International Journal of Software Innovation
Volume 8 • Issue 1 • January-March 2020

An Approach for the Transformation and Verification of BPMN Models to Colored Petri Nets Models

Said Meghzili, MISC Laboratory, Computer Science Department, University of Constantine 2-Abdelhamid Mehri, Constantine, Algeria

Allaoua Chaoui, University of Constantine 2, Constantine, Algeria

Martin Strecker, IRIT, Toulouse University, Toulouse, France

Elhillali Kerkouche, Mohammed Seddik Ben Yahia University, Jijel, Algeria

ABSTRACT

The correctness of transformations has recently begun to attract the attention of the researchers in Model Driven Engineering (MDE). The objective of this article is twofold. First, it presents an approach for transforming BPMN models to Colored Petri nets models using GROOVE and EMF/Xpand tools. Second, it proposes an approach for checking the correctness of the transformation itself. More precisely, we have defined the termination property of the transformation and the preservation of some structural properties of BPMN models by the transformation using the GROOVE graph transformation tool. The authors have also applied the approach on a case study through which the authors have verified the successful termination of the transformation using GROOVE Model Checker and the target model properties using CPN Tools.

KEYWORDS

BPMN, CPN, Formal Verification, Graph Transformation, Model Checker, Model-Driven Engineering, Termination, Transformation Correctness

INTRODUCTION

Business Process Modeling Notation (BPMN) (Kopp et al., 2012) is a language that models the steps of a planned business process in industry. It is an efficient language for designing and representing the complex behavior of systems such as synchronization, parallelism, and choice. In addition, BPMN can be used to orchestrate the operational behavior like deployment in TOSCA (Topology and Orchestration Specification for Cloud Application). TOSCA is a way to describe the structure of portable services in a topology and their management as workflows called plans (Lipton et al., 2017). Its goal is to decrease costs of applications, especially in the area of cloud computing. However, BPMN suffers from syntactic and semantic inconsistencies, which requires the verification and the validation of the resulting process models. More precisely, BPMN has no explicitly formalized semantics and suffers from lack of analysis and verification capabilities. For example, it cannot detect the flow control

DOI: 10.4018/IJSI.2020010102

errors and behavioral inconsistencies. To this end, several studies have transformed BPMN into formal models like Petri nets or process algebras. The goal of this transformation is to take benefit from the fact that these models have a formal semantics and offer several analytical techniques.

Most work in the field of MDE (Model Driven Engineering) (Da Silva, 2015), especially model transformation, concentrates on defining the rules of the transformation while a little attention has been dedicated to the verification of the transformation itself. Probably, the verification of model transformations is a difficult task and needs many efforts in order to improve the quality and the reliability of the products developed using this paradigm (Calegari & Szasz, 2013). Checking a transformation consists of proving that its desired and undesired properties are correct. These properties include confluence, termination, semantic correctness, and syntactic correctness. The most known techniques that can be used for the verification of the correctness of such system are theorem provers and model checkers. A model checker verifies that a property holds for a system by exploring the set of all its reachable states. In the case of verification failure, the model checker gives a counter example (Varró, 2004). However, it suffers from the well-known state space explosion problem. Contrary to the model checker-based approach, theorem proving can be used to specify and prove the correctness of complex systems with infinite state space. Nevertheless, a theorem prover has much cost, requires advanced proofs skills, and sometimes needs user interaction for constructing a proof.

In his paper, we propose a transformation of BPMN models to Colored Petri Nets (CPN) models using GROOVE and EMF/Xpand tools and define the termination property of the transformation as well as the preservation of some structural properties of BPMN models by the transformation using GROOVE graph transformation tool. Then, we apply the approach on a case study through which we verify the successful termination of the transformation using GROOVE Model Checker and the target model properties using CPN Tools.

The rest of this paper is organized as follows. In section 2, we discuss related work. In section 3, we present the concepts and background of this work. In section 4, we present the contributions of this paper: we describe our approach that transforms BPMN to CPN models, then we define the termination of the transformation itself and we propose some structural properties of BPMN models and their corresponding models in CPN using GROOVE. In section 5, we validate our approach using a case study that contains a structural error. The final section concludes the paper and gives some perspectives.

RELATED WORKS

The BPMN notation is a widely used and largely studied standard for the specification of business processes in an organization. To extract executable specifications from BPMN models in a way suitable for formal verification, one of the easiest ways is to use model transformation.

Many works tackled the problem of formalizing BPMN models through translating it to formal standards supported by analysis facilities. In (Rachdi et al., 2016), the authors define formal semantics of BPMN in terms of a mapping to Time Petri Nets and verify some desired properties. In (Kherbouche et al., 2013), the authors map the BPMN models to Kripke structure, and then check the validity of major properties (e.g. absence of deadlocks, livelocks, and multiple terminations). In (Meghzili et al., 2016), the authors proposed an approach for transforming BPMN models to ordinary Petri Nets models. In (Kheldoun et al., 2017), the authors propose an approach to analyze BPMN models using Maude and its LTL model checker. This approach is consisting of two steps. In the first step, it transforms the BPMN models into recursive ECATNets models using ATL tool. In the second step, it generates Maude specification from the recursive ECATNets models using Aceeleo tool. In (Dijkman et al., 2008), the authors define formal semantics of BPMN in terms a mapping to Petri nets. This output Petri net model is represented in the PNML (Petri Nets Markup language) language, which can be used to verify BPMN models using its analytical techniques. In (Mahdi et al., 2012), the authors use the meta-modeling and graph grammars to generate colored Petri Nets