Automated Adaptation of Business Process Models Through Model Transformations Specifying Business Rules

Roman Popp and Hermann Kaindl

Vienna University of Technology, Vienna, Austria
{roman.popp, hermann.kaindl}@tuwien.ac.at

Abstract. Both business processes and business rules are changing over time. In addition, when business process models try to capture all details, they are complex and need to deal with variability according to context. Manual adaptation of process models takes time and is error-prone.
We address these issues through explicit separation of process models (represented in BPMN) and specifications of business rules (represented as model-transformation rules based on BPMN). Applying such rules as model transformations to a model of a reference process (at design-time) leads to models of either refined or new models of business processes. In this way, both the change of a process per se or the inclusion of a new business rule can be handled through automated adaptation of the related business models, without an extension of the BPMN 2.0 standard.

Keywords: Business process adaptation, Model-driven transformation, BPMN

1 Introduction

Business processes in organizations are changing over time, so their models need to be kept up-to-date. Usually, such models are represented today in Business Process Model and Notation (BPMN), more precisely its version BPMN 2.0, since this language offers a standard for modeling business processes.[1]

Business processes also need to take variability into account, depending on certain conditions. According to, e.g., Hallerbach and Bauer [1], neither conditional branches inside a process nor separate process models are satisfactory solutions for representing process variability, since both can result in redundancies.

Business rules often define such variability, but informally. Manual adaptations of business processes according to business rules are tedious and lead to such redundancies.

Therefore, research approaches (discussed below) propose automated adaptation of reference processes. We present a new approach in this paper based on model-transformation, which is a core technology of the Object Management Group's Model Driven Architecture[2] (MDA). Model-transformation employs rules that transform a

[1] http://www.omg.org/spec/BPMN/2.0/
[2] http://www.omg.org/mda/

given source model to a related target model, according to precisely specified metamodels.

Our new idea for automated adaptation of business processes presented in this paper is to represent certain kinds of business rules as model-transformation rules. When specified precisely in this way, they can be more or less automatically used to adapt business processes represented as models. For BPMN 2.0, a corresponding metamodel exists and is used by our approach. Since we transform from BPMN to BPMN, the same metamodel defines both source and target model in our approach.

We present our new approach with a running example of a business rule in the context of payment and its conditional authorization. We also show how BMPN models can actually be modified automatically through such model transformations.

The remainder of this paper is organized in the following manner. First, we discuss related work. Then we explain specifying a business rule as a model-transformation rule. Based on that, we explain automatically adapting reference processes through model transformations. Finally, we discuss our approach more generally, draw some conclusions and indicate future work.

2 Related Work

In product-line engineering of software or general systems, the core theme is commonality and variability, see e.g., [2] for an approach and its formalization in the context of requirements and systems engineering. Most of this work is based on feature-oriented models as introduced early by Kang et al. [3], which typically distinguish between mandatory and optional features, as well as mutual exclusion and lists of choices. Mannion and Kaindl [2] integrated parameters into such models as well. A product line in these approaches usually encompasses all (pre-)defined variants. We are not aware, however, of any application of these ideas to adaptability of business process models.

In this paper, we deal with business processes modeled in standard BPMN 2.0, and business rules for their adaptation. So, we also focus the further discussion of related work on this topic. Still, let us also mention an earlier approach to achieving business process flexibility with business rules [4]. However, the points of variability in the reference process have to be identified manually.

Hallerbach and Bauer [1] pursue an approach named Provop, where a business process variant constitutes an adjustment of a *reference process*. Either a standard process or the most frequently used process can be taken as the reference process. Provop defines specific change operations (with pre- and post-conditions) for adaptation of the reference process, which may also be grouped into so-called options. It is not so clear, however, how business rules for adapting a reference process can, in general, be formulated in such an approach, so that they are understandable per se.

Milanovic et al. [5] devised a dedicated language for rule-based sub-process selection and workflow composition called rBPMN. It extends BPMN through weaving with the R2ML rule language. Since the resulting models are, in general, not compatible with the BPMN standard, however, the usual tooling for BPMN cannot be directly used with this approach.

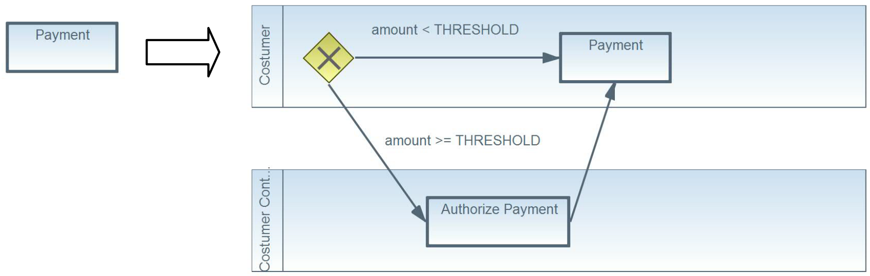

Fig. 1. Business rule example with in-situ substitution

La Rosa and Dumas [6] propose modeling one large reference workflow. Their approach provides design-time configuration for realizing workflow variants, but there is no extensible approach for including business rules.

Döhring et al. [7] defined a new metamodel for so-called vBPMN models resulting from *weaving* BMPN2 and business rules (specified in the R2ML rule language). This approach allows specifying change patterns for application at run-time. However, the reference workflow requires to have adaptive segments explicitly marked. Like rBPMN, vBPMN is an extension of BPMN (as standardized), so that the usual tooling for BPMN cannot be directly used with this approach.

Döhring et al. [8] address this latter problem through design-time model transformations from vBPMN to BPMN2 Sub-Processes (as standardized), where the transformations are hard-coded in Java. Still, the reference model requires adaptive segments to be marked.

In summary, we are not aware of any approach that allows modeling and executing business rules specified directly as model transformations according to MDA, where both source and target model conform to the BPMN 2.0 metamodel.

3 Business Rules Specified as Model Transformation Rules

In our approach, we specify business rules as model transformation rules, as independently as possible from any existing process model as possible. Of course, such a business rule has to refer to something in some existing business process, in order to be useful at all. However, these rule specifications and the process specifications should be maximally decoupled. So, we explain simple examples of such business rules even before referring to any concrete reference process.

Our chosen example deals with payment, a very usual part of typical business processes. Let us assume that payment is to be handled differently in a given business depending on the amount to be paid. If this amount exceeds a defined threshold, another business actor than the one primarily responsible for executing a payment Activity needs to authorize the payment before its execution.

One possibility for specifying this business rule in our approach is illustrated in Fig. 1, which implements an “in-situ” substitution (at the same place) of a given Payment Activity with the conditional payment authorization. In addition to this graphical illustration, such a rule needs to be specified in more detail for making it automatically executable. We specified this rule as a transformation in the ATLAS Transformation Language (ATL).[3] Each ATL rule consists at least of a *from* and a *to* part. The *from* part specifies which element is transformed (possibly with some conditions) and the *to* part specifies the corresponding part in the target model. These parts are based on BPMN in our approach, more precisely BPMN 2.0, where all the technical details have been specified in an XML representation.

We also specified this business rule in different ways, but due to lack of space, we cannot explain them all here. Still, let us mention another formalization for simply transforming a Payment Activity into a BPMN model through a Payment Sub-Process that includes additionally an authorization Activity and conditional Payment. This transformation also needs to create this Sub-Process in BPMN in such a way, that it contains this conditional payment processing.

4 Automatically Adapting Reference Processes through Model Transformations

Based on such specifications of business rules, our approach can automatically adapt reference processes through model transformations. A small example of a reference process represented in BPMN is shown in Fig. 2. It is a simplified version of the "payment handling" process of [9, p. 108]. This payment handling process starts with the creation of an invoice. This invoice is then sent to the customer. After the customer has received the invoice, she makes the payment of the invoice. Concurrently, the delivering company checks the receipt of payment (through its bank). Until the payment is received, the check is repeated periodically. Once the payment is received, it is booked as paid. After that, this example reference process is finished. Note, that payment is unconditional in this process.

This process specification per se does not include any variability. Of course, it could be simply changed manually to include the conditional payment authorization as desired in the business rule example above. This would encode this particular rule in this particular process model, with all the disadvantages already discussed in the literature (see also above).

Much as other approaches, we strive for automated adaptation, but we pursue it through model transformation. After the given business rule as shown in Fig. 1 had been formalized in ATL, applying the ATL rule engine resulted in a process model as shown in Fig. 3. This rule replaces the simple Payment Activity in a reference process with a whole process part directly inside the adapted reference process. In fact, we had to implement several technical work-arounds to make this work because of restrictions of this transformation approach. Due to lack of space, we cannot specify these here.

5 Discussion

It may seem as though it is straight-forward to add more and more business rules as transformation rules according to our approach. However, certain constraints have to

[3] `http://www.eclipse.org/atl/`

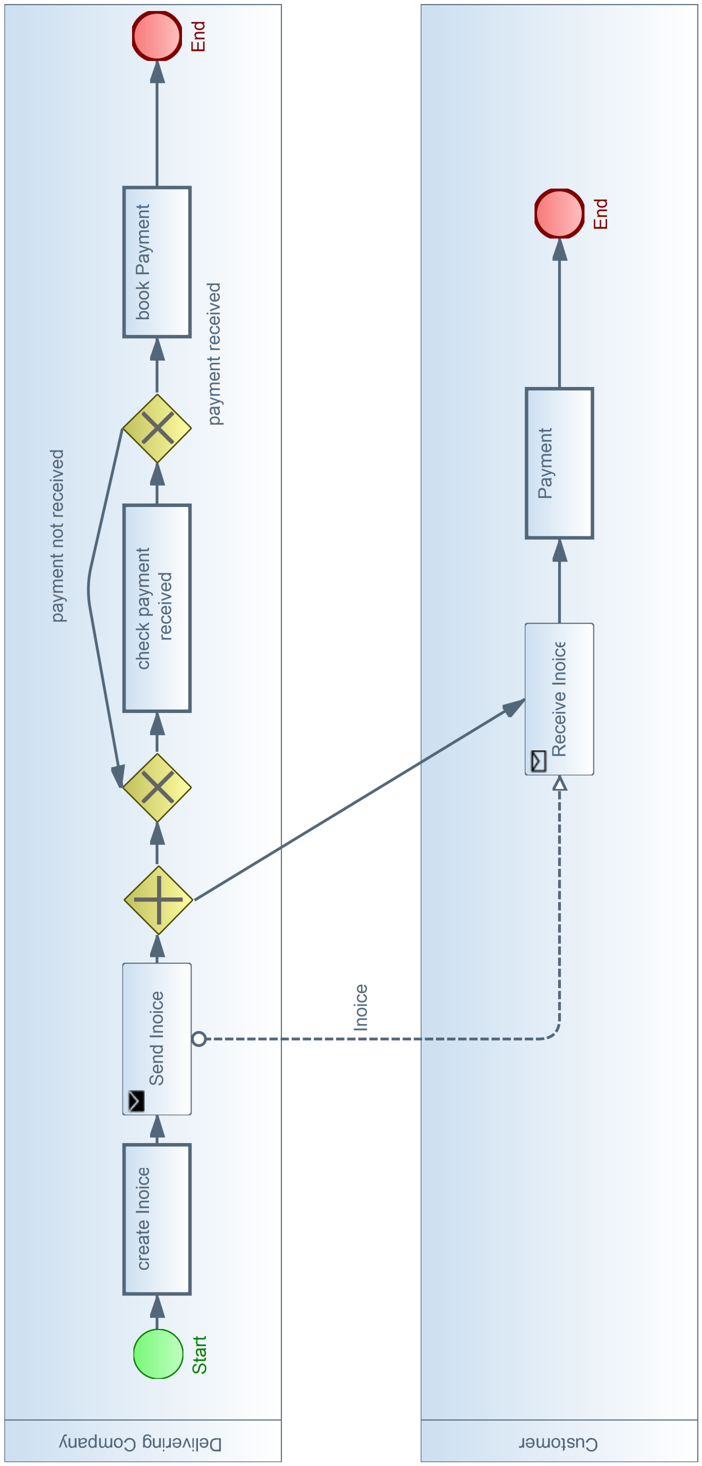

Fig. 2. Reference Process

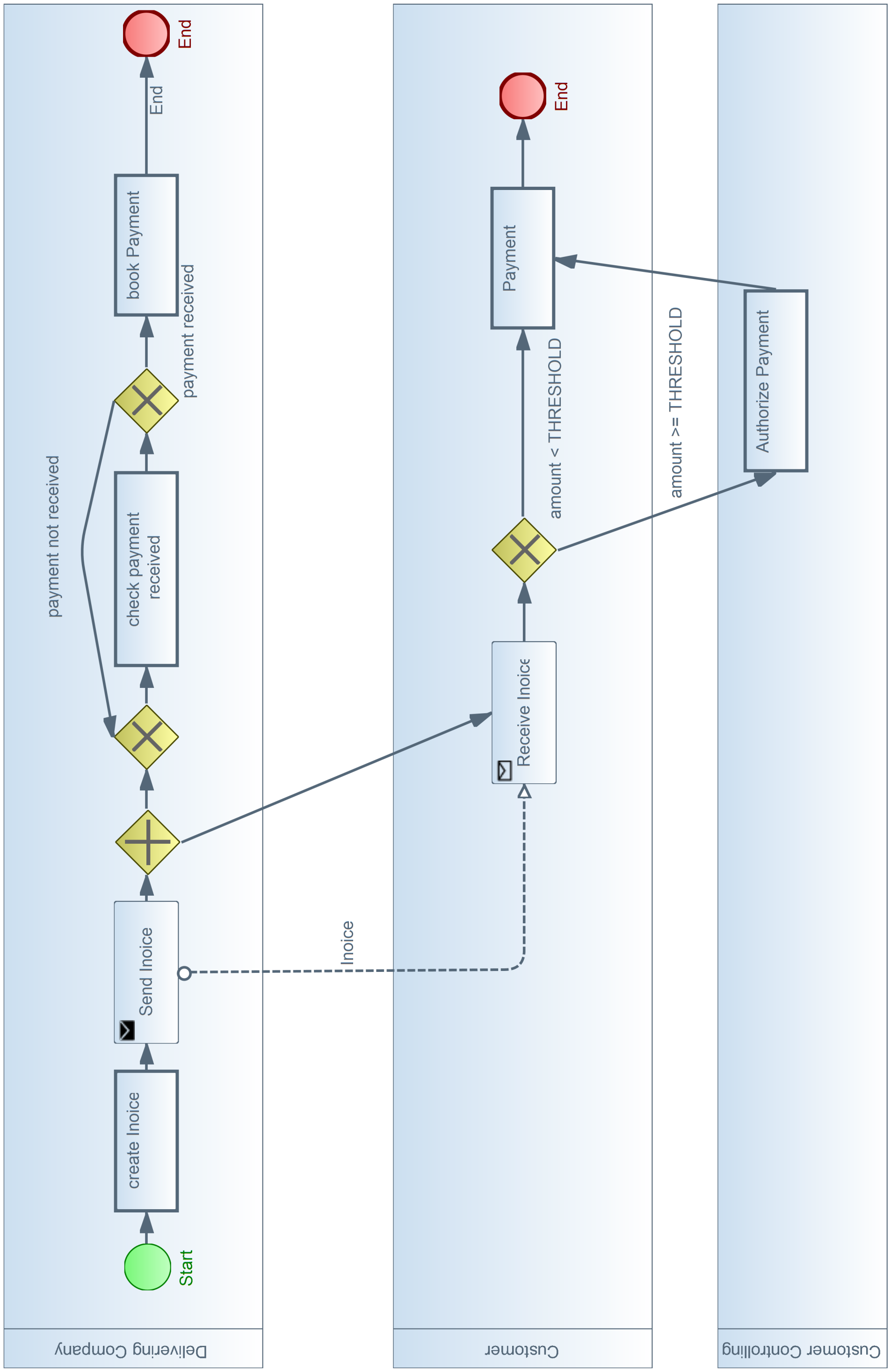

Fig. 3. in-situ Changed Process

be fulfilled, so that the business process resulting through model transformation still conforms to BPMN. For example, it is necessary that the *from* and *to* parts have the same input and output "ports". In our simple example, these are one entry point and one exit point. These constraints will have to be studied yet, possibly along the lines of [10].

Our approach presented in this paper allows strong decoupling of (new) business rules from existing business processes. So almost any existing process can potentially be adapted according to new business rules automatically.

It is also an advantage of our approach that no segments of existing processes need to be marked for adaptation, and so the designer does not need to think of possible adaptions at all, because all the adaptions are fully specified in the business rules.

We only want to modify parts of the models, but the model transformation engines are made for transforming a whole model. This leads to large overhead, like providing generic transformation rules, and because of their triggering restriction we also need the approach of Wagelaar et al. [11] for circumventing this technical problem.

A more convenient solution could be based on a specific transformation engine providing triggering and firing of more than one rule for the same source element. It could be devised along the lines of an existing one as specifically implemented in another context [12, 13], which makes the approach of Wagelaar et al. unnecessary.

6 Conclusion and Future Work

The work presented in this paper addresses adaptability of business process models and, in particular, automating it. We follow the approach to separate reference processes from business rules and propose here the new idea to represent certain kinds of business rules as model transformations.

Our implementation assumes reference processes to be represented as standard BPMN 2.0 models. The *from* and *to* parts of each transformation rule are based on BPMN 2.0 as well, so the process models resulting from such transformations are also standard BPMN. Therefore, existing BPMN execution engines / environments can be used for the resulting process models.

Viewed from a higher perspective, this approach leads to a new form of representing and managing variability of business process models. When specifying reference processes, it is not yet necessary to envisage their variability. In particular, no marking of variable parts is necessary. Vice versa, when specifying business rules in this way, the available reference processes need not be taken into account as a whole or in detail. Just knowing about parts of such processes or sub-processes is required for specifying the business rules.

Still, future work will be necessary for studies with people actually working on adaptable business processes. In order to avoid the technicalities when using currently available transformation engines, creating an advanced one will facilitate the adoption of this approach. Such an engine will not necessarily have to be general enough to handle any metamodel, but support specifically the one of BPMN 2.0. The use of such a (specific) transformation engine will support the specification of business rules as transformation rules.

7 Acknowledgments

Part of this research has been carried out in the ProREUSE project (No. 834167), funded by the Austrian FFG.

References

1. Hallerbach, A., Bauer, T., Reichert, M.: Configuration and management of process variants. In: Intl. Handbook on BPM. Springer Berlin / Heidelberg (2010) 237–255
2. Mannion, M., Kaindl, H.: Using parameters and discriminants for product line requirements. Systems Engineering **11**(1) (2008) 61–80
3. Kang, K., Cohen, S., Hess, J., Novak, W., Peterson, A.: Feature-Oriented Domain Analysis (FODA) feasibility study. Technical report, Carnegie Mellon University, Pittsburgh, PA (November 1990)
4. van Eijndhoven, T., Iacob, M.E., Ponisio, M.: Achieving business process flexibility with business rules. In: Proceedings of the 12th Enterprise Distributed Object Computing Conference (EDOC'08). (Sept 2008) 95–104
5. Milanovic, M., Gasevic, D., Rocha, L.: Modeling flexible business processes with business rule patterns. In: Proceedings of the 15th Enterprise Distributed Object Computing Conference (EDOC'11). (2011) 65–74
6. La Rosa, M., Dumas, M., t. Hofstede, A., Mendling, J.: Configurable multi-perspective business process models. IS **36**(2) (2011)
7. Döhring, M., Zimmermann, B.: vBPMN: Event-Aware Workflow Variants by Weaving BPMN2 and Business Rules. In: Proceedings of the 16th International Conference on Exploring Modeling Methods in Systems Analysis and Design. Volume 81 of LNBIP., Springer (2011) 332–341
8. Döhring, M., Schulz, A., Galkin, I.: Emulating Runtime Workflow Adaptation and Aspect Weaving by Recursive Rule-Based Sub-Process Selection - A Model Transformation Approach. In: Proceedings of the 16th Enterprise Distributed Object Computing Conference (EDOC'12), Beijing, IEEE (2012)
9. Schumacher, J., Meyer, M.: Customer Relationship Management strukturiert dargestellt: Prozesse, Systeme, Technologien. Springer-Verlag (2003)
10. Döhring, M., Heublein, S.: Anomalies in Rule-Adapted Workflows - A Taxonomy and Solutions for vBPMN. In: Proceedings of the 16th European Conference on Software Maintenance and Reengineering, Szeged, Hungary, IEEE (2012) 117–126
11. Wagelaar, D., Van Der Straeten, R., Deridder, D.: Module superimposition: a composition technique for rule-based model transformation languages. Software and Systems Modeling **9** (2010) 285–309 10.1007/s10270-009-0134-3.
12. Popp, R., Raneburger, D., Kaindl, H.: Tool support for automated multi-device GUI generation from discourse-based communication models. In: Proceedings of the 5th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '13), New York, NY, USA, ACM (2013)
13. Raneburger, D., Popp, R., Kaindl, H.: Model-driven transformation for optimizing PSMs: A case study of rule design for multi-device GUI generation. In: Proceedings of the 8th International Joint Conference on Software Technologies (ICSOFT'13), SciTePress (July 2013)