Information Systems 56 (2016) 326–346

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/infosys

Property specification, process verification, and reporting – A case study with vehicle-commissioning processes

Richard Mrasek [a,*], Jutta Mülle [a], Klemens Böhm [a], Michael Becker [b], Christian Allmann [b]

[a] Karlsruhe Institute of Technology (KIT), Institute for Program Structures and Data Organization, 76131 Karlsruhe, Germany
[b] AUDI AG, 85045 Ingolstadt, Germany

ARTICLE INFO

Article history:
Received 28 November 2014
Received in revised form
27 August 2015
Accepted 7 September 2015
Available online 26 September 2015

Keywords:
Property specification
Business process management
Workflow management
Verification
Model checking
Petri net
Industrial processes
Vehicle commissioning processes

ABSTRACT

Testing in the automotive industry is supposed to guarantee that vehicles are shipped without any flaw. Respective processes are complex, due to the variety of components and electronic devices in modern vehicles. To achieve error-free processes, their formal analysis is required. Specifying and maintaining properties the processes must satisfy in a user-friendly way is a core requirement on any verification system. We have observed that there are few property templates that testing processes must adhere to, and we describe these templates. They depend on the context of the processes, e.g., the components of the vehicle or testing stations. We have developed a framework that instantiates the templates of properties at verification time and then verifies the process against these instances. To allow an automatic verification we develop a transformation of the commissioning process to a Petri net. Using a novel approach, we are able to report the found violations to the user in a user-friendly way. Our empirical evaluation with the industrial partner has shown that our framework does detect property violations in processes. From expert interviews we conclude that our framework is user-friendly and well suited to operate in a real production environment.

1. Introduction

The systematic testing and configuration of complex products, e.g., vehicles, is an important part of a production process. For testing and configuring, certain tasks need to be executed, automatically or with the help of a human. So-called commissioning tasks test a component or put it into service, e.g., configure the software [1]. Workflows called commissioning processes describe the arrangement of these tasks. The scenario dealt with in this article is one where domain experts from industry develop the commissioning processes. Workflow management systems (WfMS) in the production domain that coordinate the testing and end-of-line manufacturing of the items produced are referred to as diagnostic frameworks.

Our overall goal is to verify if a given commissioning process is correct. This means checking that it fulfills certain properties that are given. This is in contrast to validation, which is not at a formal level, but relies on the intuition of the users to ensure that a process meets their needs and is useful. As just mentioned, for verification it is necessary to specify properties. We have collected such properties in cooperation with domain experts from

* Corresponding author.
E-mail addresses: richard.mrasek@kit.edu (R. Mrasek), jutta.muelle@kit.edu (J. Mülle), klemens.boehm@kit.edu (K. Böhm), michael1.becker@audi.de (M. Becker), christian.allmann@audi.de (C. Allmann).

http://dx.doi.org/10.1016/j.is.2015.09.005
0306-4379/

industry by analyzing existing processes, and by closely observing these experts when designing processes.

Example 1. Some tasks use a specific resource with a limited capacity. A property is that a resource must not be accessed more often than the maximum capacity. The consequence of the violation would be that the process must halt. In this case process execution time is unnecessarily long.

A common definition of correctness of a process is that it fulfills all properties required. Recent research [2,3] has shown, and we have observed this as well, that process models do not always comply with all properties required. Properties typically are formulated as property rules, which are similar to compliance rules [4,5]. For example, a property rule states that before executing Task X another Task Y has to be executed.

Verification is itself a process that consists of several phases, namely specifying the properties of the commissioning process, verifying them, and presenting the results to the users. Our concern is the design and realization of a framework supporting users throughout this entire process. This gives way to the following questions. First, how must processes as well as the properties be specified to facilitate the deployment of verification techniques? Second, how to utilize domain information to support the users specifying the formal properties? Finally, how user-friendly are respective solutions? To verify process models given in a formal representation like Petri nets against properties, there already exist efficient model checking approaches [6,7]. However, deriving and specifying the properties the model must satisfy is another issue. A core question is how a user-friendly framework for process verification should look like.

Designing such a framework gives way to several challenges: First, the knowledge on which characteristics an industrial process should fulfill is typically distributed among several employees in different departments. Often a documentation is missing, and properties merely exist in the minds of the process modelers. Second, the properties frequently are context-sensitive, i.e., only hold in specific contexts of a commissioning process. For example, some tasks need different protocols to communicate with control units for testing at different factories. Due to this context-sensitiveness, the number of properties is very large, but with many variants with only small differences. This causes maintenance problems [8]. For instance, an average process model from our use case has to comply with 39 properties. The properties and process model are constantly being revised. This leads to serious maintenance problems. Third, to apply an automatic verification technique like model checking, it is necessary to specify the properties in a formal language such as a temporal logic [9]. With vehicle-commissioning processes as well as in other domains, see, for instance [10,11], specifying the properties in this way is error-prone and generally infeasible for domain experts who are not used to formal specification. To facilitate an automatic verification, the process must be formalized in a notation that allows to directly construct its state space. To this end, it must be easy to let the properties refer to the processes modeled. Fourth, it is challenging to present the violations found to the user in a way that is both succinct and understandable. Fifth, evaluating an approach such as the one envisioned is difficult. One issue is that the evaluation criteria must be specified.

We have addressed these challenges based on the real-world use case of vehicle-commissioning processes. More specifically, we make the following contributions: We have analyzed which properties occur for vehicle-commissioning processes and the respective context information. We have observed that there are few templates these properties adhere to. We propose to explicitly represent these templates, rather than each individual property. Next, we develop a model of the context knowledge regarding vehicle-commissioning processes. Here *context*, is the components of a vehicle, their relationships and the constraints which the vehicle currently tested and configured must fulfill. We let a relational database manage the context information. To populate it, we use several sources, e.g., information on the vehicle components from production planning, constraints from existing commissioning processes, and information provided by the process designers themselves.

Our framework uses this information to generate process-specific instances of the property templates, transforms the process models to a Petri net, and verifies the models against these properties, see Fig. 1. For the verification we rely on a transformation of the notation OTX for commissioning processes to Petri nets which we have developed ourselves. Our framework is able to interpret the verification results and the characteristics of the process that most likely are responsible for a property violation. The tool uses the verification result to highlight the important elements in a visualization of the process. We use a template-specific approach whose output is more concise than the one of a generic solution. Our evaluation has shown that the framework as a whole does detect rule

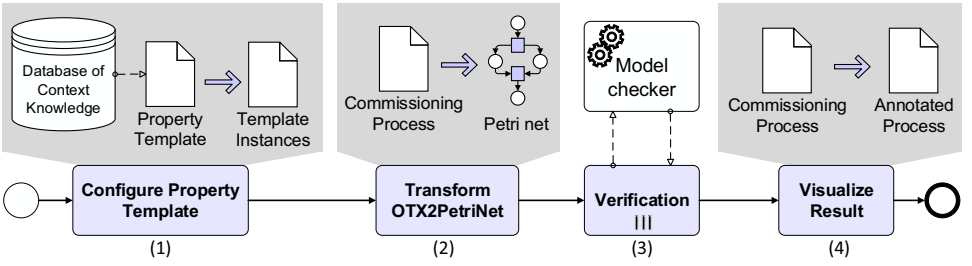

Fig. 1. Steps of the verification framework.