Proceedings of the Federated Conference on
Computer Science and Information Systems pp. 931–934

ISBN 978-83-60810-51-4

AI Approach to Formal Analysis of BPMN Models. Towards a Logical Model for BPMN Diagrams

Antoni Ligęza, Krzysztof Kluza
AGH University of Science and Technology
al. Mickiewicza 30, 30-059 Kraków, Poland
Email: {ligeza, kluza}@agh.edu.pl

Tomasz Potempa
Higher School of Tarnów
ul. Mickiewicza 8, 33-100 Tarnów, Poland
Email: tpotempa@gmail.com

***Abstract*—Modeling Business Processes has become a challenging issue of todays' Knowledge Management. As such it is a core activity of Knowledge Engineering. There are two principal approaches to modeling such processes: Business Process Modeling and Notation (BPMN) and Business Rules (BR). Both of them are to certain degree complementary. In this paper, we investigate how to build a logical model of BPMN using logic programming and rules. The main focus in on logical reconstruction of BPMN model in Prolog and defining some formal requirements on model correctness enabling formal verification of such models.**

I. Introduction

KNOWLEDGE has become a valuable and critical resource of contemporary organizations. In fact, possession of valid, most complete, up-to-date and essential knowledge has become a decisive factor of success in the so competitive market.

Unfortunately—or no—human possession and processing of knowledge turns out to be fairly inefficient for a number of reasons. Some most obvious ones include difficulties with knowledge sharing, storage, and efficient execution. Hence, *Knowledge Management* (KM) must be supported with tools and techniques coming from Software Engineering and Knowledge Engineering universe [1], [2].

Development and analysis of complex business processes require advanced methods and tools. Two approaches used for this purpose, *Business Process Modeling and Notation* (BPMN) [3] and *Business Rules* (BR) [4], [5], have recently gained wider popularity.

BPMN can be considered as *procedural knowledge representation*; a BPMN diagram represents in fact a set of interconnected procedures. On the other hand, such workflow diagram due to lack of formal model semantics makes attempts at more rigorous analysis problematic. Further, even relatively simple inference requires a lot of space for representation; there is no easy way to specify declarative knowledge, e.g. in the form of rules [6].

Business Rules (BR), also promoted by OMG [7], [8], offer an approach to specification of knowledge in a *declarative* manner [9]. The way the rules are applied is left over to the user when it comes to rule execution. Hence, rules can be considered as *declarative knowledge specification*; inference control is normally not covered by basic rules [10].

Note that rules can fulfill different roles in the system [7]. Some three most important ones cover: *declarative knowledge specification* for inference of new facts, *integrity constraints* for preserving consistency of the knowledge base, and *meta-rules* i.e. rules defining how to use another rules; this may include partial *control knowledge specification* for improving efficiency of inference process.

Rules, especially when grouped into decision modules (such as decision tables), are easier to analyze, however, the possibility of analysis depends on the accepted *knowledge representation language*, and in fact – the logic in use [11].

These two approaches are to certain degree complementary: Business Rules provide declarative specification of domain knowledge, which can be encoded into a BPMN model. On the other hand, a careful analysis of a BPMN diagram allows to extract certain rules governing the business. However, there is no consistent study on possibility of mutual conversion. The main problems seem to consist in lack of formal specification of KR language and some ambiguities of BPMN.

The main common problem of BPMN is lack of a *formal declarative model* defining precisely the semantics and logic behind the diagram. Hence, defining and analyzing correctness of BPMN diagrams (e.g. in terms of termination or determinism) is a hard task. In the papers undertaking the issues of analysis and verification of BPMN diagrams, e.g. [12], [13], the analysis is performed mostly at the *structural* level and does not take into account the semantics of dataflow and control knowledge. BPMN diagrams still wait for solving the issues of formal verification, an approach such as the one presented in [14], [15] in the context of rule-based systems.

In this paper we follow the ideas initially presented in [16]. We pass from logical analysis of BPMN component to their logical models, properties and representation in Prolog. Some prototyped procedures for checking correct data flow are also presented. The model is aimed at enabling definition and further analysis of selected formal properties of a class of restricted BPMN diagrams. The analysis should take into account properties constituting reasonable criteria of correctness. The focus is on development of a formal, declarative model of BPMN components and its overall structure. In fact, a combination of recent approaches to development and verification

The paper is supported by the *BIMLOQ* Project funded from 2010–2012 resources for science as a research project.

of rule-based systems [17], [18], [19] seems to have potential influence on BPMN analysis.

II. A BASIC STRUCTURAL MODEL FOR BPMN

A simplified structural model of BPMN diagram constitutes a restricted abstraction of crucial intrinsic workflow components [16]. We take into account such elements as events (start $\mathbb{S}$ and termination $\mathbb{E}$ events), activities (or tasks $\mathbb{T}$), gateways (split $\mathbb{G}$ and merge $\mathbb{M}$ operations), as well as workflow sequence (modeled as directed links $\mathbb{F} \subseteq \mathbb{O} \times \mathbb{O}$, where $\mathbb{O} = \mathbb{S} \cup \mathbb{E} \cup \mathbb{T} \cup \mathbb{G} \cup \mathbb{M}$). The splits and joins depend on logical conditions assigned to particular branches. It is assumed that there is defined a partial function $\texttt{Cond}\colon \mathbb{F} \to \mathbb{C}$ assigning logical formulae to links. In particular, the function is defined for links belonging to $\mathbb{G} \times \mathbb{O} \cup \mathbb{O} \times \mathbb{M}$, i.e. outgoing links of split nodes and incoming links of merge nodes. The conditions are responsible for workflow control. For intuition, a simple BPMN diagram is presented in Fig. 1.

In order to assure *correct structure* of diagrams a set of restrictions on the overall diagram structure is typically defined; they determine the so-called *well-formed diagram* [13]. Note however, that a well-formed diagram does not assure that for any input knowledge the process can be executed leading to a (unique) solution. This further depends on the particular input data, its transformation during processing, correct work of particular objects, and correct control defined by the branching/merging conditions assigned to links.

III. AN EXAMPLE OF BPMN DIAGRAM WITH RULES

We illustrate our theoretical considerations with an exemplary process, which goal is to establish the temperature for a thermostat system [20]. The temperature selection depends on several aspects, such as season, whether it is a working day or not, and the time of the day. The system consists of 18 declarative inference rules (production rules) specifying the process, e.g.

Rule 1: $aDD \in \{monday, tuesday, wednesday, thursday, friday\} \longrightarrow aTD = wd.$
Rule 2: $aDD \in \{saturday, sunday\} \longrightarrow aTD = wk.$
Rule 3: ...
Rule 18: $aSE = win \wedge aOP = ndbh \longrightarrow aTHS = 14.$

The rules mentioned above are explained in [16]; thus let us here briefly explain only the first two rules. These rules define if we have today (aTD) a workday (wd) or a weekend day (wk) based on the day of the week. Such set of rules is flat and no control knowledge is provided.

Visually, such knowledge base can be depicted using business processes, as in Fig. 1, which shows a top-level specification of the thermostat system. In the diagram, the lower control flow first determines whether the day (aDD) is a workday ($aTD = wd$) or a weekend day ($aTD = wk$) and specifies the value of today (aTD; specification provided with rules 1 and 2). Such subset of rules operating in the same context can be modeled on the lower level as a subprocess, as in Fig. 2.

Even in such a simple example, there are several open issues, such as *data flow correctness*, *split consistency*, *merge consistency*, *termination/completeness* or *determinism* [16].

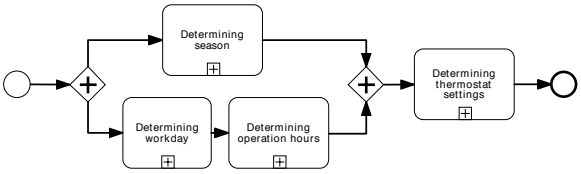

Fig. 1. An example of a top-level specification of the thermostat system

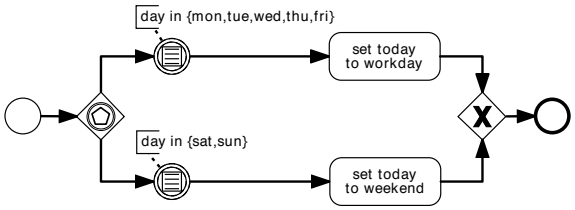

Fig. 2. An example of a detailed specification of determining the day task

IV. LOGICAL ANALYSIS OF BPMN DIAGRAMS

A BPMN diagram can model quite complex processes. Apart from *external consistency* validation an important issue is the *internal consistency* requirement for *correct structure* of the diagram and *correct workflow specification*. The first one refers to static specification of components and their connections. The second one consists in correct work of the structure for all admissible input data specification.

Even having correct structure, the process can gone wrong due to unserved data or wrong workflow control, for example. There are some minimal requirements for a process, such as (i) correct work of process components (tasks), (ii) assuring data flow, (iii) correct work of splits, (iv) correct work of merge nodes, and finally—(v) termination of the overall process.

A more complex issue consists in verification of internal workflow correctness. The analysis must take into account, at least the following aspects:

1) Local correctness requirements: correct specification and work of process components performing activities, correct specification of data flow, correct specification and work of splits, correct specification and work of joins.
2) Global correctness requirements: no deadlocks (the system must work for all admissible input data), termination (the system must terminate work within a finite period of time), determinism (the results should be repeatable for repeated input data).

V. BPMN MODEL IN PROLOG

In this section a Prolog model for the example BPMN diagram is presented. It enables logical analysis of the diagram.

A BPMN diagram is represented as a complex graph with different types of nodes. Each specific component of BPMN is mapped into a Prolog fact. In order to model the structure of a BPMN diagram in a declarative way it is proposed to use a set of PROLOG facts. A generic form can be as follows:

```
component_type(<id>,
   <input_node>,<input-formula>,
   <output_node>,<output-formula>).
```

In practice, such facts can be of different structure for different components. The thermostat example presented in Fig. 3 is defined with the Prolog code in Listing 1.

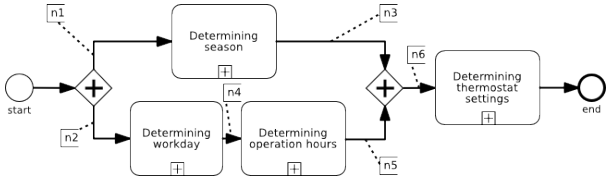

Fig. 3. An example of a top-level specification of thermostat system with the additional Prolog annotations

Algorithm 1 BPMN example coding in Prolog

```
%%%%% BPMN Knowledge Base %%%%%
%%% Start & end nodes declaration
init_node(start).
end_node(end).
nodes([n1,n2,n3,n4,n5,n6]).
%%% Tasks
% task(<id>,<input_node>,
%       <input_formula>,
%       <output_node>,
%       <output_formula>).
task(ds,n1
      [[mspr],[msum],[maut],[mwin]],
      n3,[[spr],[sum],[aut],[win]]).
task(dw,n2,[[d15],[d67]],n4,
      [[twr],[twe]]).
task(do,n4,[[twr,t18],[twr,t8],
      [twr,t917],[twe]],n5,
      [[nonbiz],[biz]]).
task(dt,n6,[[nonbiz,spr],[nonbiz,sum],
      [nonbiz,aut],[nonbiz,win],
      [biz,spr],[biz,sum],[biz,aut],
      [biz,win]],end,
      [[t14],[t15],[t16],[t18],[t20],
      [t24],[t27]]).
% split2(<id>,<split_type>,
%         <input_node>,
%         (<output_node>,
%          <logical_condition>),
%         (<output_node>,
%         <logical_condition>)).
split2(s1,and,start,(n1,true),
       (n2,true)).
% merge2(<id>,<merge_type>,
%         <input_node>,
%         <input_node>,
%         <output_node>).
merge2(m1,and,n3,n5,n6).
```

There are four tasks defined: *define season (ds)*, *define workday (dw)*, *define operation (do)*, *define temperature (dt)*. There are also one split node and one merge node.

As an example of analysis let us present a code for verification of data flow (the funnel condition) between tasks dw and do; the intuition behind is that any output generated by dw should be accepted and further processed by do (see Listing 2).

After calling the funnel check [16] for the internal node $n4$ the system proves that the output formula of task dw implies the input formula of task do. The output of the program, confirming the result of the check is given below:

```
?- funnel(n4).
dw-->n4-->do
true
```

Other static checks for data flow have been implemented and tested as well.

Algorithm 2 BPMN example coding in Prolog: funnel between tasks

```
%%% Funnel condition checking for
%%% nodes among tasks
funnel(N) :-
    task(IDOUT,_,_,N,FOUT),
    task(IDIN,N,FIN,_,_),
    implies(FOUT,FIN),
    write(IDOUT),write('-->'),write(N),
    write('-->'),write(IDIN),nl.
%%% Definition of implication
%%% for two DNF
implies(true,_) :- !.
implies([],_) :- !.
implies([MIN|T],DNF) :-
    imply(MIN,DNF),
    implies(T,DNF).
%%% Definition of implication
%%% DNF |= MIN
imply([],_) :- !.
imply(MIN,DNF) :-
    member(M,DNF), subset(M,MIN), !.
imply(MIN,DNF) :-
    find_subsets(DNF,SDNF),
    reduce(SDNF,RSDNF),
    member(M,RSDNF),
    subset(M,MIN).
find_subsets([],[]) :- !.
find_subsets([G|T],[G|T2]):-
     find_subsets(T,T2).
find_subsets([_|T],T2):-
     find_subsets(T,T2).
```

VI. Related works

To the best of our knowledge, apart from [21], there are no related works defining BPMN model in PROLOG. Andročec used Prolog for specification of business processes; however, he used it to assess the cost and time of running a process and to identify potential problems with resources. Thus, he defined the model for simulation purposes.

Similar attempts to formalization of BPMN models were carried out by Lam [22], Andersson et al. [23], Wong and Gibbons [24] as well as Dijkman and Van Gorp [25]. Other attempts to formalization of process models, however not concerning BPMN, were carried out by Gruhn and Laue [26].

The BPMN model defined by Lam in [22] was used to check the diagrams against the properties specified by a user for a particular model [27]. In our case, the properties, which are correctness requirements, can be used for any BPMN model.

Paper [23] introduced the notion of activity dependency model which identifies, classifies, and relates activities needed for executing and coordinating value transfers. In this model, relations between activities can be specified in terms of notions like resource flow, trust, coordination, and reciprocity. These types of dependencies which can be identified are not a part of the BPMN style, thus the model is not completely BPMN compliant and has another goals then ours.

Dijkman and Van Gorp [25] formalized the BPMN model using graph rewrite rules. However, their goal was also different than ours. They focused on execution semantics. Thus, their approach is suitable for simulation, animation and execution of BPMN models.

Wong and Gibbons [24] defined the semantics of BPMN models using Communicating Sequential Processes, in which a process is a pattern of behavior. They provide only verification of such issues as: hierarchical refinement, partial refinement and hierarchical independence.

VII. Concluding remarks

Our work is a part of an approach in the area of of business processes and rules integration [28]. It constitutes an attempt at providing a logical, declarative model for well-defined BPMN diagram [16]. The model is aimed at defining formal semantics of diagram components and the workflow operation. The main focus is on specification of correct components and correct dataflow. Global termination conditions are specified in a recursive way. We formulated a formal model for a subset of BPMN, defined local and global correctness requirements, and specified the detailed logic that stems from the diagram.

The original contribution of our work consists in:

- presenting open issues corresponding to the BPMN diagrams, such as consistency of elements,
- specifying a BPMN model in the declarative Prolog language, which helps to check the defined correctness requirements.

The logical analysis can be performed *off-line*, on the base of logical requirements of data. However, if such specifications are data-dependant the analysis may be possible only in *on-line* form, separately for any admissible input data.

References

[1] G. J. Nalepa, "Collective knowledge engineering with semantic wikis," *Journal of Universal Computer Science*, vol. 16, no. 7, pp. 1006–1023, 2010. [Online]. Available: http://www.jucs.org/jucs_16_7/collective_knowledge_engineering_with

[2] A. Ligęza, *Validation and verification of knowledge based systems : theory, tools and practice*. Boston, Dordrecht, London: Kluwer Academic Publishers, 1999, ch. Intelligent data and knowledge analysis and verification; towards a taxonomy of specific problems, pp. 313–325.

[3] OMG, "Business Process Model and Notation (BPMN): Version 2.0 specification," Object Management Group, Tech. Rep. formal/2011-01-03, January 2011.

[4] S. W. Ambler, "Business Rules," http://www.agilemodeling.com/artifacts/businessRule.htm, 2003.

[5] A. Giurca, D. Gašević, and K. Taveter, Eds., *Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches*. Hershey, New York: Information Science Reference, May 2009.

[6] G. J. Nalepa, "Proposal of business process and rules modeling with the XTT method," in *Symbolic and numeric algorithms for scientific computing, 2007. SYNASC Ninth international symposium. September 26–29*, V. Negru and et al., Eds., IEEE Computer Society. Los Alamitos, California ; Washington ; Tokyo: IEEE, CPS Conference Publishing Service, september 2007, pp. 500–506.

[7] OMG, "Production Rule Representation RFP," Object Management Group, Tech. Rep., 2003.

[8] ——, "Semantics of Business Vocabulary and Business Rules (SBVR)," Object Management Group, Tech. Rep. dtc/06-03-02, 2006.

[9] D. Rodríguez-García, E. G. Barriocanal, S. S. Alonso, and C. R.-S. Nuzzi, "Defining software process model constraints with rules using OWL and SWRL," *International Journal of Software Engineering and Knowledge Engineering*, vol. 20, no. 4, pp. 533–548, 2010.

[10] G. J. Nalepa and A. Ligęza, *Software engineering: evolution and emerging technologies*, ser. Frontiers in Artificial Intelligence and Applications. Amsterdam: IOS Press, 2005, vol. 130, ch. Conceptual modelling and automated implementation of rule-based systems, pp. 330–340.

[11] A. Ligęza, *Logical Foundations for Rule-Based Systems*. Berlin, Heidelberg: Springer-Verlag, 2006.

[12] R. M. Dijkman, M. Dumas, and C. Ouyang, "Formal semantics and automated analysis of BPMN process models. preprint 7115," Queensland University of Technology, Brisbane, Australia, Tech. Rep., 2007.

[13] C. Ouyang, M. D. Wil M.P. van der Aalst, and A. H. ter Hofstede, "Translating BPMN to BPEL," Faculty of Information Technology, Queensland University of Technology, GPO Box 2434, Brisbane QLD 4001, Australia Department of Technology Management, Eindhoven University of Technolog y, GPO Box 513, NL-5600 MB, The Netherlands, Tech. Rep., 2006.

[14] F. Coenen *etal*, "Validation and verification of knowledge-based systems: report on eurovav99," *The Knowledge Engineering Review*, vol. 15, no. 2, pp. 187–196, 2000.

[15] A. Ligęza, "Toward logical analysis of tabular rule-based systems," *International Journal of Intelligent Systems*, vol. 16, no. 3, pp. 333–360, 2001, special issue on Verification and Validation Issues in Databases, Knowledge-Based Systems, and Ontologies.

[16] ——, "BPMN – a logical model and property analysis," *Decision Making in Manufacturing and Services*, vol. 5, no. 1-2, pp. 57–67, 2011.

[17] G. J. Nalepa and A. Ligęza, "HeKatE methodology, hybrid engineering of intelligent systems," *International Journal of Applied Mathematics and Computer Science*, vol. 20, no. 1, pp. 35–53, March 2010.

[18] A. Ligęza and G. J. Nalepa, "A study of methodological issues in design and development of rule-based systems: proposal of a new approach," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 2, pp. 117–137, 2011. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/widm.11/pdf

[19] G. J. Nalepa, *Semantic Knowledge Engineering. A Rule-Based Approach*. Kraków: Wydawnictwa AGH, 2011.

[20] M. Negnevitsky, *Artificial Intelligence. A Guide to Intelligent Systems*. Harlow, England; London; New York: Addison-Wesley, 2002, iSBN 0-201-71159-1.

[21] D. Andročec, "Simulating BPMN models with Prolog," in *Proceedings from the Central European Conference on Information and Intelligent Systems, CECIIS – 2010*, 2010, pp. 363–368.

[22] V. S. W. Lam, "Equivalences of BPMN processes," *Service Oriented Computing and Applications*, vol. 3, no. 3, pp. 189–204, 2009.

[23] B. Andersson, M. Bergholtz, A. Edirisuriya, T. Ilayperuma, and P. Johannesson, "A declarative foundation of process models," in *Advanced Information Systems Engineering*, ser. Lecture Notes in Computer Science, O. Pastor and J. FalcÃčo e Cunha, Eds. Springer Berlin / Heidelberg, 2005, vol. 3520, pp. 339–377.

[24] P. Y. H. Wong and J. Gibbons, "A process semantics for bpmn," in *Proceedings from the 10th International Conference on Formal Engineering Methods, ICFEM 2008, Kitakyushu-City, Japan, October 27-31, 2008*, ser. Lecture Notes in Computer Science, S. Liu, T. S. E. Maibaum, and K. Araki, Eds., vol. 5256. Springer, 2008, pp. 355–374.

[25] R. M. Dijkman and P. V. Gorp, "Bpmn 2.0 execution semantics formalized as graph rewrite rules," in *Proceedings from the Business Process Modeling Notation – Second International Workshop, BPMN 2010, Potsdam, Germany, October 13-14, 2010*, ser. Lecture Notes in Business Information Processing, J. Mendling, M. Weidlich, and M. Weske, Eds., vol. 67. Springer, 2011, pp. 16–30.

[26] V. Gruhn and R. Laue, "Checking properties of business process models with logic programming," in *Proceedings of the 5th International Workshop on Modelling, Simulation, Verification and Validation of Enterprise Information Systems, MSVVEIS-2007, In conjunction with ICEIS 2007, Funchal, Madeira, Portugal, June 2007*, J. C. Augusto, J. Barjis, and U. Ultes-Nitsche, Eds. INSTICC PRESS, 2007, pp. 84–93.

[27] V. S. W. Lam, "Formal analysis of BPMN models: a NuSMV-based approach," *International Journal of Software Engineering and Knowledge Engineering*, vol. 20, no. 7, pp. 987–1023, 2010.

[28] K. Kluza, T. Maślanka, G. J. Nalepa, and A. Ligęza, "Proposal of representing BPMN diagrams with XTT2-based business rules," in *Intelligent Distributed Computing V. Proceedings of the 5th International Symposium on Intelligent Distributed Computing – IDC 2011, Delft, the Netherlands – October 2011*, ser. Studies in Computational Intelligence, F. M. Brazier, K. Nieuwenhuis, G. Pavlin, M. Warnier, and C. Badica, Eds. Springer-Verlag, 2011, vol. 382, pp. 243–248. [Online]. Available: http://www.springerlink.com/content/d44n334p05772263/