism to support, control, and monitor business processes [4]. Typical examples of such systems, driven by implicit or explicit process models, are enterprise resource planning systems (ERP), customer relationship management systems (CRM), groupware collaboration systems and workflow management systems (WfMS), specifically built for modelling and executing business processes.

All these tools operate on workflow models, typically focused on control-flow captured in procedural process notations. Some more standardised modelling languages exist, such as the Business Process Execution Language (BPEL), the Business Process Modelling Notation (BPMN) and Event-Driven Process Chains (EPCs). Despite partial standardisation, no consensus exists on the representation or conceptual model of complete workflows [12], hampering interoperability, exchange and analysis of process models.

The XML Process Definition Language (XPDL) [13] has been developed for describing and exchanging process models between different applications and their proprietary models and is currently supported by over seventy products\(^1\). In contrast to BPEL, which can be seen as the quasi-standard for process execution over Web services, it serves as a description language only and is not meant for execution.

In this article, we focus on a business intelligence scenario, where analysis data is gathered from several process-aware systems. Since process data is combined from different systems, the data needs to be integrated before the business analysis can be performed. Furthermore, the process data should typically be combined with business information such as partner data, stock data, or other background knowledge.

1.1 Motivating example

We illustrate these challenges with a simple XPDL process of a quoting and purchasing scenario modelled according to the definition in RosettaNet standard processes PIP3A1 and PIP3A4. The BPMN representation of the process is shown on the left of Figure 1, an excerpt of the corresponding XPDL document is shown on the right. In a business analysis scenario, we would like to answer a defined set of competency questions [14] based on the process model and different background ontologies. Due to space limitations we only present exemplary questions here, the full list can be found in [8].

1. How many times has the “Purchase” process been suspended?
2. Which activities of a particular process belong to some subprocess?
3. Does a message in PIP3A4 include some substitutable product?
4. In which processes does John Doe participate in?
5. Which partners should implement a web service operation in a process type?

\(^1\)See http://www wfmc.org/standards/xpdl.htm
Figure 1: RosettaNet quoting and purchasing process

1.2 Approach: ontologising XPDL

We explicitly represent the informal underlying semantics of XPDL in a formal and expressive Web ontology language. We do not impose our interpretation on process models, but carefully formalise the semantics specified in the XPDL standard. Formalising the XPDL standard allows automatic translation of XPDL models (instance documents) into Web documents based on standardised Web ontology languages such as OWL or WSML. Based on these Web ontology languages, the ontologised process models can be interlinked and enriched with business knowledge in existing background ontologies. We further propose to extend the oXPDL model with instance related properties (oXPDL+). These extensions can model the events taking place during the execution of a process model. The concepts in oXPDL dealing with run-time information are based upon standard workflow log meta models such as the one described in MXML [3].

2 Related work

Our work is partially based on PSL [6], a first-order logic process ontology: the PSL axiomatisation can be used for behavioural reasoning in our ontology. In contrast to our work, PSL does not facilitate integration and reuse of other ontologies with background knowledge in Web ontologies.

Other process interchange formats include the Petri Net Markup Language [1], the XML Metadata Interchange specification [11] and the Graph eXchange Language [15]. However, these have only informal semantics, focus on control-flow with little attention to data modelling or organisational modelling, and do not integrate with background knowledge.

Business intelligence techniques based on process models have been proposed by Grigori et al. [5], who describe a process mining toolset on top of HP’s Process Manager. The approach is limited to deadline estimation exception prediction and does not propose a model for more generic business intelligence operations.

3 oXPDL ontology

The ontology covers all of XPDL and consists of around 125 concepts and some 50 logical axioms. Our ontology and the modelling approach are agnostic of the choice for a particular ontology language. Without loss of generality for the overall approach, we chose the WSML [2] ontology language as modelling language and output of the XPDL translator. The ontology and translation could also be implemented with another Web ontology language. As mentioned before, the ontology is not designed for manual process modelling: our tool automatically translates exported process instances to oXPDL and allows analysis through query answering and reasoning. For readability we present the complete ontology in five sections, focusing on respectively the functional, control, informational, organisational, and operational aspects [9]. The complete ontology and the translation tool can be found online2.

3.1 Functional Aspect

The functional aspect provides modelling facilities to define workflow hierarchies and generic properties of workflow models. The majority of functional properties in the oXPDL model are referenced through the PackageHeader concept, defining the hierarchical layering of workflows, the type declarations, applications and artifacts shared between process models, and the properties related to the graphical layout to represent a BPMN model or a proprietary graph-based model.

Additional information required to model process instances in oXPDL+ are mainly concerned with status information, data and performative aspects of a process. Most importantly, a OccurrenceStatus concept is used to model the execution state of the entire process or parts of it. The status information (i.e. occurrence states such as active, suspended, resumed, cancelled, aborted, assigned etc.) can be used to determine the current state of execution (if queried at runtime) or to analyse the execution results of individual tasks or the entire process after completion of the process. Our competency question #1 (“How many times has the “Purchase” process being suspended?”) can be answered with a query ranging over status properties as shown in listing 1.

---

2See http://www.m3pe.org/oXPDL/
3.2 Control Aspect

The control aspect is concerned with the task ordering in a process model and the routing along those tasks. In XDPDL as well as in the proposed organisation in oXPDPL, the Process type constitutes the primary modelling element. It groups related activities, data, and resources together. Activities in XDPDL represent the reusable task behaviour in a process and take one of the following types, a triggered event, a route activity that constrains the ordering of activities or a block activity. In oXPDPL all activity types are modelled accordingly.

Listing 1: Answering competency question #1

3.3 Informational Aspect

The informational aspect deals with data production and data consumption, i.e. the data flow between tasks. We provide a direct mapping from all data types to the ontology. However, more importantly, the modelling of messages can significantly benefit from an ontological modelling approach. Whereas messages in XDPDL are only referenced by an external identifier, in oXPDPL such identifiers can point to an external message ontology. This allows one not only to use the type hierarchy of the message in the evaluation of expressions or in the oXPDPL mapping class, but control flow or event triggers can actually be based on an evaluation of the message content. For our motivating example we can reuse a RosettaNet message ontology to reference the types of the PIP3A1 and PIP3A4 messages. This approach, however, requires that there is an infrastructure in place that translates the actual XML messages used in the communication to ontology instances as described in Haller et al. [7]. If such an approach is followed and the RosettaNet ontology is imported, we can query our model for the competency question #3 (does a message in PIP3A4 include some substitutable product) with the query shown in Listing 3.

Listing 2: Subprocesses and their associated activities

Listing 3: Answering competency question #2

Listing 4: Answering competency question #3

This query is useful to determine if the provider (the process owner in our motivating example) quotes a substitutable product that is a replaceable to the initially requested product.

3.4 Organisational Aspect

The organisational aspect defines who is responsible to perform a task in a workflow and gives the modeler means to assign constraints on the agent (be it a human or a machine). The organisational aspect in an XPDPL document is only weakly defined; a process participant is simply a token of the following types: resource set, resource, organisational unit, role, human, or system. In oXPDPL these types are modelled as independent classes based on representations in the Suggested Upper Merged Ontology (SUMO) [10], a richly axiomatised formal ontology. Modelling the organisational properties semantically allows one to query the knowledge base for competency question #4 (in which processes does John Doe participate) with the query shown in Listing 5.

Listing 5: Answering competency question #4

3.5 Operational Aspect

The operational aspect deals with the modelling of applications to be managed and invoked by the WMS. These applications can be very different in nature, but the technical details of the application programs are
always kept transparent in the workflow model. Although XPDL models different application types, such as EJBs, Java Objects, XSLT scripts, Forms and Business Rules, the most important modelling element are Web Services. The Web Service modelling in oXPDL was enriched as follows. First, the associated Input and Output messages are referenced as a type Message. Second, PartnerLinks and PartnerLinkTypes are explicitly linked, similarly to the modelling of Web Services in BPEL. In oXPDL the PartnerLink modelled in the PackageHeader, as described in section 3.1, references this partnerLinkType and defines which role is taken by the process itself and which role is taken by a partner. In this way, the partnerLinkType describes a contract between two partners in terms of their roles (which in oXDPL are also related to the Pool concept) and the corresponding WSDL portTypes the partners have to provide.

Through the explicit referencing of messages and the PartnerLinkType we can query the model for competency question #5 (which partners should implement a web service operation in ProcessType2) with the query shown in Listing 6.

Listing 6: Answering competency question #5

4 Converter Implementation

The process model conversion involves two steps. First, the oXPDL ontology is loaded, serving as the language grammar including the concept hierarchy, its attributes and their cardinalities and second, the XPDL instance are analysed. The complete conversion algorithm of an XML instance according to the XPDL schema is illustrated in the activity diagram in figure 2. The result of the conversion process is a knowledge base with the set of classes and properties from oXPDL and the associated instances created based on the XML input.

5 Conclusion

We have presented an approach to ontologise the XPDL standard, resulting in our oXPDL ontology, extended with workflow log modelling capabilities defined in oXPDL+. We have explained the ontologisation methodology based on a guiding set of “competence questions”, and have shown how oXPDL models can answer these questions on an automatically translated XPDL process model.

References