Message Assertions and Predicate-Based Control-Flow Unfolding Revisited

Thomas S. Heinze[1], Wolfram Amme[1], and Simon Moser[2]

[1] Friedrich Schiller University of Jena, Institute of Computer Science
[t.heinze,wolfram.amme]@uni-jena.de
[2] IBM Research & Development Böblingen
smoser@de.ibm.com

Abstract. In our previous work, we considered, on the one hand, the predicate-based unfolding of a business process's conditional control flow and, on the other hand, the introduction of assertions for the contents of messages exchanged between processes. In this paper, we will sketch how both approaches can be smoothly combined for the automated provisioning of precise low-level Petri net models of business processes.

1 Introduction

The quality of Petri-net-based verification of business processes is tightly coupled to the precision of the process model used. In particular, if verification targets the soundness or controllability of full-specified processes, i.e., executable BPMN or WS-BPEL processes, the Petri net model of a process must reflect the process's control flow as well as process data relevant to the control flow. Otherwise, a thus imprecise process model comprises the danger of an erroneous verification. Yet, a precise and verifiable Petri net model can not be given in the general case due to the Turing-completeness of languages BPMN and WS-BPEL.

In our previous work [2,3], we presented methods which aim at the generation of precise low-level Petri net models for business processes. At the core of the methods is the semantic-preserving transformation of a process's data-based choices into unconditional control flow, such that process data relevant to the choices' conditions, and therefore verification, does not need to be included in the Petri net model. Despite not being effectful in all cases, the methods allow for the generation of precise process models in a number of substantial cases.

In this paper, in order to widen the range of cases our methods are successfully applicable, we integrate the individual methods, namely predicate-based control-flow unfolding [3] and assertions for the contents of incoming messages [2], into a consolidated technique. Specifically, we propose:

- the use of predicate-based abstractions on process data to derive assertions,
- the generalized application of assertions to incoming and outgoing messages,
- the definition of a data-sensitive communication model, which fits with existing algorithms and theories for Petri-net-based process verification.

N. Herzberg, M. Kunze (eds.): Proceedings of the 6th Central European Workshop on Services and their Composition (ZEUS 2014), Potsdam, Germany, 27-03-2014, published at http://ceur-ws.org

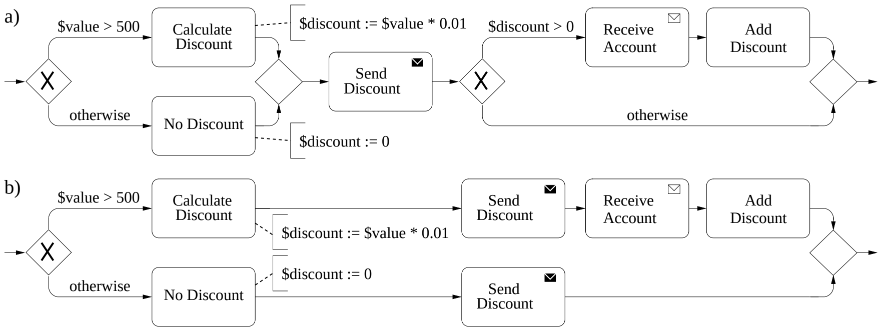


Fig. 1. Snippet of distributed business process (a) and its unfolding (b)

2 Predicate-Based Control-Flow Unfolding

In principle, predicate-based control-flow unfolding [3] aims at resolving the data-based choices of a business process. To this end, a predicate-based abstraction for process data is derived by static analysis, which is afterwards used to evaluate and eliminate the choices' conditions. However, a single choice's condition can be evaluated to true on one path of the control flow and to false on another, according to the values of process data valid at the respective path. Therefore, the process's control flow is unfolded in a controlled fashion, such that differing (abstract) values of process data are assigned separate control flow paths.

Figure 1 a) gives an example: It shows a snippet of a business process, in which the discount for a customer's order is calculated (`$discount`), sent to the customer, and booked to the customer's account if greater than zero. In Figure 1 b), the result of unfolding the snippet is shown. While unfolding, abstractions `$discount` > 5 and `$discount` $= 0$ were automatically derived for `$discount`'s value using static analysis and assigned individual control flow paths. Thereupon, the data-based choice with condition `$discount` > 0 could be successfully evaluated and resolved based on the abstract values.

3 Data-Sensitive Message Channels

However, mapping the unfolded snippet in Figure 1 b) to a Petri net, using the Petri net semantics of [6], still does not result in a precise process model. In the snippet, an order with a value greater 500 requires the customer to send another message while an order smaller or equal 500 implies no further interaction. Unfortunately, this relation is not reflected in the Petri net, since the snippet's (remaining) choice is therein modeled by nondeterminism, so that a partner is not able to determine if it is expected to send another message or not.

In [2], we introduced assertions for incoming messages, based upon simple relational expressions over constants, in order to distinguish between messages with different message contents. Apparently, this approach can as well be applied to outgoing messages and arbitrary predicates. Thus, a (data-sensitive) message channel C_P is now assigned an assertion $\forall x\colon P(x)$, where x denotes the message

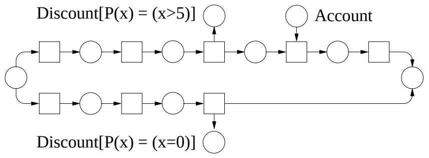


Fig. 2. Precise Petri net model, i.e., open workflow net, for snippet in Figure 1 b)

content and P a first-order predicate. As a remark, this definition can be easily extended to messages containing multiple data items and to assertions with formulas over more than one predicate. Doing so in particular allows for reusing the predicate-based abstraction for process data, which has been derived and utilized in the process of unfolding, as assertions for (outgoing) messages.

For the example in Figure 1 b), message channels with assertions $\forall x\colon (x > 5)$ and $\forall x\colon (x = 0)$ are introduced for outgoing message `Discount`. Consequently, mapping the snippet to a Petri net now yields a precise process model (Figure 2), such that a partner is enabled to determine whether it is expected to send another message or not by considering the assertions assigned to message `Discount`.

Since existing Petri net models for (distributed) business processes do not support assertions for message contents, we need to extend their communication model. For open workflow nets [4], such an extension is obtained by modifying the composition operator so that it considers the assertions while gluing interface places. Note that this can be challenging, i.e., requires the use of a SMT solver, for messages with differing assertions. Further, each incoming message is attached an additional channel with assertion *true*, conflicting the message's other data-sensitive channels, and conventional channels are assigned assertion *true* too.

4 Related Work

Existing approaches for mapping business processes to Petri nets either omit process data entirely or restrict themselves to finite data [1,4,5]. This also applies to the process-to-Petri-net compiler in [4], where, despite being defined on high-level nets, a low-level net is generated in which data-based choices are mapped to nondeterminism and messages are modeled as indistinguishable tokens. High-level nets support data modeling, though, can in general not be verified in the presence of infinite data [7]. To the authors' knowledge, there is no other work done on the application of predicate abstraction to a process-to-Petri-net-mapping.

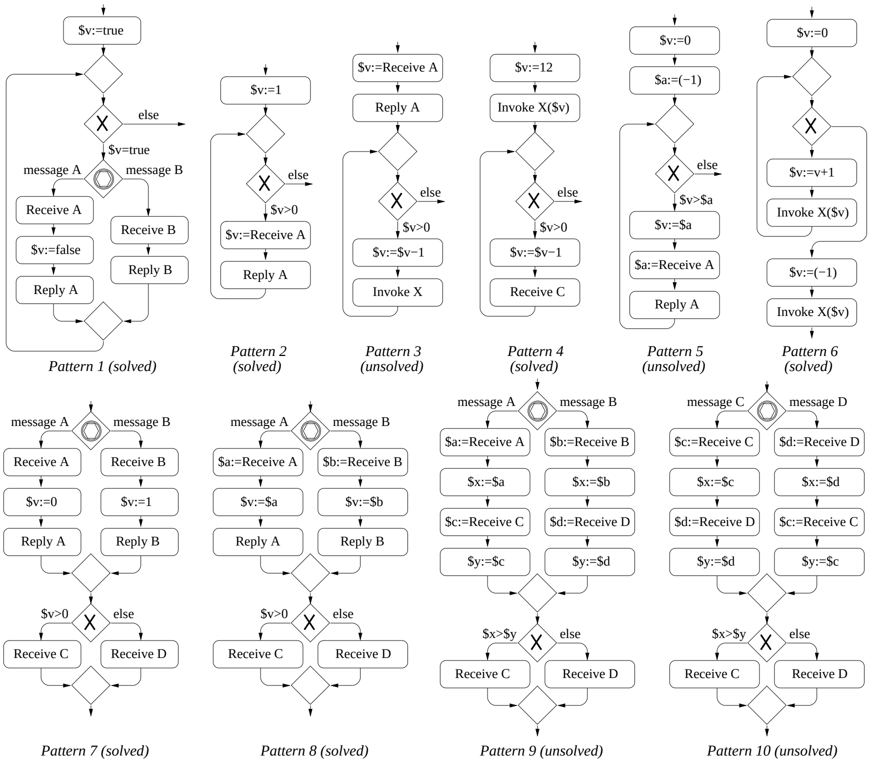
5 Conclusion

In this paper, we have sketched the smooth integration of our previously introduced methods into a consolidated technique by way of an example. Using the technique then allows for generating more precise Petri-net-based process models for business processes in a significant number of cases. In particular, applying the technique to a set of problematic patterns of distributed processes, supplied by an industrial partner (see Appendix A), revealed the successful generation of precise process models for six out of ten process patterns. However, the thorough assessment of efficacy remains subject to future work. To this end, we plan to evaluate the consolidated technique using our process-to-Petri-net compiler.

References

1. Dijkman, R.M., Dumas, M., Ouyang, C.: Semantics and analysis of business process models in BPMN. Information and Software Technology 50(12), 1281–1294 (2008)
2. Heinze, T.S., Amme, W., Moser, S.: Process Restructuring in the Presence of Message-Dependent Variables. In: ICSOC Workshops. pp. 121–132. Springer (2011)
3. Heinze, T.S., Amme, W., Moser, S.: Control Flow Unfolding of Workflow Graphs Using Predicate Analysis and SMT Solving. In: ZEUS 2013. pp. 1–8 (2013)
4. Lohmann, N.: A Feature-Complete Petri Net Semantics for WS-BPEL 2.0. In: WS-FM 2007. pp. 77–91. Springer (2008)
5. Ouyang, C., Verbeek, E., van der Aalst, W.M.P., Breutel, S., Dumas, M., ter Hofstede, A.H.M.: Formal semantics and analysis of control flow in WS-BPEL. Science of Computer Programming 67(2–3), 162–198 (2007)
6. van der Aalst, W.M.P., Hirnschall, A., Verbeek, H.M.W.: An Alternative Way to Analyze Workflow Graphs. In: CAiSE 2002. pp. 535–552. Springer (2002)
7. van der Aalst, W.M.P., Stahl, C., Westergaard, M.: Strategies for Modeling Complex Processes using Colored Petri Nets. In: ToPNoC VII, pp. 6–55. Springer (2013)

Appendix A – Process Patterns



Pattern 1 (solved) *Pattern 2 (solved)* *Pattern 3 (unsolved)* *Pattern 4 (solved)* *Pattern 5 (unsolved)* *Pattern 6 (solved)*

Pattern 7 (solved) *Pattern 8 (solved)* *Pattern 9 (unsolved)* *Pattern 10 (unsolved)*