Reasoning on Business Processes and Ontologies in a Logic Programming Environment

Michele Missikoff[1], Maurizio Proietti[1], Fabrizio Smith[1,2]

[1] IASI-CNR, Viale Manzoni 30, 00185, Rome, Italy
[2] DIEI, Università degli Studi de L'Aquila, Italy
{missikoff, maurizio.proietti, fabrizio.smith}@iasi.cnr.it

Abstract. In this paper we present a semantic approach to Business Process (BP) Management. The proposal is based on a synergic use of an ontological framework (OPAL), to capture the semantics of a business scenario, and a business process modelling framework (BPAL), to represent the underlying application logic. Both frameworks are grounded in a logic-based formalism (Logic Programming) and therefore it is possible to apply effective reasoning methods to make inferences over a BPKB (Business Process Knowledge Base) stemming from the fusion of the two.

1 Introduction

Business Process (BP) management is constantly gaining popularity in various industrial sectors and in the public administration. But, despite the growing academic interest and the penetration in the business domain, heterogeneous and ad-hoc solutions that often lack a formal semantics have been so far proposed to deal with several arisen issues, such as: cross-enterprise integration and collaboration, adoption of organizational and data models in conjunction with workflow models, query and retrieval of BP fragments, BP composition.

In order to increase the level of automation in the specification, analysis, implementation and monitoring of BPs, various papers have advocated the enhancement of BP management tools by means of well-established techniques from the area of the Semantic Web, like, for instance, computational ontologies [1,2]. The use of an ontology allows an unambiguous definition of the entities occurring in the domain, and eases the interoperability between software applications and the reuse/exchange of knowledge between human actors. However, there are still several open issues regarding the combination of workflow languages (with their specific execution semantics) and ontologies, and the accomplishment of reasoning tasks involving both these components.

In this paper we present a logic-based framework that aims at providing a uniform and formal representation of both the behavioral (i.e., workflow-related) and the structural (i.e., ontology-related) domain knowledge about a business process. Our framework is also equipped with a powerful inference mechanism supported by the tools developed in the area of Logic Programming [3].

1 Business Process Knowledge Representation

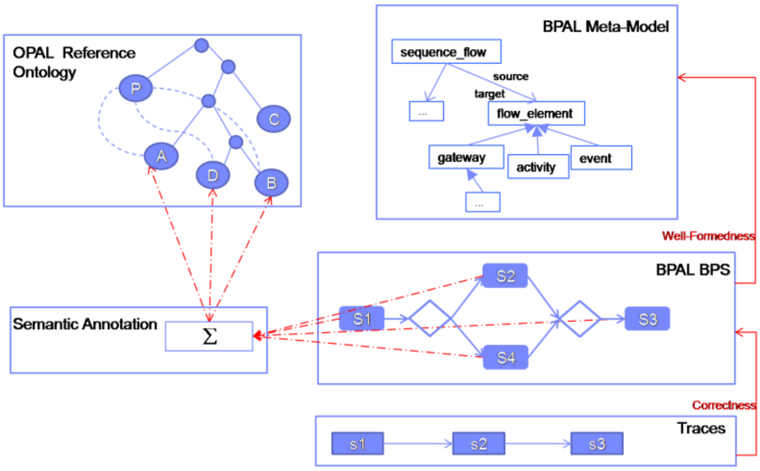

Fig. 1. Business Process Knowledge Base

The knowledge about business processes and the context where they operate, is stored in a *Business Process Knowledge Base (BPKB),* as exemplified in Figure 1 and briefly described in the following.

OPAL [4] is an ontology representation framework supporting business experts in building a structural ontology, where concepts are defined in terms of their information structure and static relationships. OPAL provides a set of upper level concepts and a set of design principles (patterns) to capture the active entities (actors), passive entities (objects), and transformations (processes). A significant core of an OPAL ontology can be formalized by a fragment of OWL, by using the OWL-RL [5] profile. OWL-RL, is an OWL subset designed for practical implementations using rule-based technologies such as logic programming [6].

BPAL [7] (Business Process Abstract modelling Language), is a logic-based language (grounded in Horn Logic) that provides a declarative modeling method capable of fully capturing the procedural knowledge in a business process. BPAL constructs are based on the BPMN 2.0 specification [8] and provide a comprehensive modelling method that spans from the ground level (to model the traces that are produced by the execution of a BP), to the BP schema (BPS) modelling level (where the designer actually defines the diagram that represents the business logic of the BP), to the meta-modelling level (the basic formation rules that guide the designer in the specification of the BP schema).

Semantic Annotation is a correspondence between elements of the BP schema and elements of the Reference Ontology specified using the 'sigma' predicate. It consists of a set of assertions of the form $\sigma(Act,Conc)$, where *Act* is a constant that denotes an entity of a BP schema, and *Conc* is a constant used to denote a concept defined in the ontology. This relation allows us to specify the meaning of the entities of a business process in terms of a suitable conceptualization of the domain of interest.

2 Reasoning with the Business Process Knowledge Base

The components of the *BPKB* introduced in the previous section are formalized by a First Order Logic theory, defined as

$$\boldsymbol{BPKB} = \boldsymbol{BRO} \cup \boldsymbol{\Sigma} \cup \boldsymbol{M} \cup \boldsymbol{B} \cup \boldsymbol{T}$$

where: ***BRO*** is an OPAL Business Reference Ontology; **Σ** is the semantic annotation, i.e. a set of assertions of the form $\sigma(Act,Conc)$; ***M*** is the theory formalizing the meta-model and the related notion of well-formedness of a BP schema; ***B*** is a set of BPAL BP schemas, i.e. a set of assertions (ground facts) constructed from the BPAL alphabet; ***T*** is the theory formalizing the trace semantics of a BP schema and the notion of correctness of a trace w.r.t. that schema.

A relevant property of the *BPKB* is that it has a straightforward translation to a logic program [3], which can be effectively used for reasoning within a Prolog environment. This translation allows us to deal within a uniform framework with several kinds of reasoning tasks and combinations thereof, within the uniform framework of logic programming. Every component of the *BPKB* defines a set of predicates that can be used for querying the knowledge base. The reference ontology ***BRO*** and the semantic annotation **Σ** allow us to express queries in terms of the ontology vocabulary. The predicates defined by the meta-model theory ***M*** and by the BP schemas ***B*** allow us to query the schema level of a BP, verifying properties regarding the flow elements occurring in it (*activities, events, gateways*) and their relationships (*sequence flows*). Finally, the predicates defined by the trace theory ***T***, allow us to express queries about the behavior of a BP schema at execution time, i.e., verify properties regarding the execution semantics of a BP schema.

In order to provide the user with a simple and expressive query language that does not require to understand the technicalities of the logic engine, we proposed in [9] *QuBPAL*, a simple query language based on the SELECT-FROM-WHERE paradigm that can be translated to Prolog[1] queries for their evaluation. As example we report in the following a *QuBPAL* query:

*SELECT <?p,?s,?e >FROM **
WHERE activity(?s::ReceivingPO), activity(?e::Delivering),
precedence(WaitingClearence,Delivering,?p,?s,?e)

This query returns all the well-formed process fragments (i.e., structured blocks [7]) such: (i) start with an activity of *ReceivingPO* (i.e., an activity annotated with the concept *ReceivingPO*), (ii) end with an activity of *Delivering,* and (iii) contain an activity of *WaitingClearence* which is always executed (not necessary immediately) before *Delivering.* The SELECT statement defines the output of the query evaluation, which in this case is a process fragment identified by the triple <*?p,?s,?e*>, where *?p* is a BP identifier, *?s* is the starting element, and *?e* is the ending element. The FROM statement indicates the process(es) from which data is to be retrieved, in this case "*"

[1] In particular queries not involving ***T*** are translated to Datalog queries with stratified negation.

stands for the whole repository. In the **WHERE** statement it can be specified an expression which restricts the data returned by the query.

3 Implementation

A prototype of the proposed framework has been implemented as a Java application, interfaced with the XSB logic programming and deductive database system [10]. The BPAL platform is depicted in Figure 2. On the left part of this figure, enclosed in a dotted line, we have are grouped the components involved in the *set up phase*, where the BPKB is built.

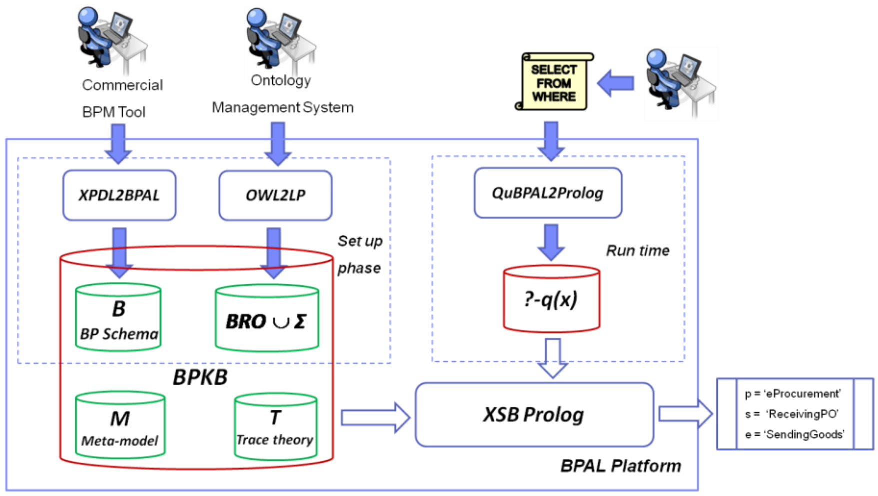

Fig. 2. Architecture of the BPAL Platform

The process repository ***B*** is populated by process schemas modeled by business experts in XPDL (XML Process Definition Language [11]) and translated into BPAL by means of the service *XPDL2BPAL*. The business reference ontology ***BRO*** is imported from the Athos OPAL Ontology Management System [12] and is added to the *BPKB* together with the semantic annotation **Σ** of the BP schemas**.** Both ***BRO*** and **Σ** are represented in the OWL language. OWL ontologies (restricted to the RL profile) are imported into the *BPKB* in the triple notation by the service *OWL2LP.* A Prolog translation of the OWL 2 RL/RDF rules [5] is also included in the *BPKB* to implement reasoning over the ontology. The *BPKB* is completed by the logic programs encoding the meta-model theory ***M***, and the trace theory ***T****.* After the population of the *BPKB* the reasoning tasks can be performed at *run time* by querying the knowledge base through *QuBPAL* queries, that are translated into Prolog by the service *QBPAL2Prolog* and evaluated as goals by the XSB engine. These component are enclosed in a dotted rectangle on the right part of Figure 2.

4 Conclusions

In this paper we presented the main ideas of a framework conceived to complement existing business modeling tools by providing advanced reasoning services. The proposed platform consists of several parts: (i) an ontological framework, OPAL, to capture the semantics of the business scenario; (ii) a business process modeling framework, BPAL, to capture the application logic; (ii) a reasoning engine, based on Logic Programming, that operates on the above two structures in an integrated way; (iv) a BP query language, developed on top of the reasoning engine; finally, (v) a verification mechanism, tightly connected to the latter.

The discussed *BP Knowledge Base* constitutes the base of a knowledge representation framework that we want to extend in several directions. First of all by handling any graph-structured BP schema (without the blocked assumption), and hence the verification of behavioral properties over (possibly) infinite sets of traces. We are also investigating the verification at *run time* (i.e. over a running instance of the process during its enactment) and the *a-posteriori* analysis (i.e., log mining) over the information stored during the execution. On an engineering ground, we are exploring the problem of manipulating, merging and aggregating a set of business process fragments in the contexts of BP Composition and BP Re-engineering.

5 References

1. Hepp, M., Leymann, F., Domingue, J., Wahler, A., Fensel., D.: Semantic business process management: A vision towards using semantic web services for business process management. In: ICEBE 2005, pp. 535–540. IEEE Computer Society, Los Alamitos (2005).
2. Roman, D., et al.: Web Service Modeling Ontology. Applied Ontology, 1(1): 77-106, 2005.
3. Lloyd, J.W.: Foundations of Logic Programming. Springer-Verlag, Berlin, 1987.
4. D'Antonio, F., Missikoff, M., Taglino, F.: Formalizing the OPAL eBusiness ontology design patterns with OWL. In the 3rd I-ESA Conference, 2007.
5. OWL 2: Profiles, http://www.w3.org/TR/owl2-profiles.
6. Grosof, B. N., Horrocks, I., Volz, R., Decker, S.: Description Logic Programs: Combining Logic Programs with Description Logic. In the Proc. of WWW 2003, Budapest, Hungary.
7. De Nicola A., Missikoff M., Proietti M., Smith F.: An Open Platform for Business Process Modeling and Verification, International Conference on Database and Expert Systems Applications, DEXA, September 2010. LNCS 6261 Springer 2010.
8. Business Process Model and Notation V. 2.0, 2009, http://www.omg.org/spec/BPMN/ 2.0.
9. Missikoff M., Proietti M., Smith F. A Business Process Knowledge Base for Composite Services Development. International Workshop BSME, Malaga, June 2010.
10. The XSB Logic Programming System. Version 3.1, Aug. 2007, http://xsb.sourceforge.net.
11. XPDL 2.1 Complete Specification, Oct. 2008, http://www.wfmc.org/xpdl.html.
12. ATHENA, D.A3.2 "Updated version of the Ontology Authoring and Management System with semantic search functions", ATHENA IP, deliverable, 2005.