Process Verification and Synthesis – The Use Case of Commissioning Processes in the Automobile Industry

Richard Mrasek

Karlsruhe Institute of Technology (KIT)
Institute for Program Structures and Data Organization
76131 Karlsruhe, Germany
richard.mrasek@kit.edu

Supervisor: Prof. Dr.-Ing. Klemens Böhm

Abstract. In the automobile industry, commissioning process models describe the end-of-line manufacturing and testing of vehicles. Due to the increase of electronic components in modern vehicles, the process models tend to become more complex. At the same the number of different model series is constantly increasing leading to a larger amount of process models. The increase in process models and complexity lead to higher cost for the process design and decrease the quality of the individual process model. In this Ph.D. project we want to support process modeling. *First*, by developing a framework to test if a given process model fulfills all properties required (*process verification*). *Second*, we want to support the process design by approaches for a semiautomatic generation of process models (*process synthesis*). *Third*, for *process verification* and *process synthesis* one needs a *specification* of the allowed behavior of the process models.

Keywords: Commissioning Processes, Process Verification, Process Synthesis, Business Process Modeling

1 Introduction

In the automobile industry, commissioning process models describe the end-of-line manufacturing and testing of vehicles. Process developers define these processes with development tools. Workflow Management Systems (WfMS), here referred to as Diagnostic Frameworks, execute these processes [25]. Vehicle commissioning includes, say, to check for each vehicle produced, whether all its Electronic Control Units (ECU) are integrated correctly and to put them into service. ECUs are components built in to the vehicle which control specific functionalities of the car, e. g., the ECU MOT controls the engine electronics. Each ECU needs to be tested and put into operation, e. g., by installing certain software. To this end, the WfMS executes several tasks for each ECU. Tasks can be executed automatically like the configuration of the control unit, or they may require a factory worker equipped with a hand terminal. Figure 1 shows the general architecture of a

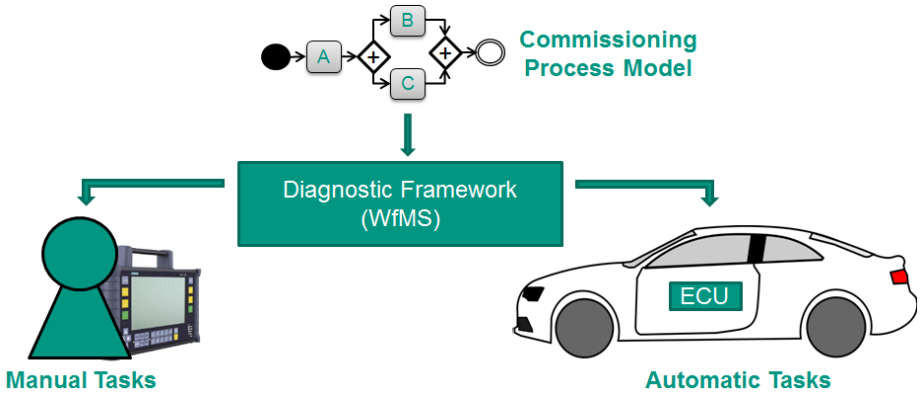

Fig. 1. The Simplified Architecture of a Diagnostic System

diagnostic system. Commissioning processes have the characteristic to be complex. Typically there are hundreds of tasks for each vehicle, arranged in up to 14 parallel lanes.

Due to the increase of electronic components in modern vehicles, the process models tend to become more complex. At the same the number of different model series is constantly increasing leading to a larger amount of process models. In this Ph.D. project we want to support process modeling, making research on testing schemes, whether a given process model fulfills all properties required (*process verification*), and on approaches for a semiautomatic generation of process models (*process synthesis*). For *process verification* and *process synthesis* one need *specification* of the allowed behavior of the process models. Formally, let P be the process model of a commissioning process, and $\mathcal{L}_P$ denote the complete log of the process, i. e., all possible traces of the process model. Let $\mathcal{C}$ denote the set of all traces allowed by the properties. We can now define *Specification*, *Verification*, and *Synthesis* as follows:

$$\textit{Specification} \ : \ \text{Define the set of allowed traces } \mathcal{C} \tag{1}$$

$$\textit{Verification} \ : \ \text{For a given process model } P \text{ check if } \mathcal{L}_P \subseteq \mathcal{C} \tag{2}$$

$$\textit{Synthesis} \ : \ \text{Generate a process model } P \text{ with } \mathcal{L}_P \subseteq \mathcal{C} \tag{3}$$

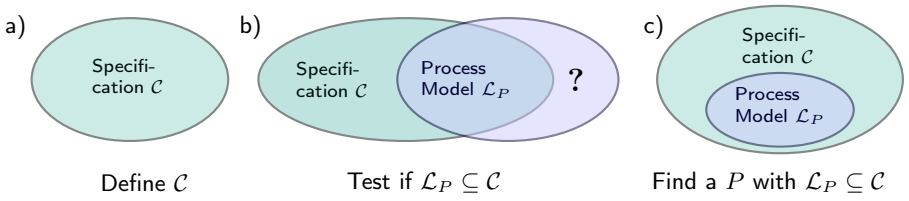

Fig. 2. Our Problem Statements Specification (a), Verification (b), and Synthesis (c).

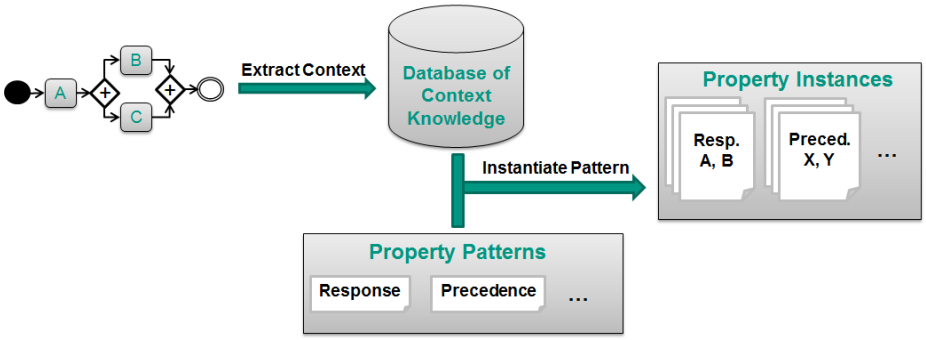

Fig. 3. The Approach of the Instantiating of the Contextual Property Pattern

2 Specification

Before *verification* and *synthesis* can take place we have to *specify* the allowed behavior $\mathcal{C}$. The allowed behavior is induced by a set of properties Φ. The allowed behavior $\mathcal{C}$ consists of all traces fulfilling the properties Φ, i. e., $\mathcal{C} := \{t \mid \forall \phi \in \Phi : t \models \phi\}$. The specification of the properties Φ gives way to several challenges: *First*, the knowledge which characteristics a commissioning process should fulfill is typically distributed among several employees in different departments. Often documentation is missing and properties merely exist in the minds of the process modelers. *Second*, the properties frequently are context-sensitive, i. e., they only hold in specific contexts of a commissioning process. For example, certain tasks require a protocol to communicate with control units for testing depending on the factory the testing takes place. Due to this context-sensitiveness, the number of properties is very large, but it consists of a lot of variants with only small differences. This causes maintenance problems [11]. For instance the new generation of an electronic control unit in the car uses a different communication protocol than the previous generation. This protocol change leads to a large set of new properties and render several properties invalid. *Third*, to apply an automatic verification or synthesis technique, it is necessary to specify the properties in a formal language such as a temporal logic [21]. With vehicle-commissioning processes as well as in other domains, see for instance [6], [15], specifying the properties in this way is error-prone and generally infeasible for domain experts who are not used to formal specifications.

Research Question:

How to generate the correct set of properties given a process model and context.

Approach:

In [19] we have presented an approach to address these challenges based on our real-world use case of vehicle-commissioning processes. More specifically, we use

the following approach: We have analyzed the properties occur for vehicle commissioning processes as well as the respective context information. We have observed that there are few patterns to which these properties adhere to. We propose to explicitly represent these patterns, rather than each individual property. Next, we develop a model of the context knowledge regarding vehicle-commissioning processes. Here, *context* are the components of a vehicle, their relationships and the constraints which the vehicle actually 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 patterns. Figure 3 illustrates the approach.

Evaluation:

Our goal is for a user-friendly approach for the specification of properties, i. e., high usability. According to ISO 9241-11 [10] usability has three different aspects to be evaluated separately: *Effectiveness* (Whether the user can complete his tasks and achieve the goals), *Efficiency* (The amount of the resource usage to achieve the goals), *Satisfaction* (The level of comfort the users experience achieving the goals). The *effectiveness* is proven by testing if the resulting specification gives a meaningful result for the later verification or synthesis. Our approach has shown to be able to generate the hundreds of property instances in under one second, proving a high *efficiency*. For the *satisfaction* we have used an established questionnaire the System Usability Scale (SUS). The result states that our approach leads to a high satisfaction with results higher than the average.

3 Verification of Process Models

Verification means to test if the behavior of the process model $\mathcal{L}_P$ complies with the allowed behavior $\mathcal{C}$. The verification is not trivial because, it is not possible to explicitly generate $\mathcal{C}$ and $\mathcal{L}_P$. $\mathcal{C}$ is in general not bounded and the size $\mathcal{L}_P$ can increase exponentially with the size of the process model, or it can even be infinite. This is well known as state-space explosion [4]. It leads to unacceptable runtime or renders the verification not executable. This is often caused by parallel branches in the model. To overcome this problem, reduction techniques can be used, either (a) during construction of the $\mathcal{L}_P$ or (b) on the level of the process model already. Approaches like stubborn set reductions [22] fall into the first category. However, many of the industrial processes to be analyzed in our evaluation are too large to be verified only with stubborn set reductions. Even with stubborn set reduction, there are more than 1 million traces in 78% of the processes we have evaluated; thus, verification has not been possible in reasonable time. Regarding (b), only few proposals exist, although preprocessing of the process model is promising to achieve a significant reduction of the state

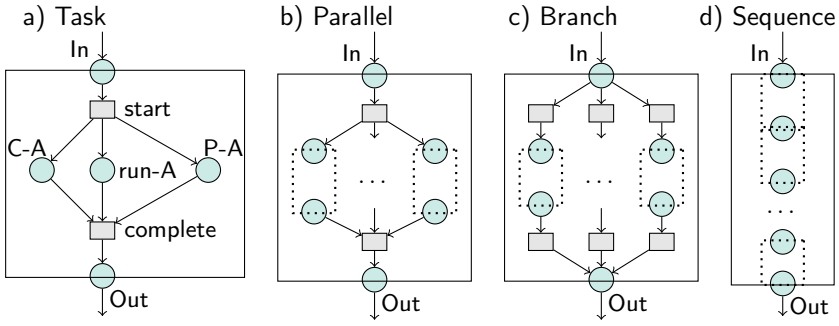

Fig. 4. The Simplified Templates for Different OTX Elements, from [19].

space. An example is given in [2]. They specify the requirements in BPMN-Q. BPMN-Q is a visual language to query business process models. [2] however is not expressive enough to express all requirements from our real-world application scenario. Furthermore, they apply reduction rules on the process schema in an iterative way. After each reduction step, another reduction rule may become again applicable. Thus, a rescan of the whole process may be necessary after each step, rendering this kind of approach expensive. In the industrial setting envisioned here, it is necessary to verify hundreds of properties per process, in short time. Compared to the processes dealt with by others [7], ours are much larger and more complex, leading to an exploding state space.

Research Questions:

There are two research question of our concerns. First, how to allow the verification for the industry standard OTX. Second, how to verify a process model having a state space to large to generate explicitly.

Approach:

In order to allow the verification one have to generate the traces $\mathcal{L}_P$ for a process model. It is not possible to directly generate $\mathcal{L}_P$ for a commissioning process in the notation of OTX. Therefore, we developed a mapping of OTX to Petri net suitable for our verification. For the transformation we define for each object in OTX a Petri net subnet. For the transformation we parse the process model of a OTX process model and generate a Petri net according to the templates. Figure 4 shows the simplified template for four OTX elements.

The verification of our real commissioning processes has performance issues. Verification means to check if a process model P complies with all required properties Φ, formally to check $\forall \phi \in \Phi : P \models \phi$. Our basic idea is to generate

a smaller process model P_ϕ for each property ϕ. The reduction should preserve each property, or formally:

$$\forall \phi \in \Phi : P \models \phi \Leftrightarrow \forall \phi \in \Phi : P_\phi \models \phi$$

In [16] we showed an algorithm that traverses the process model and identifies the regions of the process that are relevant for verification of a given complex property ϕ. Identifying the relevant regions of a process is far from trivial. Even an elementary task cannot be removed in all cases. Our approach features a criterion for process-graph reduction, which we refer to as relevance function. The algorithm proposed creates a formal, reduced representation of the process for each property. In particular, the reduction of parallel regions help to decrease the size of the state space and hence the runtime of the verification.

Evaluation

The approach has been evaluated with commissioning process models for testing newly produced vehicles in the factories of a German car manufacturer. One result is that even complex processes with many parallel branches can be verified in less than 10 seconds on a commodity PC. Our approach is able to detect property violations in realistic commissioning processes.

4 Process Syntheses

Process Syntheses is to find a process model P with the complete log $\mathcal{L}_P \subseteq \mathcal{C}$. To allow a transformation into OTX we are looking for a block-based process model. As we show in [18] it is in general not possible to find a block-based process model with $\mathcal{L}_P = \mathcal{C}$. Furthermore, as we see in our use case, it is not possible to find a single best process model P. In general, a vast amount of process models is possible.

Research Question:

How to synthesize an acyclic process model from a declarative specification that is good according to a given quality criteria.

Approach:

The approach presented later at the conference [18] generates a process model from a declarative specification. The input to our approach is a declarative specification in graph forms the Ordering Relationship Graph (org). In [17] we show how to generate such a graph from other specification languages. First we apply a modular decomposition on the graph. The technique decomposes the graph in several subgraph of different granularity. The subgraphs called modules are arranged in a hierarchical form, called Modular Decomposition Tree (MDT). The modular decomposition allows us to detect the under-specified parts of the specification. We use a probabilistic search to find a good solution for the under-specified regions according to a predefined fitness function.

Evaluation:

As we show in the evaluation with thousands of non-trivial process models, our approach is efficient, i. e., is able to test thousands of models in under a second. We use a real life specification for commissioning from our industrial partner in our evaluation. On average, our approach nearly halves the processing time compared to the reference processes which already are the output of a careful, intellectual design. It is able to handle complex real-world specifications containing several hundred dependencies as well as more than one hundred tasks. In our evaluation, the process models generated contain between 98 and 185 tasks, and their arrangement typically is nontrivial.

5 Related Work

Recent research works present different graphical notations for the property specification, e. g., for the verification. See, for example, the Compliance Rule Graphs (CRG)[15], or BPMN-Q [2]. The graphical specification allows for a more user-friendly and intuitive specification compared to the textual specification, say, in a temporal logic. But they do not support the major challenges of our work: The context sensitivity and the distributed knowledge. [6] introduce a set of property patterns for the specification. They share some common patterns with our set of commissioning property patterns but lack necessary domain specific information.

A related field of research is business process compliance [13]. Compliance is ensuring that a process model is in accordance with prescribed norms [20], e. g., Sarbanes-Oxley, Basel II, HIP AA. In general, two approaches toward process compliance exists. Expensive manual checks (after-the-fact) and automated detection. For the automated detection the norms have to be specified formally. As well as in our use case, specifying the norm leads to maintenance problems [14]. Approaches exist to ensure the compliance of an existing process model by, e. g., model checking [2][12][8] or to synthesize a new process model which complies with the norm [3][9].

A lot of work is done in the verification of the soundness property for process models. The soundness verification leads to a similar state space explosion compared to our approach. [1][5] tries to handle the state space explosion by using reduction rules on the process-model level. These reduction rules are not applicable for our use case, in general. Other works like the stubborn set reduction [23, 22], try to reduce the state-space generation of a Petri net. These techniques are orthogonal to our reduction and can be used in combination. Our experiments have shown that these low level reductions alone are not sufficient for our process models.

[24] and [3] synthesize a process model from a declarative specification. To this end, [24] uses a collection of small state machines representing property patterns, and [3] from LTL formulas. The approach of [24] does not consider the case of a under specification, i. e., more than one process model is possible for

the specification. [3] requires a manual solving of these cases. Both approaches indicate performance issues when dealing with large specifications like the ones in our use case.

6 Conclusions

In this Ph.D. project we research the *specification*, *verification* and *synthesis* of commissioning process models in the automobile industry. The verification frameworks are quite mature and actually applied in the factory of our industry partner. The framework has shown to be able to increase the quality of the process models. The first results of the *synthesis* show a great potential in applying these techniques. In the last year of these project we plan to apply the *synthesis* technique for the design of the new process models for the next generation of vehicles.

References

1. W. M. P. v. d. Aalst, A. Hirnschall, and H. M. W. Verbeek. "An Alternative Way to Analyze Workflow Graphs". In: *Advanced Information Systems Engineering*. 2002.
2. A. Awad, G. Decker, and M. Weske. "Efficient Compliance Checking Using BPMN-Q and Temporal Logic". In: *Business Process Management*. 2008.
3. A. Awad et al. "An Iterative Approach for Business Process Template Synthesis from Compliance Rules". In: *Advanced Information Systems Engineering*. 2011.
4. E. M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. 1999.
5. B. F. van Dongen, W. M. P. v. d. Aalst, and H. M. W. Verbeek. "Verification of EPCs: Using Reduction Rules and Petri Nets". In: *Advanced Information Systems Engineering*. 2005.
6. M. B. Dwyer, G. S. Avrunin, and J. C. Corbett. "Property Specification Patterns for Finite-state Verification". In: *Proceedings of the Second Workshop on Formal Methods in Software Practice*. 1998.
7. D. Fahland et al. "Instantaneous Soundness Checking of Industrial Business Process Models". In: *Business Process Management*. 2009.
8. A. Forster et al. "Verification of Business Process Quality Constraints Based on Visual Process Patterns". In: *First Joint IEEE/IFIP Symposium on Theoretical Aspects of Software Engineering, 2007. TASE '07*. 2007.
9. S. Goedertier and J. Vanthienen. "Designing Compliant Business Processes with Obligations and Permissions". In: *Business Process Management Workshops*. 2006.
10. *ISO 9241-11, Ergonomics of Human-Computer Interaction - Part 11: Guidance on Useability*. 1998.
11. S. Kabicher, S. Rinderle-Ma, and L. T. Ly. "Activity-Oriented Clustering Techniques in Large Process and Compliance Rule Repositories". In: *Proc. BPM'11 Workshops*. 2011.
12. Y. Liu, S. Müller, and K. Xu. "A static compliance-checking framework for business process models". In: *IBM Systems Journal* (2007).
13. L. T. Ly. *SeaFlows - a compliance checking framework for supporting the process lifecycle [Elektronische Ressource] / Linh Thao Ly*. 2013.

14. L. T. Ly et al. “Compliance of Semantic Constraints - A Requirements Analysis for Process Management Systems”. In: 2008.
15. L. T. Ly et al. “SeaFlows Toolset – Compliance Verification Made Easy for Process-Aware Information Systems”. In: *Information Systems Evolution*. 2011.
16. R. Mrasek, J. Mülle, and K. Böhm. “A new verification technique for large processes based on identification of relevant tasks”. In: *Information Systems* (2014).
17. R. Mrasek, J. Mülle, and K. Böhm. *Automatic Generation of Optimized Process Models from Declarative Specifications*. Technical Report 2014-15. KIT Scientific Publishing, 2014.
18. R. Mrasek, J. Mülle, and K. Böhm. “Automatic Generation of Optimized Process Models from Declarative Specifications”. In: *Advanced Information Systems Engineering*. 2015.
19. R. Mrasek et al. “User-Friendly Property Specification and Process Verification - a Case Study with Vehicle-Commissioning Processes”. In: *Business Process Management*. 2014.
20. S. Sadiq, G. Governatori, and K. Namiri. “Modeling Control Objectives for Business Process Compliance”. In: *Business Process Management*. 2007.
21. H. Schlingloff, A. Martens, and K. Schmidt. “Modeling and Model Checking Web Services”. In: *Electronic Notes in Theoretical Computer Science* (2005).
22. K. Schmidt. “Stubborn Sets for Model Checking the EF/AG Fragment of CTL”. In: *Fundam. Inf.* (2000).
23. K. Schmidt. “Stubborn Sets for Standard Properties”. In: *Application and Theory of Petri Nets 1999*. 1999.
24. J. Yu et al. “Synthesizing Service Composition Models on the Basis of Temporal Business Rules”. In: *Journal of Computer Science and Technology* (2008).
25. W. Zimmermann and R. Schmidgall. *Bussysteme in der Fahrzeugtechnik - Protokolle, Standards und Softwarearchitektur*. 2011.