On the Behavioural Dimension of Correspondences between Process Models

Matthias Weidlich and Mathias Weske

Hasso-Plattner-Institute, University of Potsdam, Germany
{matthias.weidlich,weske}@hpi.uni-potsdam.de

Abstract. Enterprise-wide process harmonisation initiatives require the analysis of commonalities of existing business process models. That is, correspondences between activities are identified, such that the behavioural equivalence of the models can be assessed thereafter. Due to refinements, these correspondences can relate sets of activities to each other, i.e., there are complex 1:n or n:m correspondences. In this paper, we discuss how notions of behaviour inheritance can be applied in this context. In addition, we elaborate on how structural information can be leveraged to identify violations of behaviour inheritance.

1 Introduction

Enterprises model their business processes for various purposes, e.g., documentation of business operations, staff planning, or process automation. As a consequence, process models might show deviations in terms of level of detail, control flow, and activity labels, even if they capture a common real-world process.

In order to detect inconsistencies between related process models, their commonalities and differences have to be identified [1]. As a first step, this requires the identification of activities that *correspond* to each other. Note that this might not imply semantic equivalence of the activities (e.g., 'Get Quote' might correspond to 'Enter Quote Details' although both activities are not semantically equivalent). Such correspondences can be elementary 1:1 matches between activities, or complex 1:n (or even n:m) matches due to refinements of activities. Correspondences may stem from a system analyst inspecting the models or from automatic matching, cf., [2, 3]. As a second step, the behavioural equivalence of process models aligned by correspondences is assessed to detect inconsistencies and guide process harmonisation efforts.

Although not in a straight-forward manner, existing notions of behaviour inheritance might be applied for this purpose even in the presence of complex correspondences between two process models. However, we argue that certain violations of behaviour inheritance can be detected structurally. In this paper, we show that a structural decomposition of a sound free-choice WF-system might provide valuable hints at correspondences that violate behaviour inheritance. In particular, we focus on the notion of projection inheritance in the context of trace semantics, i.e., an adapted version of the trace equivalence criterion [4].

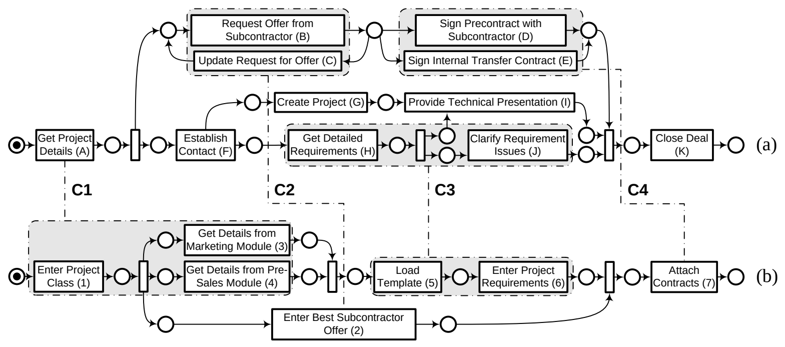

Request Offer from Subcontractor (B)
Update Request for Offer (C)
Sign Precontract with Subcontractor (D)
Sign Internal Transfer Contract (E)
Create Project (G)
Provide Technical Presentation (I)
Get Project Details (A)
Establish Contact (F)
Get Detailed Requirements (H)
Clarify Requirement Issues (J)
Close Deal (K)
(a)
C1
C2
C3
C4
Get Details from Marketing Module (3)
Enter Project Class (1)
Get Details from Pre-Sales Module (4)
Load Template (5)
Enter Project Requirements (6)
Attach Contracts (7)
(b)
Enter Best Subcontractor Offer (2)

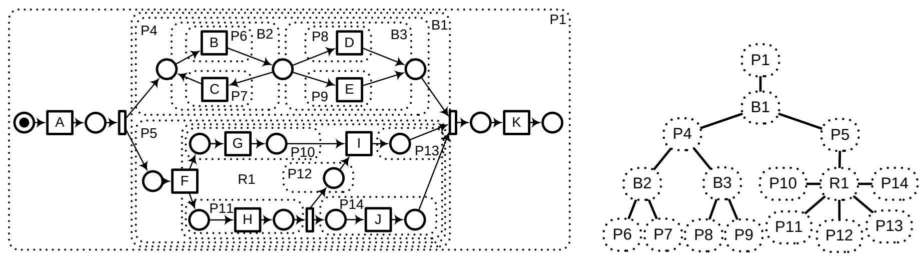
Fig. 1. Two exemplary process models, (a) focussing on the organisational perspective, (b) depicting the respective technical process, along with complex correspondences

Fig. 1 illustrates the problem addressed by this paper using two Petri nets that model a project planning process. Model (a) assumes an organisational perspective, whereas (b) shows the processing according to a supporting IT system. There are four complex correspondences highlighted using dash-dotted lines. Note that τ-transitions are never considered to be part of any correspondence. Clearly, the execution dependencies for transitions that are part of correspondences are not always consistent in both models. For instance, transitions D and H might be enabled concurrently in model (a), whereas their corresponding transitions, e.g., 5 and 7, are causally related in model (b). The question of whether and how such inconsistencies can be detected structurally is addressed in this paper.

The remainder of this paper is structured as follows. Section 2 gives preliminaries for our work. We discuss structural properties of correspondences in Section 3. Section 4 elaborates on behavioural analysis of correspondences. Finally, Section 5 reviews related work, before Section 6 concludes the paper.

2 Preliminaries

We recall basic definitions for workflow (WF-) systems based on [5, 6]. A *net* is a tuple $N = (P, T, F)$ with P and T as finite disjoint sets of places and transitions, and $F \subseteq (P \times T) \cup (T \times P)$ as the flow relation. We write $X = (P \cup T)$ for all nodes and F^+ for the transitive closure of F. For a node $x \in X$, $\bullet x := \{y \in X \mid (y, x) \in F\}$, $x\bullet := \{y \in X \mid (x, y) \in F\}$, $in_N(x) := \{(n, x) \mid n \in \bullet x\}$, and $out_N(x) := \{(x, n) \mid n \in x\bullet\}$. A net N is *free-choice*, iff $\forall\ p \in P$ with $|p\bullet| > 1$ holds $\bullet(p\bullet) = \{p\}$. N is conntected, if $\forall\ x_1, x_2 \in X\ [\ x_1 F^+ x_2\ \wedge\ x_2 F^+ x_1\ ]$. A *workflow (WF-) net* is a net $N = (P, T, F)$, such that there is exactly one place $i \in P$ with $\bullet i = \emptyset$, exactly one place $o \in P$ with $o\bullet = \emptyset$, and the short-circuit net $N' = (P, T \cup \{t_c\}, F \cup \{(o, t_c), (t_c, i)\})$ is connected. $M : P \mapsto \mathbb{N}$ is a *marking* of N and M_p denotes the marking that puts a token on p and no token elsewhere for a place $p \in P$. Given a WF-net N with M_i as its initial marking, we refer to the tuple $S = (N, M_i)$ as a *WF-system* and assume N to be defined always as $N = (P, T, F)$. We do not recall semantics for WF-systems but refer to [5, 6]. (N, M_i) is *sound*, iff the short-circuit system (N', M_i) is live and bounded [7].



(a) (b)

Fig. 2. (a) WF-system and its canonical fragments, (b) its RPST (without T-fragments)

We apply the structural decomposition technique introduced as the RPST in [8] to parse a WF-system into a hierarchy of fragments with a single entry node and a single exit node. Then, the RPST is a containment hierarchy of canonical fragments of the system that can be computed in linear time.

Definition 1 (Fragments, RPST). *Let $N = (P, T, F)$ be a WF-net with initial place i and final place o.*

- *A node $x \in X'$ of a connected subnet $N' = (P', T', F')$ of a net N is a* boundary node*, if $\exists\ e \in in_N(x) \cup out_N(x)$ $[\ e \notin F'\ ]$. If x is a boundary node, it is an* entry *of N', if $in_N(x) \cap F' = \emptyset$ or $out_N(x) \subseteq F'$, or an* exit *of N', if $out_N(x) \cap F' = \emptyset$ or $in_N(x) \subseteq F'$.*
- *Any connected subnet ω of N, is a* fragment*, if it has exactly two boundary nodes, one entry and one exit.*
- *A fragment $\omega = (P_\omega, T_\omega, F_\omega)$ is* canonical *in a set of fragments Ω of N, iff $\forall\ \gamma = (P_\gamma, T_\gamma, F_\gamma) \in \Omega$ $[\ \omega \neq \gamma \Rightarrow (F_\omega \cap F_\gamma = \emptyset) \vee (F_\omega \subset F_\gamma) \vee (F_\gamma \subset F_\omega)\ ]$.*

The RPST of N is a tuple $\mathcal{R}_N = (\Omega, \chi, t)$, where:

- *Ω is a set of all canonical fragments of N,*
- *$\chi : \Omega \rightarrow \mathcal{P}(\Omega)$ is a function that assigns to fragment its child fragments,*
- *$t : \Omega \rightarrow \{T, P, B, R\}$ is a function that assigns a type to a fragment.*

Fig. 2 shows the canonical fragments for one of the WF-systems of our example along with the corresponding RPST. Note that each canonical fragment can be classified as being either a trivial (T) fragment (a single edge), a polygon (P) fragment (a sequence of fragments), a bond (B) (fragment collection with common boundary nodes), or a rigid (R) fragment (any other structure).

3 Structural Properties of Correspondences

This section clarifies the notion of a correspondence between two WF-systems. Following on the terminology known from the domain of schema matching or ontology matching [9], a correspondence is defined by two sets of transitions of the WF-systems.

Definition 2 (Correspondence). *Let (N', M_i') and (N'', M_i'') be two WF-systems. The* correspondence relation $\sim \subseteq \wp(T') \times \wp(T'')$ *associates corresponding* *transitions of both systems to each other. Let $T_1 \subseteq T'$ and $T_2 \subseteq T''$. If $T_1 \sim T_2$ then (T_1, T_2) is referred to as a* correspondence. *(T_1, T_2) is called* elementary*, iff $|T_1| = |T_2| = 1$, and* complex *otherwise.*

Further on, we discuss the structural relation between multiple correspondences between two systems by means of two structural properties. First, correspondences might be overlapping, i.e., they share a transition in at least one of the WF-systems.

Definition 3 (Overlapping Correspondences). *Let (N', M_i') and (N'', M_i'') be two WF-systems and $C_1 = (T_1, T_2)$ and $C_2 = (T_3, T_4)$ two correspondences between them, such that $T_1, T_3 \subseteq T'$ and $T_2, T_4 \subseteq T''$. C_1 and C_2 are called* overlapping*, iff $T_1 \cap T_3 \neq \emptyset$ or $T_2 \cap T_4 \neq \emptyset$.*

Regarding the example in Fig. 1, we see that all correspondences are non-overlapping.

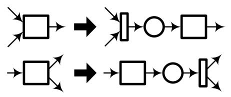


Fig. 3. Pre-processing

Second, we provide an additional property for the structural relation between two correspondences, which is based on the RPST. Here, we have the assumption that all transitions that are part of a correspondence have a single incoming and single outgoing flow arc. If not, a behaviour preserving pre-processing is done, cf., Fig. 3. Note that the systems in Fig. 1 are pre-processed. The property requires that two correspondences are not overlapping in terms of their lowest enclosing fragments in the RPST.

Definition 4 (Structurally Independent Correspondences). *Let (N', M_i') and (N'', M_i'') be two WF-systems, $\mathcal{R}'_{N'} = (\Omega', \chi', t')$ the RPST of N', and $C_1 = (T_1, T_2)$ and $C_2 = (T_3, T_4)$ with $T_1, T_3 \subseteq T'$ and $T_2, T_4 \subseteq T''$ two correspondences. C_1 and C_2 are* structurally independent *in N', iff $|T_1| = |T_3| = 1$ or $\forall\ \omega = (P_\omega, T_\omega, F_\omega) \in \Omega'$ $[\ T_1 \cap T_\omega \neq \emptyset \wedge T_3 \cap T_\omega \neq \emptyset \Rightarrow T_1 \subseteq T_\omega \wedge T_3 \subseteq T_\omega\ ]$.*

Our example in Fig. 1 illustrates correspondences that are structurally independent in one or both systems. For instance, the pair of correspondences $C1$ and $C2$ is independent in model (a), as every fragment containing transition A and either B or C contains all three transitions, cf., Fig. 2. However, the very same pair of correspondences is not independent in model (b), as there is a bond fragment that represents a subnet comprising transitions 3, 4, and 5, but which does not contain transition 1. In contrast, $C1$ and $C4$ are an example for a pair of correspondences that are structurally independent in both models (a) and (b).

4 Behavioural Analysis of Correspondences

Based on the structural properties of correspondences introduced in the previous section, this section discusses behavioural analysis of correspondences. First, we elaborate on the behavioural ambiguity of overlapping correspondences. Second, the application of common notions of behaviour inheritance in the context of complex correspondences is discussed. Subsequently, we introduce heuristics to detect violations of behaviour inheritance for structurally independent correspondences.

Overlapping Correspondences. Overlapping correspondences raise various questions regarding their intended meaning. That is, it is unclear whether an occurrence of a transition that is part of both overlapping correspondences should be considered for one or for both correspondences. For instance, consider the systems depicted in Fig. 4. Here, the trace $A1, X$ in the lower system might correspond to both, the occurrence of transition A only, or the occurrence of both transitions, A and B, in the upper system. Thus, the semantic ambiguity of such overlapping correspondences has to be addressed as a prerequisite for any behavioural analysis.

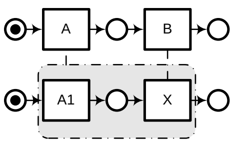


Fig. 4. Overlapping Correspondences

Behaviour Inheritance. In order to judge about behaviour equivalence in the context of correspondences between process models, first and foremost, the treatment of transitions without counterparts has to be addressed. To this end, two notions of behaviour inheritance can be distinguished, i.e., protocol inheritance or projection inheritance, cf., [10, 11]. That is, transitions in one model that are without counterpart in the second model are either blocked (protocol inheritance) or hidden (projection inheritance). Although these notions have been defined based on branching bisimulation, they can easily be transferred to trace-based behaviour analysis. It is worth to mention that the term inheritance is slightly misleading in our context as it implies a directed relation between two models (one model is a subclass of the other model). As illustrated by our example in Fig. 4, such a directed relation might not exist between aligned models.

The choice between protocol inheritance or projection inheritance is independent of the structural characteristics of the correspondences. However, we focus on projection inheritance in the context of this paper. In particular, this notions seems to be more appropriate when comparing models that focus on different modelling perspectives (e.g., organisational vs. technical) as intermediate process steps that are mandatory in one perspective may be without counterpart in the other model. For instance, transition F of model (a) in Fig. 1 is not part of any correspondence even though its execution is required for completion of process (a). Thus, any notion of protocol inheritance would be bound to failure.

In the general case, projection inheritance in terms of trace semantics between two models is assessed as follows. The language of both process models is checked for equivalence, when all correspondences are *resolved*. In case of 1:1 elementary correspondences this resolution is straightforward as an occurrence of one transition in one model corresponds to the occurrence of another transition in the other model. For the case of complex correspondences, this resolution involves a replacement of all valid subtraces involving the transitions of the correspondence in the one model with all valid subtraces comprising the corresponding transitions in the other model. Consider the example in Fig. 1 and neglect correspondence $C2$ and $C3$ for the time being. Then, the traces A, D and A, E in model (a) (all other transitions are hidden) would be rewritten as follows. According to the correspondences, an occurrence of A is replaced by the subtraces $1, 3, 4$ or $1, 4, 3$, while the subtraces D and E are replaced by an occurrence of transition 7. That yields the two traces $1, 3, 4, 7$ and $1, 4, 3, 7$. As both traces described the language of model (b) when taking only transitions of correspondences $C1$ and $C4$ into account, we conclude that both correspondences do not violate projection inheritance. Note that during this analysis, the interleaving of transitions belonging to different correspondences has to be considered. Based thereon, each pair of correspondences can be assessed. If all correspondences show pairwise projection inheritance, projection inheritance for the two models and their alignment in terms of a set of correspondences can be concluded.

Structural Characterisation of Violations. While the aforementioned approach of comparing the set of all resolved traces can always be taken, violations of projection inheritance may already be discovered using a heuristic approach. It uses the RPST decomposition technique for sound free-choice WF-systems and is applicable for correspondences that are structurally independent in both systems (cf., Section 3). In previous work, we showed that the RPST of sound free-choice WF-systems can be annotated with behavioural details. That is, bond fragments are either transition bordered or place bordered, but there cannot be a mix of place and transition boundary nodes [12]. The former represents a parallel block, whereas the latter represents either an exclusive block or a loop fragment.

This information, in turn, can be leveraged to identify violations of projection inheritance that are induced by correspondences. Given two correspondences we proceed as follows.

1. Determine the lowest common ancestor (LCA) in the respective RPST of all trivial T fragments for which the entry node is a transition aligned by one of the two correspondences. Derive the LCA in both models.
2. Compare the type of the LCA fragments.
   - If one LCA fragment is of type P, the other LCA fragment has to be of type P as well and the order of the respective child fragments (transitively) containing all T fragments of the correspondences has to coincide in both LCA fragments.
   - If one LCA fragment is of type B and not cyclic, the other LCA fragment has to be of type B and acyclic as well. In addition, the fragments are either both place bordered or both transition bordered.
   - If one LCA fragment is of type B and cyclic, the other LCA fragment has to be of type B and cyclic as well.

This observation can be explained as follows. As both correspondences are structurally independent, the LCA of all T fragments that are related to the transitions of the correspondence must have at least two children, one (transitively) containing all T fragments related to one correspondence and the other one (transitively) containing all T fragments related to the other correspondence. Therefore, the type of the LCA induces a constraint on the potential order of occurrence between all transitions of one correspondence and all transitions of the other correspondence. This constraint has to hold in the second model as well. For instance, if there is a sequential order (LCA is P fragment) between the respective T fragments (and, therefore, the transitions) of the two correspondences in one model, there cannot be a potential concurrent enabling (LCA is B fragment that is transition bordered) of the corresponding transitions in the other model. Such a setting will never show projection inheritance. It is important to see two limitations of this approach. On the one hand, conclusions cannot be drawn, if one of the two LCAs is a rigid (R) fragment, as the constraints on the order of potential execution cannot be derived directly. On the other hand, this check is solely a heuristic that formulates a necessary, but not a sufficient condition for a violation of projection inheritance. Even if the type of the LCA is equal, projection inheritance might be violated by the two correspondences. For instance, if both LCAs are of type P, it might still be the case that the occurrence of all transitions of one correspondence is optional in one model, whereas it is mandatory in the other model.

Besides $(C1, C2)$ and $(C1, C3)$, all pairs of correspondences in Fig. 1 are structurally independent in both models, such that the types of the respective LCA fragments can be compared. That, in turn, yields the following result. For the pairs $(C1, C4)$, $(C2, C3)$, and $(C2, C4)$ the LCA fragments show a consistent type in both models. In contrast, for correspondence $(C3, C4)$ the LCA fragment is of type B (transition bordered) in model (a) and of type P in model (b). Thus, the structural analysis revealed that correspondences $C3$ and $C4$ violate projection inheritance between both models. Transitions of both correspondences might be enabled concurrently in model (a), which is not possible in model (b).

Regarding correspondences that are not structurally independent in both systems, we already discussed the case of correspondences $C1$ and $C2$ of our example in Fig. 1. These correspondences obviously violate projection inheritance. However, our example also illustrates that correspondences that are structurally dependent do not necessarily lead to violations. Consider, for instance, correspondences $C1$ and $C3$, which are not structurally independent in model (b), due to fragments that represent subnets containing transitions $3, 4, 5, 6$, but not 1. Nevertheless, the order between all transitions of correspondence $C1$ and $C3$ as imposed by model (a) is also reflected in model (b). Albeit beyond the scope of this discussion, we foresee that necessary conditions for violations can also be specified for the case of correspondences that are not structurally independent.

5 Related Work

We already discussed related work in the field of behaviour equivalence [4] and behaviour inheritance [10, 11]. Further on, work on trace-based assessment of similarity is related to our work. For instance, the set n-grams of possible traces two models can be compared [13]. In other work, a trace is replayed in a model and the number of valid replay steps is leveraged for measuring similarity [14].

Moreover, the assessment of behaviour inheritance in the presence of complex correspondences is related to work on behaviour preserving model refinements, see [15] for a thorough survey. A transition refinement is defined by replacing a transition with a block-structured refinement net with a dedicated set of initial and final transitions. However, other approaches allow refinement nets to also show so-called distributed input or distributed output places [15].

6 Conclusion

In this paper, we addressed the challenge of assessing behaviour equivalence for process models aligned by complex correspondences. In particular, we discussed how the notion of behaviour inheritance can be applied for complex correspondences and also introduced necessary conditions for violations of this notion by a pair of correspondences based on structural decomposition techniques for sound free-choice workflow systems.

As a next step, the discussed application of behaviour inheritance in the context of complex correspondences needs to be formalised. In addition, we want to further investigate the structural characterisation of correspondences and their relation to the notions of behaviour inheritance. Here, the goal would be the specification of structural characterisations that are not only a necessary, but also a sufficient condition for a violation of behaviour inheritance.

References

1. Dijkman, R.M.: Diagnosing differences between business process models. In: BPM. Volume 5240 of LNCS., Springer (2008) 261–277
2. Dijkman, R., Dumas, M., García-Bañuelos, L., Käärik, R.: Aligning business process models. In: EDOC. (2009) 45–53
3. Nejati, S., Sabetzadeh, M., Chechik, M., Easterbrook, S.M., Zave, P.: Matching and merging of statecharts specifications. In: ICSE, IEEE Computer Society (2007) 54–64
4. van Glabbeek, R.: The linear time - brancing time spectrum I. The semantics of concrete, sequential processes. In: Handbook of Process Algebra. Elsevier (2001) 3–99
5. Aalst, W.: The application of Petri nets to workflow management. Journal of Circuits, Systems, and Computers **8**(1) (1998) 21–66
6. Desel, J., Esparza, J.: Free Choice Petri Nets. Cambridge University Press (1995)
7. Aalst, W.: Verification of workflow nets. In: ICATPN. Volume 1248 of LNCS. (1997) 407–426
8. Vanhatalo, J., Völzer, H., Koehler, J.: The refined process structure tree. In: BPM. Volume 5240 of LNCS. (2008) 100–115
9. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer-Verlag (2007)
10. Basten, T., Aalst, W.: Inheritance of Behavior. Journal of Logic and Algebraic Programming **47**(2) (2001) 47–145
11. Schrefl, M., Stumptner, M.: Behavior-consistent specialization of object life cycles. ACM Trans. Softw. Eng. Methodol. **11**(1) (2002) 92–148
12. Weidlich, M., Polyvyanyy, A., Mendling, J., Weske, M.: Efficient computation of causal behavioural profiles using structural decomposition. In: ICATPN. (2010) accepted for publication.
13. Wombacher, A.: Evaluation of technical measures for workflow similarity based on a pilot study. In: OTM (1). Volume 4275 of LNCS., Springer (2006) 255–272
14. de Medeiros, A.K.A., van der Aalst, W.M.P., Weijters, A.J.M.M.: Quantifying process equivalence based on observed behavior. Data Knowl. Eng. **64**(1) (2008) 55–74
15. Brauer, W., Gold, R., Vogler, W.: A survey of behaviour and equivalence preserving refinements of petri nets. In: ICATPN. Volume 483 of LNCS., Springer (1989) 1–46