Chapter 17
Enhancing the Correctness of BPMN Models

Anacleto Correia
Instituto Politécnico de Setúbal (IPS/EST), Portugal

Fernando Brito e Abreu
ISTAR, Instituto Universitário de Lisboa (ISCTE-IUL), Portugal

ABSTRACT

While some of the OMG's metamodels include a formal specification of well-formedness rules, using OCL, the BPMN metamodel specification only includes those rules in natural language. Although several BPMN tools claim to support, at least partly, the OMG's BPMN specification, we found that the mainstream of BPMN tools do not enforce most of the prescribed BPMN rules. Furthermore, the verification of BPMN process models publicly available showed that a relevant percentage of those BPMN process models fail in complying with the well-formedness rules of the BPMN specification. The enforcement of process model's correctness is relevant for the sake of better quality of process modeling and to attain models amenable of being enacted. In this chapter we propose supplement the BPMN metamodel with well-formedness rules expressed as OCL invariants in order to enforce BPMN models' correctness.

1. INTRODUCTION

BPMN (Business Process Modeling and Notation) is one of the most recent process modeling languages (OMG, 2011). So, it is grounded on the experience of earlier ones, which ontologically makes it one of the most complete process modeling languages available (J.C. Recker, Indulska, Rosemann, & Green, 2005; J.C. Recker, Rosemann, Indulska, & Green, 2009). BPMN is also nowadays the business process notation most used among BPM practitioners, and the process modeling language with more modeling tools available[1] (Harmon & Wolf, 2011). BPMN has also transformations to other notations available, such as Petri-Nets and CSP, which allow the use of accessible tools for formal verification (Dijkman, Dumas, & Ouyang, 2007; Wong & Gibbons, 2008).

Version 2 of the BPMN standard, is a step forward in the alignment of process modeling with the OMG's initiative of Model Driven Architecture (MDA) (OMG, 2001). The BPMN language definition

DOI: 10.4018/978-1-5225-9615-8.ch017

is based upon a metamodel built with the UML, the standard *de facto* for software engineering modeling (UML, 2007a, 2007b). Therefore, the BPMN standard formalization of the process modeling concepts and their relationships is accomplished by means of a metamodel. The specification defines different types of conformance that tools' implementers can adhere for: (i) process modeling (elements that are part of the orchestration in a single process, as well as elements that participate in the collaboration among processes); (ii) BPMN process execution (the operational semantics support and interpretation of activity life-cycle); (iii) BPEL process execution (mapping of a BPMN model to WS-BPEL); and (iv) choreography modeling (emphasizing modeling of the interaction among participants).

The BPMN standard specification can be referred, for the definition and meaning of each element, as well as regarding how they can be connected. However, it is a complex technical document not intended to common business modelers. Besides, the standard does not provide guidance on how the modeling notation should be used to attain a comprehensible and expressive BPMN model. Moreover, a great deal of definitions and rules are presented informally in plain English. To fulfill this gap, best modeling practices and complementary well-formedness rules for BPMN models, have been proposed by academics and practitioners (Allweyer, 2010; Becker, Rosemann, & Von Uthmann, 2000; Correia & Brito e Abreu, 2012; Jan Mendling, Reijers, & Cardoso, 2007; J. Mendling, Reijers, & van der Aalst, 2010; Silver, 2009; Vanderfeesten, Reijers, Mendling, van der Aalst, & Cardoso, 2008; White & Miers, 2008).

BPMN is a semantically rich modeling language. While, for instance, a UML Activity Diagram has around 20 different modeling constructs, a BPMN process model diagram (the more complex of the 3 available ones) has around 100 different modeling constructs, including 51 event types, 8 gateway types, 7 data types, 4 types of activities, 6 activity markers, 7 task types, 4 flow types, pools, lanes, etc. If it is given to BPMN modelers the freedom to combine such a large plethora of modeling constructs, without a powerful validation and recommendation facility embedded in a modeling tool, inconsistencies and even invalid models could easily arise.

A metamodel (M2) describes the abstract syntax of a language by means of meta-classes, meta-associations and cardinality constraints. When UML is adopted for expressing metamodels, Object Constraint Language (OCL) clauses can be used in a declarative way, similar to 1st order predicate logic, to strengthen metamodel syntax and semantics, namely by imposing well-formedness and best practices rules that reduce the sources of modeling malformation (OMG, 2006).

Adding preciseness to OMG's BPMN metamodel, by using such OCL clauses, is the first objective of the work presented herein. The second objective is to validate BPMN models. The USE tool (UML based Specification Environment) was used to embed OCL clauses on the BPMN metamodel and to instantiate it with actual process models (Gogolla, Buttner, & Richters, 2007).

This work intends to contribute for enhancing the correctness of produced business process models, by providing a set of static semantic rules[2] and best-practices rules for designing BPMN models (Aaby, 1996). Since the rules are embedded in the BPMN metamodel, business process models' correctness become intrinsically verified by the language and not ensured by rules implemented in other languages, external tools or checkers.

Some of those rules, were withdrawn from the process modeling language specification, scattered by the text and tables of the document (OMG, 2011). They were expressed in the standard, in natural language yielding sometimes, a dubious interpretation. Other rules came from disseminated best-practices references both from academics and practitioners. The Object Constraint Language (OCL), the language that supplements the UML, was used to rigorously specify and implement the mentioned rules by means