SemantifyingBPMN: a tool to generate BPMN models using DEMO

Sérgio Guerreiro[1,2,3], Pedro Sousa[1,2,3]

[1]INESC-ID, R. Alves Redol 9, 1000-029 Lisbon, Portugal
[2]Instituto Superior Técnico, University of Lisbon, Av. Rovisco Pais 1, 1049-001 Lisbon, Portugal
[3]Link Consulting SA, Av. Duque de Ávila 23, 1000-138 Lisbon, Portugal

Abstract

BPMN is a specification language widely used by industry and researchers for business process modeling and execution. It defines how to articulate its concepts, but no method is provided on how to design the business process models. We address this problem proposing a tool (SemantifyingBPMN) that generates the BPMN models using a social interaction pattern grounded in DEMO, so that models can be discovered and enhanced with a bounded effort. The SemantifyingBPMN tool covers the patterns of: happy-flow, declination, rejection, revocation, customized view and composition of business transactions.

Keywords

BPMN, business transaction, DEMO, model, pattern, social interaction

1. Introduction

A business process is a collection of events, activities, and decisions that brings value to the customers of an organization. Business Process Model and Notation[1] is the *de facto* standard used by industry and researchers for business process modeling and execution. It is also available under the standard ISO/IEC 19510:2013.

As corroborated in the literature, BPMN defines very clearly how to articulate its concepts, but do not provide a semantic for the consequent model. The meaning is usual expressed in the natural language words that are used to name the activities, events, gateways, *etc.* [1, 2]. Accordingly with [3], *"a language is considered to be formal if both its syntax and semantics can be precisely defined. When the semantics is formally defined, sentences in the language then have a unique interpretation"*, this property does not hold in BPMN models. In fact, it is on the modeler's responsibility side to interpret the meaning of a given BPMN model. Conversely, any other modeler could interpret the same model differently, *e.g.*, uBPMN [4] extends BPMN with more notations to deal with ubiquitous computing technologies alleging that BPMN do not offer support for this domain. The study in [5] corroborates this problem statement and reinforces it when remote environments are imposed driving to more coordination problems,

ER FORUM @ 41st International Conference on Conceptual Modeling (ER 2022)
sergio.guerreiro@tecnico.ulisboa.pt (S. Guerreiro); pedro.manuel.sousa@tecnico.ulisboa.pt (P. Sousa)
https://github.com/SemantifyingBPMN/SemantifyingBPMN (S. Guerreiro)
0000-0002-8627-3338 (S. Guerreiro); 0000-0001-7453-5778 (P. Sousa)

[1]BPMN 2.0 Specification, 2011, https://www.omg.org/spec/BPMN/2.0.2/PDF

e.g., due to off-shore teams allocation

The business processes expressed by customers usual refers only to the *"happy flow"* and disregards some other forms of exception handling [6]. This option is not acceptable in business process models under scenarios subject to uncertainty [7]. Therefore, real business processes implementations usually relies on those incomplete semantic models, and then, this problem is further propagated, and amplified, to the BPMN executable process models. The implications of this problem within organizations are manifold *(i)* misunderstanding in business process modeling, reflecting in implementation failures, and *(ii)* inconsistent design in the business process models, reflecting in time consuming tasks for implementation, re-engineering and knowledge dissemination due to an higher learning curve.

To address this problem we propose a tool that enforces the social interaction pattern available in the Enterprise Ontology [8] body of knowledge, specifically in the Design & Engineering Methodology for Organizations (DEMO) theory, onto BPMN models[2]. DEMO Ψ-theory (Performing in Social Interaction-theory) is a communication-centric view proposal that considers the people' cooperation in enterprises, and is detailed in [9]. This pattern-based approach is also found in the literature. [10] shows that many publications discuss their patterns in isolation, and no embracing pattern exists for the many business process modeling patterns proposals available. We stress that our goal is to study solutions that improve the BPMN models discover and enhancement using knowledge available from other semantic-based approach, and not to change the BPMN language itself nor to add new construct to the language.

This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UIDB/50021/2020 (INESC-ID).

2. Mapping DEMO complete pattern onto BPMN models

To explain the mapping of DEMO onto BPMN a motivating example is used. Considering a single business transaction of *producing a product*, at least two business actors need to be considered: the transaction initiator (TI) and the transaction executor (TE). Considering that both actors are located on two different organizations, then two BPMN pools are considered (*cf.* Figure 1). When TI decides to request a product, the task of *Request product* is executed resulting in *(i)* the emission of a communication act (C-act) with the purpose of *Requesting a product* to the TE, and *(ii)* the creation of a new communication fact (C-fact) in the world. This double outcome is repeated in all the business transaction steps, except on the *Execute product* step where no C-act is expected but a P-act is. Figure 1 expands all the business transaction steps included in a DEMO pattern, clarifying how the sequence of states originate a new P-fact in the world. Whenever an actor role is expecting a C-act from the other, a BPMN intermediate catching and an intermediate throwing message events are used. All the business transactions respect this pattern, even when some acts or facts are not observable. In that situation, the acts or facts are considered implicit in the execution of the business transaction.

If a business actor disagrees with another, then a decision point is achieved.

[2]All the models presented in this paper are publicly available at https://github.com/SemantifyingBPMN/DEMO_complete_pattern_in_BPMN

In case of a declination[3], it is up to the TI to issue a new request; on the contrary, in case of rejection, the TE needs to evaluate the rejection arguments before deciding between stopping or re-declaring the product. On the one hand, the C-act decline corresponds to an impossibility of producing a product (*e.g.*, due to stock shortage) and triggers the TE end of process flow. On the other hand, TI upon the reception of a declination could decide to emit a new request or also to stop his/her process flow. The rejection C-act is symmetric with decline. Whenever a TI receives a C-act declare then a product check is performed. If TI do not agree with the delivered product then a C-act of rejection is emitted. Understanding that P-fact already exists in the world, is on the TE responsibility to decide if the rejection is valid or not. If so, a stop C-act is emitted and the TI and TE process flow ends. Otherwise, the declare C-act is re-emitted and the loop is restarted. Again, a possible deadlock could occur, and in the limit can only be solved using litigation.

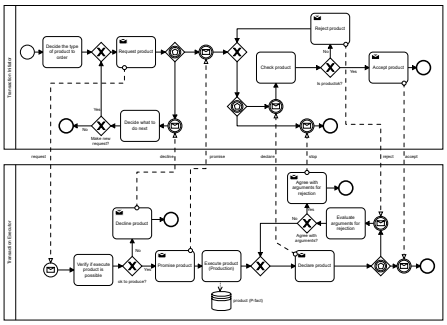

Figure 1: The DEMO standard pattern of a transaction between two actors in BPMN.

For revocations (*revoke a request*, *revoke a promise*, *revoke a declare* and *revoke a accept*) please refer to the full paper available at [9].

In real environments, instead of a single business transaction, they are defined within the scope of a network of actors and business transactions, *e.g.*, a *Payment* transaction succeeds a *Production* transaction. Therefore, the composition between them needs to be addressed. Applied scenarios of these situations are detailed in the proof-of-concepts section. A composition pattern occurs whenever an executor actor role assumes the responsibility of initiating another transaction. The known transaction composition patterns are: request after execution (RaE) and request after declare (RaD) (*cf.* explained in [9]).

2.1. Tool usage and performance

This DEMO mapping onto BPMN is implemented in an open-source software tool named as SemantifyingBPMN and is publicly available online[4]. The required inputs of the software tool are: *(i)* a list of actors, *(ii)* a list of transactions where the actors are involved and specifying the result expected, and *(iii)* the transactions dependencies (RaP, RaE or RaD). The output is a BPMN model file based in the available BPMN 2.0 XSD definition. To facilitate the rendering, each business transaction view, can be configured as happy flow, or happy flow with declinations and rejections, or the complete pattern. Figure 2 depicts some preliminary performance results to generate BPMN models by the SemantifyingBPMN tool. The median of the operating time is around 1 second to produce a BPMN model containing 1 business transaction. Yet, computing time increases along with the increasing number of elements that are configured in TKView.

[3]A declination is considered in this paper as the formal refusal of a participant.
[4]https://github.com/SemantifyingBPMN/SemantifyingBPMN

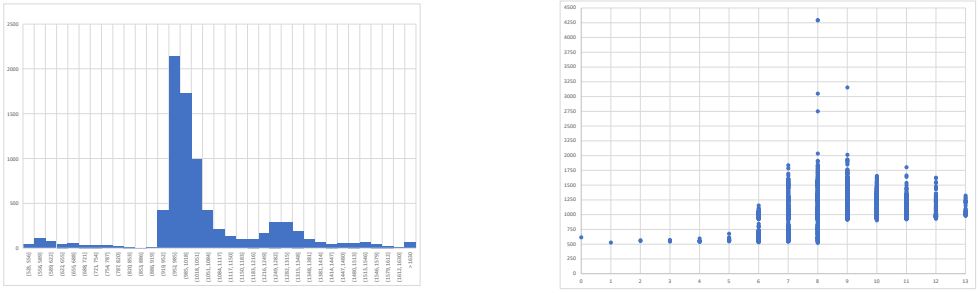

Figure 2: Left side: Histogram depicting the distribution of time that is consumed to generate a BPMN collaboration from 1 DEMO business transaction, 8192 combinations of TKView tested once. Right side: Time consuming to generate a BPMN collaboration from 1 DEMO business transaction, 8192 combinations of TKView tested once. Miliseconds-timebase used.

References

[1] R. M. Dijkman, M. Dumas, C. Ouyang, Semantics and analysis of business process models in BPMN, INFORMATION AND SOFTWARE TECHNOLOGY 50 (2008) 1281–1294. doi:{10.1016/j.infsof.2008.02.006}.

[2] G. Jošt, J. Huber, M. Heričko, G. Polančič, An empirical investigation of intuitive understandability of process diagrams, Computer Standards & Interfaces 48 (2016) 90–111.

[3] H. Mili, G. Tremblay, G. B. Jaoude, E. Lefebvre, L. Elabed, G. E. Boussaidi, Business process modeling languages: Sorting through the alphabet soup, ACM Comput. Surv. 43 (2010). URL: https://doi.org/10.1145/1824795.1824799. doi:10.1145/1824795.1824799.

[4] A. Yousfi, C. Bauer, R. Saidi, A. K. Dey, ubpmn: A bpmn extension for modeling ubiquitous business processes, Information and Software Technology 74 (2016) 55–68.

[5] E. Poppe, R. Brown, J. Recker, D. Johnson, I. Vanderfeesten, Design and evaluation of virtual environments mechanisms to support remote collaboration on complex process diagrams, INFORMATION SYSTEMS 66 (2017) 59–81. doi:{10.1016/j.is.2017.01.004}.

[6] S. Guerreiro, S. J. van Kervel, A. Vasconcelos, J. Tribolet, Executing enterprise dynamic systems control with the demo processor: the business transactions transition space validation, in: Mediterranean Conference on Information Systems, Springer, 2012, pp. 97–112.

[7] A. Jiménez-Ramírez, B. Weber, I. Barba, C. Del Valle, Generating optimized configurable business process models in scenarios subject to uncertainty, Information and Software Technology 57 (2015) 571–594.

[8] J. L. Dietz, H. B. Mulder, Enterprise Ontology: A Human-Centric Approach to Understanding the Essence of Organisation, Springer Nature, 2020.

[9] S. Guerreiro, P. Sousa, A framework to semantify bpmn models using demo business transaction pattern, 2020. URL: https://arxiv.org/abs/2012.09557. doi:10.48550/ARXIV.2012.09557.

[10] M. Fellmann, A. Koschmider, R. Laue, A. Schoknecht, A. Vetter, Business process model patterns: state-of-the-art, research classification and taxonomy, Business Process Management Journal 25 (2019) 972–994.