Mapping Interconnection Choreography Models to Interaction Choreography Models

Oliver Kopp, Frank Leymann, and Fei Wu

Institute of Architecture of Application Systems, University of Stuttgart, Germany
`lastname@iaas.uni-stuttgart.de`

Abstract Choreographies offer a global view on interacting processes. There are two ways to capture this global view: interaction models and interconnection models. Although there is a mapping from interaction models to interconnection models, there is no mapping vice versa. This paper fills this gap and provides a first approach mapping interconnection models to interaction models: The presented approach transforms BPMN models into iBPMN models by using Petri nets as intermediate format.

1 Introduction

Service choreographies describe the interactions between multiple processes [1]. Currently, there are two modeling types available: interaction models and interconnection models. The basic building block of interaction models is the interaction activity. This activity describes a message exchange between two participants in a choreography. In the case of interconnection models, messaging activities of different processes are interconnected. These two metamodels can be seen as different views on the same choreography.

Decker and Weske [2, 3] show a way to generate interconnection models out of interaction models. Currently, there is no work to generate interaction models out of interconnection models. This paper presents a first idea for such a mapping. The motivation is to provide a high-level view on existing interacting services. For this paper, we assume that an interconnection BPMN model—a BPMN model having pools with activities and message flows—has already been constructed out of interacting services.

Consequently, this paper is structured as follows: First, Sect. 2 provides background information and presents related work in the field. Section 3 presents an overview on the approach and Sect. 4 discusses the approach. Finally, Sect. 5 concludes and provides an outlook on future work.

2 Background and Related Work

Figure 1 presents different viewpoints in service-oriented design, adapted to current languages in the Web services stack. (A) On the top service value networks are shown. Service value networks provide a high view on the relationship between services without giving details on the collaboration [4]. (B) Below the service value networks, we see interaction choreography models, such as BPMN 2.0

choreographies [5], iBPMN [6] or BPEL^{gold} [7]. They provide a high-level view on the collaboration protocol by modeling message exchanges as single activities. Most interaction choreography languages hide internal behavior [8]. (C) Interconnection choreography models model each participant as a separate process and may provide internal behavior. Samples for interconnection choreography languages are BPMN 2.0 collaborations, BPMN 1.2 models with more than one pool [9], BPMN^{+} [10] or BPEL4Chor [11]. (D) The behavior of each participant can be expressed as a single BPMN pool or an abstract BPEL process. Additional possibilities to specify behavior of a participant include the open net approach [12]. (E) At the lowest level of abstraction, executable BPMN models or executable BPEL models reside. Each of these models can be deployed on a workflow engine and be enacted.

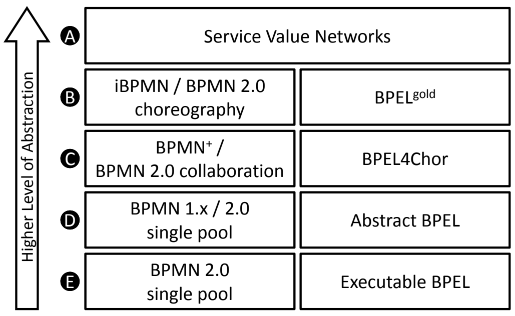
Higher Level of Abstraction
A
Service Value Networks
B
iBPMN / BPMN 2.0 choreography
BPELgold
C
BPMN+ / BPMN 2.0 collaboration
BPEL4Chor
D
BPMN 1.x / 2.0 single pool
Abstract BPEL
E
BPMN 2.0 single pool
Executable BPEL

Figure 1. Different viewpoints

Different viewpoints in service-oriented design were first presented in [13], where especially the choreography view and the orchestration view are distinguished. The choreography view does not distinguish between interaction and interconnection models. Thus, it is unclear, whether interconnection models really are on a lower level than interaction models. That aspect should be investigated in future work. Barros et al. [14] propose role-based views, milestone models and scenario models as views on top of choreography modeling. They regard interaction models as the most abstract choreography models available. Using the technique presented in this paper interaction models can be derived out of interconnection models. The generated interaction models in turn may form a basis to extract a role-based view. This view can then be used to identify potentials for improvement.

The need to generate view on the protocol of running service interactions has been identified by Motahari-Netzhad et al. [15], too. One motivation for them is the evolution of services: in case the implementation of a service evolves, protocol discovery provides an up-to-date protocol definition. Motahari-Netzhad et al. call interaction models protocols and model them by finite state machines [16]. In contrast to our work, they generate interaction models out of audit logs and not out of interconnection models.

It has not yet been investigated whether interaction choreography models or interconnection choreography models are more suitable to capture choreographies. The expressiveness of the two modeling styles, however, is not equal: Decker and Weske [2,3] show that there are anti-patterns in BPMN which cannot be rendered in iBPMN. One example is anti-pattern “D2: Impossible data-based decisions”. For instance, a bidder deciding for himself whether he has won an auction—and not waiting for a decision message—is an instance of that anti-pattern. An additional distinction is that most interaction models do not support internal activities [8].

Regarding the mapping of interaction models to interconnection models, Decker et al. [17] study the property of “non-desynchronizability”: it is impossible to map all interaction models mapped to interconnection models by just splitting an interaction task to a send task and a receive task. The reason is that interaction models assume synchronous communication, whereas interconnection models assume asynchronous communication. In our mapping, we go from asynchronous communication to synchronous communication. Our issue is then the property of "synchronizability", which has been studied by Fu et al. [18]: a set of interacting processes is synchronizable if they are (a) synchronous compatible and (b) autonomous with respect to the choice of the next action. A set of interacting processes is synchronous compatible if for each state where a send task is active, a receive task is reachable by internal transitions only. The autonomous condition demands that each process can only terminate, send or receive a message at one time. This conditions disallows the choice between sending and receiving a message.

A "realizable" choreography model is an interaction model which can be implemented by a set of processes where the composition exactly shows the specified message exchange behavior [19, 20]. Realizability is a property of an interaction choreography model. Our transformation generates a realizable interaction choreography model, since the input of the transformation is a set of processes which realizes the generated interaction choreography model.

Currently, following transformations between the languages presented in Fig. 1 are available: The interplay between Service Value Networks and choreographies is shown in [4]. The integration between BPMN 1.2 choreographies and BPEL4Chor has been investigated in [21]. A mapping from BPEL^{gold} to BPEL4Chor is presented in [7]. A mapping from executable BPEL processes to a BPEL4Chor description is presented in [22]. Currently, there is neither a mapping from BPMN 1.2 choreographies to iBPMN nor a mapping from BPEL4Chor to BPEL^{gold}. This paper presents a first approach of a possible mapping. In case the techniques are applied to a BPEL4Chor to BPEL^{gold} mapping, it is possible to generate an interaction choreography view out of executable BPEL processes.

Figure 2 presents a sample choreography for investments using the BPMN 1.2 notation. First, a financial advisor offers a product to a client. Subsequently, the client has 24 hours time to decide whether he accepts the investment proposal or rejects the proposal. The example is used in [23] to check choreography conformance during runtime. The equivalent interaction model is presented in Fig. 3. Here, the control flow is rendered between the pools and the local messaging activities have been replaced by interaction activities. The timer event and the data-based exclusive gateway are annotated with the controlling role customer indicating that the customer is responsible to control the time and the decision which message to send.

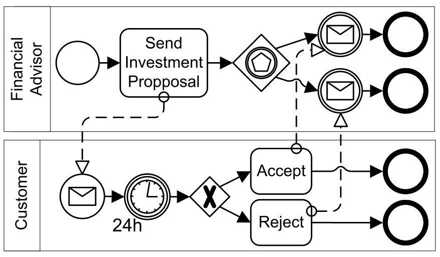

Figure 2. Interconnection model describing investment offers

Decker and Barros [6] introduce interaction Petri nets (iPNs) as formal foundation of iBPMN. Transitions are separated into interaction transitions, controlled silent transitions and uncontrolled silent transitions. Interaction transitions represent an interaction between two participants, controlled silent transitions represent decisions and timers controlled by a set of dedicated participants. Uncontrolled silent transitions are used solely for routing purposes. We use iPNs as an intermediate format to go from BPMN to iBPMN.

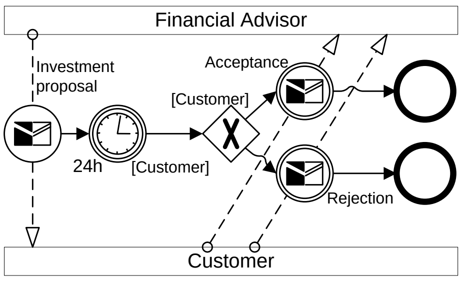

Figure 3. Interaction model describing investment offers

Reduction techniques for Petri nets are presented in [24] and [25]. Murata [24] presents basic techniques to collapse redundant structures, such as a sequence of places and transitions, where a transition only has one incoming edge and one outgoing edge. Berthelot et al. [25] introduce the definition of redundant places. We use their definition to reduce the Petri net resulted from the mapping.

Dijkman et al. [26] present a mapping from BPMN to classical Petri nets. The mapping supports multi-instance loops if the number *n* of instances is known at design time. In this case, the part of the Petri net representing the respective part of the loop is duplicated *n* times. As the semantics of the OR join in BPMN is not formally defined, the work does not map OR joins at all. The work of Dijkman et al. is an integral part of our transformation.

Lohmann et al. [27] present a survey on current transformation approaches from BPEL, BPMN and EPC process models to Petri nets. There, the work of Dijkman et al. is also regarded as the de facto approach to transform BPMN models to Petri nets.

3 Overview on the Approach

The goal of the transformation is to keep the ordering of the message exchanges and to provide an iBPMN model with as least nodes and edges as possible. Thus, we want to keep the set of traces as well as the branching behavior as Decker et al. did in their mapping from interconnection models to interconnection models [19]. Since this paper presents a first idea, we do not present a formal definition of the properties we want to preserve here.

We require the input BPMN model to be sound and safe. Furthermore, we require them to contain only following elements: sequence flows, plain start events, start message events, intermediate message events, tasks, data-based exclusive gateways, event-based exclusive gateways, parallel gateways, intermediate timer events as well as plain end events. Tasks may be configured as while or repeat until loop. Regarding message flows, we demand that tasks and events are only connected to at most one message flow and that each message flow has a source and a target. Thus, our work focuses on the positive control flow only and excludes exception, termination and compensation handling. Finally, we require the BPMN model to be free of the anti-patterns presented in [2, 3] and to be synchronizable.

Figure 4 presents the overview on the approach. We start with a BPMN 1.2 model. This model is transformed directly into an interaction Petri net model. In this step, control flow structures are transformed using the approach presented in [26] and pairs of interaction activities are directly transformed to interaction transitions. The restriction on message flows in the input model ensures that this transformation is unambiguous. Tasks configured as loops are expanded to gateways surrounding the mapped content of the task.

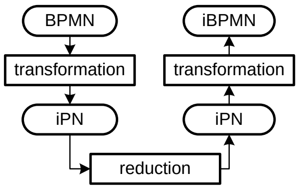

Figure 4. Overview

The control flow surrounding the interaction transitions is not modified in the first step. To gain a proper interaction control flow, the resulting Petri net is reduced using the techniques of [24] and [25]. The reduced iPN model is then transformed into an iBPMN model.

In the following, we present the idea of the algorithm by transforming the interaction model presented in Fig. 2 into an iBPMN model. A detailed and formal description of the transformation is out of scope of this paper, but is presented in [28].

Figure 5 presents the interaction Petri net after mapping the BPMN choreography to iPNs. The nodes $p_0, t_0, p_1, t_1, p_2, t_2, t_7, p_3$ represent the financial advisor. p_2 in combination with t_2 and t_7 represents the event-based gateway. The nodes $p_4, t_3, p_5, t_1, p_6, t_4, p_7$, t_5, p_8, $t_2, t_6, p_9, t_7, p_{10}$ represent the customer. t_4 is the mapped timer event with the controlling role customer. We introduce an additional timer marking to controlled transitions to be able to correctly map timer events back to timer events. If the marker was not present, these transitions would have been mapped to a gateway. Finally, p_7, t_5, t_6 represent the data-based exclusive gateway.

A place p is redundant to a place $q \neq p$ if p is not marked in the initial marking and p is always marked if q is marked [25]. Thus, the analysis of the iPN leads to following results: 1) p_0, t_0 and p_4, t_3 can be removed, 2) then, p_1 and p_5 have no preceding transition and target the same transition. Hence, p_5 can be removed. 3) t_5, p_8 and t_6, p_9 can be removed, 4) p_2 is redundant to p_7 and can be removed, 5) p_3 is redundant to p_{10} and can be removed.

Result 4 cannot be gained by applying the structural reduction rules of [24] only, because the rules are only applicable for a transition between two places or a place between two transitions. The intermediary steps are not shown due to space limitations. The overall result is presented in Fig. 6.

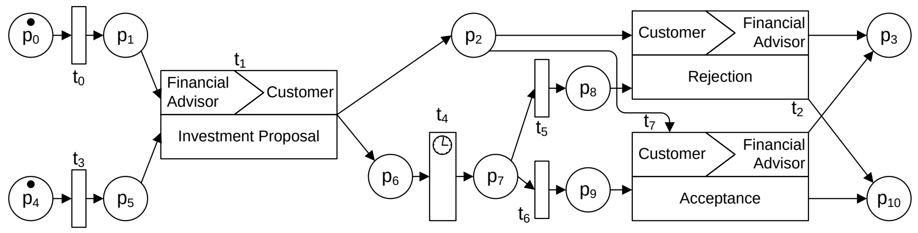

Figure 5. Interaction Petri net derived from the investment offer scenario

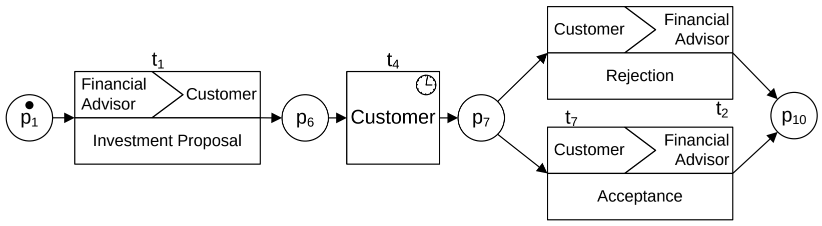

Figure 6. Reduced interaction Petri net

To map the reduced iPN to an iBPMN model, the iPN has to be modified to enable a pattern-based transformation. The modification ensures that all interaction transitions have exactly one incoming and one outgoing arc. If an interaction transition t_i has more than one incoming arc, a transition t is added before t_i and all arcs targeting t_i are retargeted to t and a new arc from t to t_i is added. A similar approach is taken for outgoing arcs. Now, parallel gateways are always generated out of a transition with multiple outgoing arcs.

iBPMN supports both event-based and data-based gateways to offer a choice between alternative message exchanges. iBPMN demands that an event-based gateway is used in the case a timer-event is involved in the current conversation and a data-based gateway in all other cases. In case the interaction transition is preceded by a place which is followed by multiple transitions, the place has to be transformed into a data-based gateway or to an event-based gateway depending on the type of the subsequent transitions. BPMN does not support mixed choices [17]. That means, the interactions following a place are always initiated by the same sender. In case a timer transition is present after the place, the place is transformed into an event-based gateway. Otherwise, the place is transformed into a data-based gateway with controlling S, where S is the sender in all interaction transitions following.

The remaining constructs are transformed by applying the rules of [2, 3] "backward". That means, we map the constructs from iPN to iBPMN instead of mapping iBPMN to iPNs. The result is presented in Fig. 3.

4 Discussion of the Approach

Interconnection choreography models assume asynchronous communication, whereas interaction models assume synchronous communication. The latter means, the sender is blocked until the receiver consumes the message. In this paper, we assume that the asynchronous model is "synchronizable". Future work, however, has to provide a detailed investigation whether the synchronizability definitions given by Fu et al. [18] are applicable to interconnection BPMN models.

The approach uses Petri nets as intermediate format for the mapping. The approach cannot handle multi-instance loops without a priori knowledge of the number of instances. As the Petri net is not used for verification, the entry place of the loop can be labeled with a marker. That marking can later be used to transform the loop back to a multi-instance loop in iBPMN.

BPMN 1.2 does not foresee to mark participants as multi-instance participants. These extensions have been introduced in [29] to overcome this limitation. Thus, the start place of the participant can be marked if it is a multi-instance participant to enable a transformation to a multi-instance participant in iBPMN.

The idea of markings can also be used for OR joins. An OR join can be transformed into a transition marked with "OR". Then, the reduction part of the algorithm does not reduce that transition and the mapping from iPN to iBPMN transforms that transition to an OR gateway.

5 Conclusion and Outlook

This work presented a first mapping from BPMN to iBPMN using Petri nets as intermediate formalism. We mapped the positive control flow only and thus leaving the negative control flow for future work. As the algorithm was sketched, future work has to provide a formal presentation of the algorithm as well as a formal presentation of the properties the algorithm preserves and proofs of them.

Besides using Petri nets as intermediate formalism, it seems to be possible that BPMN can be directly mapped to iBPMN. The phases of that approach are: (i) transformation of pairs of messaging activities to one interaction activity and (ii) reduce the resulting iBPMN model. Thus, we currently investigate whether and how the reduction rules presented in [24, 25] can be applied to iBPMN models. Subsequently, we are planning a detailed investigation on the direct BPMN to iBPMN mapping and a comparison to the presented approach.

Acknowledgments We thank the anonymous reviewers for their valuable comments and new insights on the opportunities and limitations of the approach.

References

1. Decker, G., Kopp, O., Barros, A.: An Introduction to Service Choreographies. Information Technology **50**(2) (2008) 122–127
2. Decker, G., Weske, M.: Interaction-centric Modeling of Process Choreographies. (2010) in submission.
3. Decker, G.: Design and Analysis of Process Choreographies. PhD thesis, Hasso Plattner Institute, University of Potsdam (2009)
4. Bitsaki, M., et al.: An Architecture for Managing the Lifecycle of Business Goals for Partners in a Service Network. In: ServiceWave 2008, Springer (2008) 196–207
5. Object Management Group (OMG): Business Process Model and Notation (BPMN) Specification 2.0. (2009) v2.0 Beta 1.
6. Decker, G., Barros, A.P.: Interaction Modeling Using BPMN. In: 1st International Workshop on Collaborative Business Processes 2007, Springer (2007) 208–219
7. Engler, L.: BPEL^{gold}: Choreography on the Service Bus. Diploma thesis, University of Stuttgart, IAAS (2009)
8. Kopp, O., Leymann, F.: Do We Need Internal Behavior in Choreography Models? In: ZEUS 2009. Volume 438., CEUR-WS.org (2009) 68–73
9. Object Management Group (OMG): Business Process Modeling Notation (BPMN) Version 1.2. (January 2009) `http://www.bpmn.org/`.

10. Pfitzner, K., Decker, G., Kopp, O., Leymann, F.: Web Service Choreography Configurations for BPMN. In: WESOA 2007, Springer (2007) 401–412
11. Decker, G., Kopp, O., Leymann, F., Weske, M.: Interacting services: from specification to execution. Data & Knowledge Engineering **68**(10) (2009) 946–972
12. Wolf, K.: Does my service have partners? LNCS T. Petri Nets and Other Models of Concurrency **5460**(2) (2009) 152–171
13. Dijkman, R., Dumas, M.: Service-oriented Design: A Multi-viewpoint Approach. International Journal of Cooperative Information Systems **13**(4) (2004) 337–368
14. Barros, A., Decker, G., Dumas, M.: Multi-staged and multi-viewpoint service choreography modelling. In: SEMSOA 2007. (2007) 1–15
15. Motahari-Nezhad, H.R., et al.: Deriving Protocol Models from Imperfect Service Conversation Logs. IEEE Transactions on Knowledge and Data Engineering **20** (2008) 1683–1698
16. Benatallah, B., Casati, F., Toumani, F.: Representing, analysing and managing web service protocols. Data Knowl. Eng. **58**(3) (2006) 327–357
17. Decker, G., Barros, A.P., Kraft, F.M., Lohmann, N.: Non-desynchronizable Service Choreographies. In: ISCOC 2008. (2008) 331–346
18. Fu, X., Bultan, T., Su, J.: Synchronizability of Conversations among Web Services. IEEE Trans. Softw. Eng. **31**(12) (2005) 1042–1055
19. Decker, G., Weske, M.: Local Enforceability in Interaction Petri Nets. In: Business Process Management 2007, Springer (2007) 305–319
20. Fu, X., Bultan, T., Su, J.: Conversation protocols: a formalism for specification and verification of reactive electronic services. Theor. Comput. Sci. **328**(1-2) (2004) 19–37
21. Decker, G., Kopp, O., Leymann, F., Pfitzner, K., Weske, M.: Modeling Service Choreographies using BPMN and BPEL4Chor. In: CAiSE '08, Springer (2008) 79–93
22. Steinmetz, T.: Generierung einer BPEL4Chor-Beschreibung aus BPEL-Prozessen. Student Thesis, University of Stuttgart, IAAS (2007) in German.
23. Kopp, O., van Lessen, T., Nitzsche, J.: The Need for a Choreography-aware Service Bus. In: YR-SOC 2008. (2008) 28–34
24. Murata, T.: Petri nets: Properties, analysis and applications. Proceedings of the IEEE **77**(4) (1989) 541–580
25. Berthelot, G., Lri-Iie: Checking properties of nets using transformations. In: Advances in Petri Nets 1985, Springer (1986) 19–40
26. Dijkman, R.M., Dumas, M., Ouyang, C.: Semantics and analysis of business process models in BPMN. Information and Software Technology **50**(12) (2008) 1281–1294
27. Lohmann, N., Verbeek, E., Dijkman, R.: Petri Net Transformations for Business Processes – A Survey. In: ToPNoC, Springer (2009) 46–63
28. Wu, F.: Mapping interconnection choreography models to interaction models. Diploma thesis, University of Stuttgart, IAAS (2009)
29. Decker, G., Puhlmann, F.: Extending BPMN for Modeling Complex Choreographies. In: CoopIS 2007, Springer (2007) 24–40