Realizability of Interaction Models

Gero Decker

Hasso-Plattner-Institute, University of Potsdam, Germany
gero.decker@hpi.uni-potsdam.de

Abstract. In scenarios where a set of independent business partners engage in complex conversations, interaction models are a means to specify the allowed interaction behavior from a global perspective. Atomic interactions serve as basic building blocks and behavioral dependencies are defined between them. The notion of *realizability* centers around the question whether there exist a set of roles that collectively realize the specified behavior. This notion has been studied in the literature in different flavors. This paper aims at providing an overarching framework for realizability.

1 Introduction

Two approaches for choreography modeling can be identified in the literature. *Interconnection models* are collections of observable behavior models for interacting roles. Each observable behavior model belongs to one role and contains communication activities and the behavioral dependencies between them. Corresponding communication activities are then interconnected. Choreography languages following this style are BPMN [1] and BPEL4Chor [5]. *Interaction models*, on the other hand, consist of interactions and global behavior dependencies between them. They are global in the sense that they are not explicitly assigned to any role and it remains unspecified who is responsible for enforcing them. Choreography languages following this style are WS-CDL [8] and iBPMN [3].

It turns out that some interaction models are not realizable. Imagine that an interaction between C and D must only happen after A and B have interacted. Here, C and D cannot know when the first interaction has actually happened. While this example is obviously not realizable, there are other scenarios where realizability might be given under certain assumptions. This paper will provide a classification of the different dimensions of realizability.

The remainder of this paper is structured as follows. The next section lists a number of motivating examples for the notion of realizability. Section 3 identifies the dimensions of realizability. Section 4 reports on related work, before Section 5 centers around realizability checking. Section 6 concludes.

2 Motivating Examples

Conversation models, as presented in [7], are a formalism for interaction models. They are finite state automata with an alphabet $R \times M \times R$. R denotes the set

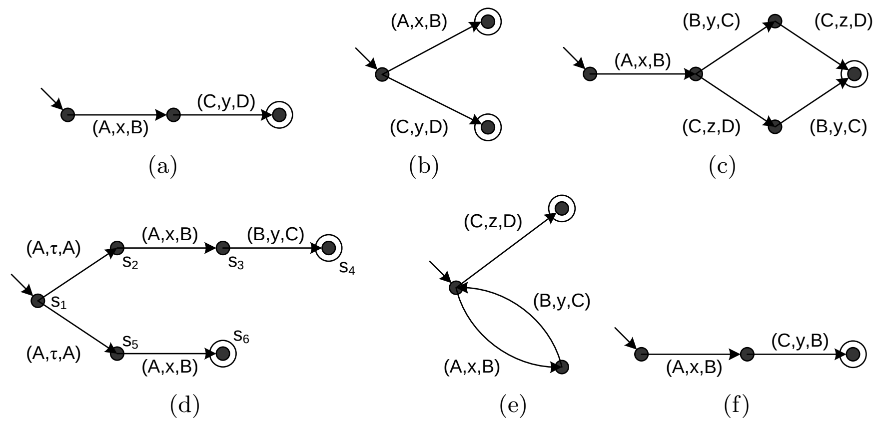


Fig. 1. Conversation model examples

of roles and M the set of message types. The 3-tuple (s, m, r) then describes the sender, the type and the receiver of a message.

Figure 1 illustrates a number of sample interaction models. States are depicted by circles and the transitions by arrows. The initial state is targeted by an arrow without source state and the final state is denoted by a double-circle.

Figure 1(a) shows the example described in the introduction. It is not possible to find interacting roles that exactly show the specified behavior. Nevertheless, it would be possible to find interacting roles that show a subset of the specified behavior: Imagine two roles A and B that interact and roles C and D simply do nothing. In this case, however, a conversation would not terminate properly, as the final state cannot be reached.

Figure 1(b) shows a choice between two interactions. Similarly to the previous example, A and B cannot know whether C and D have already interacted and vice versa. In contrast to the previous example, we can find roles that collectively realize a subset of the specified behavior with proper termination. Imagine again that only roles A and B interact while C and D do nothing. However, we are not able to find a set of roles that realize a subset of the behavior where all interactions from the conversation model are reachable.

Similarly to the first example, the enablement dependency between the AB interaction and the CD interaction is the problem in Figure 1(c). As a solution, C could wait for the message from B before interacting with D. That way, the resulting behavior would be a properly terminating subset of the initially specified behavior.

Figure 1(d) shows an example containing a non-deterministic choice. This conversation model represents that A should internally be able to decide whether B will interact with C later on. However, B cannot observe this decision as in any case it will get a message x from A. As A does not have any control over the BC interaction, the decision whether this interaction takes place or not will be independent form A's initial choice. When only considering the possible traces of the conversation model we can easily create roles that collectively produce exactly the same traces. The main difference is that B or C can decide whether the final interaction takes place or not in the realization. We see that considering the branching structure is crucial whenever the ownership of (and the moment of) choices is of importance. It might be argued that local choices are irrelevant in choreographies. This might be true if choreographies are considered to be a collection of mere interaction sequences. However, from a business perspective it makes a major difference who makes a branching decision. Therefore, this should be reflected in the formal model as well.

Figure 1(e) shows a cyclic example containing a choice between an AB interaction and a CD interaction, similarly to the second example. The difference here is that by expanding the cycle to a sequence, we can at least find roles that realize a subset of the behavior.

Finally, Figure 1(f) shows an example that is perfectly realizable in a synchronous world, where C can block B until it has interacted with A. However, when considering an asynchronous world, where message sending and receiving do not happen in one step, the order of the send activities would not conform to the order of interactions in the conversation model.

3 Dimensions of Realizability

The examples from the previous section show that we need to distinguish different dimensions of realizability. The following three dimensions apply.

Complete behavior vs. subset of behavior. Choreographies define constraints and obligations of the roles involved. Constraints apply as the choreography enumerates all allowed interactions in every conversation state, obligations apply as a final state must be reached which is only possible through the execution of the given interactions.

In this context, we can either demand that it must be possible to carry out the complete behavior specified in the choreography. Or, a subset of the behavior might already be sufficient. Here, the follow-up question is what a valid subset would be. For instance, proper termination of conversations might be a basic criterion. Furthermore, reachability of all interactions from the original choreography might also be demanded.

Communication model. Synchronous communication could be assumed, where sending and receiving of messages must happen at the same time. Two flavors are possible in this context: it might be allowed that a sender blocks until the receiver is ready to receive the message. Alternatively, the conversation fails if a role can only send in a given state without any other role being able to receive the message.

In asynchronous settings, message send and receive do not happen in one step. Here, message buffers are introduced for storing the incoming messages. We might assume that there is only one queue, e.g. with FIFO message delivery, or that there is a buffer where any incoming message can be received from.

The order of interactions is of central importance. However, especially in the case of asynchronous communication, there are different options of what ordering relationships to consider. For instance, only the ordering of send transitions might be considered, or the ordering of receive transitions or the ordering of communication transitions within the individual roles might be of importance.

Equivalence notion. Having agreed on what ordering relationships to consider, it is important to choose an equivalence notion for comparing the original choreography and the collective behavior of the roles. Here, trace-based techniques can be applied. This is sufficient when dealing with deterministic behavior in the choreography and the roles. Branching structures are of relevance in the presence of non-determinism. Here, bisimulation-like techniques can be used.

In order to formally capture the different notions of realizability we need to introduce the following concepts. C denotes the set of all choreographies (also with silent transitions). R denotes the set of all role behavior models. $\oplus : \wp(R) \to C$ is a function that composes a choreography out of a set of role behavior models. $\sim \subseteq C \times C$ is a binary relation on choreographies.

Please note that $\oplus$ heavily depends on the communication model chosen and, in the case of asynchronous communication, the ordering relationships to be considered. In the special case of considering the order of communication transitions within a role, $\oplus$ depends on the role under investigation. $\sim$ depends on whether the complete or only a subset of the behavior is demanded and it also depends on the equivalence notion chosen.

Definition 1 (Realizability). *A choreography $c \in C$ is realizable, iff there exists a set of roles $r_1, \ldots, r_n \in R$ such that $\oplus(r_1, \ldots, r_n) \sim c$.*

4 Related Work

Realizability checking for conversation models was presented in [7]. Here, the notion of realizability does not consider branching structures in the conversation models and focuses on trace equivalence between the collective behavior of the roles and the original conversation model. Asynchronous communication is considered where each role has one FIFO queue for all incoming messages. Realizability is broken down to three requirements. (1) Synchronous compatible condition: The conversation model is projected to the different roles, which are then interconnected under the assumption of synchronous communication (called the *syn-configuration*). The condition for each state in the syn-configuration is that whenever a role is ready to send a message there must be another role that is ready to receive this message. (2) Autonomous condition: It is demanded for each role projection that there is no state where the role is ready to send *and* to receive a message. Rather, in each state the role projection is either ready to send one out of a set of messages or ready to receive one out of a set of message. Furthermore, it must not be possible to send or receive message in a final state.

(3) Lossless join condition: The join of the role projections must show exactly the same behavior as the original conversation model. Realizability for message sequence charts was studied in [2].

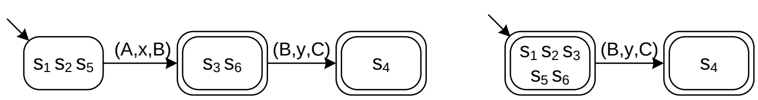
The notion of local enforceability was first introduced in [10]. Here, only a subset of behavior is demanded as well as the reachability of all interactions from the original choreography. Enforceability checking is carried out using structural rules rather than considering the state space. Synchronous communication was assumed. Realizability and local enforceability was also studied in the context of interaction Petri nets in [6]. Again, synchronous communication is assumed and proper termination and reachability of all interactions is demanded. In contrast to the previous work on local enforceability, enforceability checking is done using the state space. Realizability is defined based on branching bisimulation.

The notion of desynchronizability investigates whether a choreography that is realizable under the assumption of synchronous communication properly terminates under the assumption of asynchronous communication (with one buffer per message type) [4].

5 Realizability Checking

Similarly to the approaches in [7] and [6] we construct the role projections for every role in a conversation model and then study the composition of these projections. As [7] does not consider branching structures, role projection is based on minimal finite state automata containing only those interactions where the particular role is involved in.

As we want to preserve the branching structure we carefully need to consider the observability of choices within each role. We construct the role behavior for a role r in a similar way like in the operating guidelines approach [9]. (a) Start with the initial state $s = s_0$ and create a new node n. (b) Determine those states that can be reached from s without involvement of r. All these states are added to node n. (c) Identify all transitions with involvement of r that originate in one of the states belonging to n. For each transition label (s, m, r) determine if there is already a node n' corresponding to all states s' that are reachable via transitions with label (s, m, r). If such an n' does not exist, create it. For every s' and n' continue with step (b).



(a) Role projection for B (b) Role projection for C

The nodes and their connections are the resulting role projection c_r for r. The initial node n becomes the initial state of c_r and all nodes containing final states become final states of c_r. Figures 2(a) and 2(b) illustrate this for roles B and C and the conversation model from Figure 1(d).

An exception to rule (c) applies to all those cases where several transitions with the same label originate in the same state. In this case, different nodes must be identified / created. If multiple states belonging to the same node have several transitions with the same label (s, m, r), nodes must be identified / created for the different combinations.

6 Conclusion

This paper motivated different dimensions for realizability and investigated role projection for those realizability notions that are based on bisimulation. Here, the branching structures must be considered carefully in order to cater for the moment of observability of choices. Composition of role projections for different communication models was outside of the scope of this paper due to the space restrictions. Also the binary relation for comparing the composition and the original conversation model was not covered.

The work presented in this paper is based on conversation models. Interaction Petri nets [6] are an alternative formalism for interaction models with concurrency. It is desirable to preserve concurrency as much as possible during role projection. Therefore, future work will center around an approach, where transformation rules similar to those presented in [6] are applied.

References

1. Business Process Modeling Notation, V1.1. Technical report, OMG, Jan 2008.
2. R. Alur, K. Etessami, and M. Yannakakis. Inference of message sequence charts. In *ICSE*, pages 304–313, New York, NY, USA, 2000. ACM.
3. G. Decker and A. Barros. Interaction Modeling using BPMN. In *CBP*, number 4928 in LNCS, pages 206–217, Brisbane, Australia, September 2007.
4. G. Decker, A. Barros, F. M. Kraft, and N. Lohmann. Non-desynchronizable service choreographies. In *ICSOC*, LNCS, Sydney, Australia, Dec 2008. Springer Verlag.
5. G. Decker, O. Kopp, F. Leymann, and M. Weske. BPEL4Chor: Extending BPEL for Modeling Choreographies. In *ICWS*, pages 296–303, Salt Lake City, Utah, USA, July 2007. IEEE Computer Society.
6. G. Decker and M. Weske. Local Enforceability in Interaction Petri Nets. In *BPM*, number 4714 in LNCS, pages 305–319, Brisbane, Australia, September 2007. Springer Verlag.
7. X. Fu, T. Bultan, and J. Su. Conversation protocols: A formalism for specification and analysis of reactive electronic services. *Theoretical Computer Science*, 328(1-2):19–37, November 2004.
8. N. Kavantzas, D. Burdett, G. Ritzinger, and Y. Lafon. Web Services Choreography Description Language Version 1.0, W3C Candidate Recommendation. Technical report, November 2005. http://www.w3.org/TR/ws-cdl-10.
9. N. Lohmann, P. Massuthe, and K. Wolf. Operating guidelines for finite-state services. In *ICATPN*, volume 4546 of *LNCS*, pages 321–341, Siedlce, Poland, June 2007. Springer Verlag.
10. J. M. Zaha, M. Dumas, A. ter Hofstede, A. Barros, and G. Decker. Service Interaction Modeling: Bridging Global and Local Views. In *EDOC*, pages 45–55, Hong Kong, Oct 2006. IEEE Computer Society.