Efficient Consistency Measurement based on Behavioural Profiles of Process Models

Matthias Weidlich, *Member, IEEE,* Jan Mendling, and Mathias Weske, *Member, IEEE*

Abstract—Engineering of process driven business applications can be supported by process modelling efforts in order to bridge the gap between business requirements and system specifications. However, diverging purposes of business process modelling initiatives have led to significant problems of aligning related models at different abstract levels and different perspectives. Checking the consistency of such corresponding models is a major challenge for process modelling theory and practice. In this article, we take the inappropriateness of existing strict notions of behavioural equivalence as a starting point. Our contribution is a concept called behavioural profile that captures the essential behavioural constraints of a process model. We show that these profiles can be computed efficiently, i.e., in cubic time for sound free-choice Petri nets w.r.t. their number of places and transitions. We use behavioural profiles for the definition of a formal notion of consistency, which is less sensitive to model projections than common criteria of behavioural equivalence and allows for quantifying deviation in a metric way. The derivation of behavioural profiles and the calculation of a degree of consistency have been implemented to demonstrate the applicability of our approach. We also report the findings from checking consistency between partially overlapping models of the SAP reference model.

Index Terms—Process model analysis, process model alignment, behavioural abstraction, consistency checking, consistency measures

✦

1 INTRODUCTION

TRANSLATING business requirements into a system specification is a crucial task of any software engineering project in a business environment. A major challenge in this area are the different perspectives that business analysts and system analysts take on the same real-world phenomenon in an enterprise. The modelling of business processes has been identified as an important step towards bridging the gap between business and software development, and, among others, facilitating structured design [1], business-IT alignment [2], or engineering of process-aware information systems [3]. There are different solutions that should contribute to a smooth progression from business analysis to software implementation. Methodologies for integrated system design propose to derive technical realisations from business requirements directly via refinements [4], [5], [6]. In the same vein, the standardisation of the Business Process Modeling Notation (BPMN) [7] by the OMG received much attention, due to the translation to the Web Services Business Process Execution Language (BPEL) [8] that is part of the specification. There are also various tools on the market that support business process modelling and corresponding transformations. Still, the well-known 'Business-IT-Gap' (cf., [4], [9], [10]) appears to persist in practice.

There are different factors that make the alignment of business and IT a tedious task (see [2]). In this article, we will focus on a specific challenge that is of a conceptual nature. We already mentioned that business analysts and software designers tend to model the same business process in quite different ways, which often impedes efficient communication. Clearly, pragmatics is an inherent feature of every conceptual model. While mapping and reducing the reality are essential for creating a model, the *purpose* of the model determines what to map and what to reduce [11]. As business analysts and software designers have quite diverging concerns when looking at a business process, it is no surprise that business process models differ from software design models of the same process significantly. In fact, these models are not related by any refinement operation in the sense of an integrated system design as mentioned above. Thus, the question of how well such models are aligned is fundamental to process-oriented software development. We will argue throughout this article that a formal concept for discussing the *consistency* of an alignment between two process models is missing. It is needed for identifying inconsistencies, as well as to enable change propagation between these models. In the software engineering community, consistency refers to a *'degree of uniformity, standardisation, and freedom of contradictions'* [12]. Evidently, there is a trade-off between strictness of a consistency notion and appropriateness of process models serving different purposes.

There is some prior research that is useful for establishing a notion of consistency between process models. An alignment of process models requires the identification of model *correspondences*, which is a well-researched topic in the database community. Correspondences relate elements that have matching semantics in the context of an

---

- *M. Weidlich and M. Weske are with the Hasso Plattner Institute, Potsdam, Germany.*
  E-mail: {matthias.weidlich,weske}@hpi.uni-potsdam.de
- *J. Mendling is with Humboldt-Universität zu Berlin, Germany.*
  E-mail: jan.mendling@wiwi.hu-berlin.de

alignment of two models (note that the semantics might differ in absolute terms). Given a set of correspondences, the question whether two data models are consistent is similar to the question whether a mapping between data schemas is valid, which is known from the field of data integration. In this area, various properties for evaluating the validity of a schema mapping have been proposed. For instance, *satisfiability* of a mapping between two schemas requires the existence of a pair of instances that satisfies the constraints of the mapping as well of the respective schemas [13]. Translated into the domain of behavioural models, this yields a consistency notion, which requires the existence of a single trace that is possible in both models after the corresponding elements have been resolved. Obviously, this is a rather weak requirement. A stronger property for a schema mapping is *losslessness*, which requires all data elements needed to answer a certain query in one schema to be captured by the mapping and have a counterpart in the other schema. Again, we might draw the analogy to behavioural models under the assumption of behavioural constraints as the elementary elements. For a given projection between two models, that is a partial correspondence relation, all behavioural constraints on traces of one model are preserved in the traces of the other model.

Although the context is different, these examples illustrate the existence of rather weak validity criteria for schema mappings. In contrast, existing work in the field of behavioural models focusses on very strict notions of behavioural equivalence. There is a multitude of equivalence criteria in the linear time – branching time spectrum [14]. However, the *lower bound* of this spectrum is commonly seen as being trace equivalence. This criterion is still rather strict, and might not be appropriate for deciding on a consistent alignment between two process models. First, trace equivalence is not invariant to so-called *forgetful refinements* of activities [15]. Forgetful refinement refers to a change, in which an activity is forgotten due to its replacement with an empty activity. In other words, an activity is subject to projection. However, projections are a substantial aspect of model abstractions [11] and, therefore, of high relevance for model alignment. As a consequence, we argue that, for our purpose, a notion of *consistency* that guarantees *'freedom of contradictions'* as required by [12] should be less strict than a notion requiring all information of one model to be present in another model as well. Furthermore, all behavioural equivalence criteria being discussed in this area, including trace equivalence, provide a true/false result. Such results are not informative for an alignment scenario. It is likely that a software designer might deviate from the requirements defined in a business process model in order to come up with a more elegant solution on a technical level. Those design decisions may be acceptable if they deviate from the business process model only to a small degree. Any boolean notion will fail to make such a small deviation explicit. Following on the idea of trace equivalence, potential deviations can be quantified using a degree of trace consistency computed based on the ratio of traces of one model that can be mirrored in another model. Still, a relatively small deviation in the process model structure (e.g., interchanging two sequential activities) impacts on this degree of trace consistency drastically. Finally, all notions of the linear time –- branching time spectrum are computationally hard [14]. This is a problem since process models from practice can include easily more than 100 activities. This makes the application of trace equivalence and other criteria in many interactive modelling scenarios unrealistic.

Against this background, this article provides the following contributions. We introduce the formal concept of a behavioural profile. These profiles capture the essential behavioural constraints of a process model, such as mutual exclusion of activities or partial order. The behavioural profile enables us to overcome three major shortcomings of an application of trace equivalence in an alignment scenario.

1) Behavioural profiles are less sensitive to projections than trace equivalence. We will show that behavioural profiles of two process models remain unchanged even if additional start and end branches are introduced in one of the models. While informal arguments on this advantage of behavioural profiles have been brought forth in [16], we here present a rigorous formalization.
2) The structure of a behavioural profile provides us with a straight-forward way to define a degree of consistency ranging from 0 to 1.0, referred to as the degree of profile consistency. In this way, we can feed back detailed information to business analysts and software designers on how far and where two models deviate from each other.
3) The concept of a behavioural profile builds on formal properties of free-choice Petri nets. This class of nets has been used for the formalization of most process modelling languages. We prove in this article that profile consistency can be checked for sound free-choice Petri nets in $O(n^3)$ time with n being the number of places and transitions.

The derivation of a behavioural profile and the calculation of a degree of profile consistency have been implemented to demonstrate the applicability of our approach. In this article, we also report the findings from checking consistency between partially overlapping process models of the SAP reference model, a collection of industry process models that describe the functionality of the SAP business software.

The remainder of this article is structured as follows. Section 2 illustrates the alignment of process models using an example and discusses the weaknesses of trace equivalence as a consistency criterion. Formal preliminaries on Petri nets are introduced in Section 3. In Section 4, we provide definitions and theorems on behavioural profiles. Their application is illustrated in

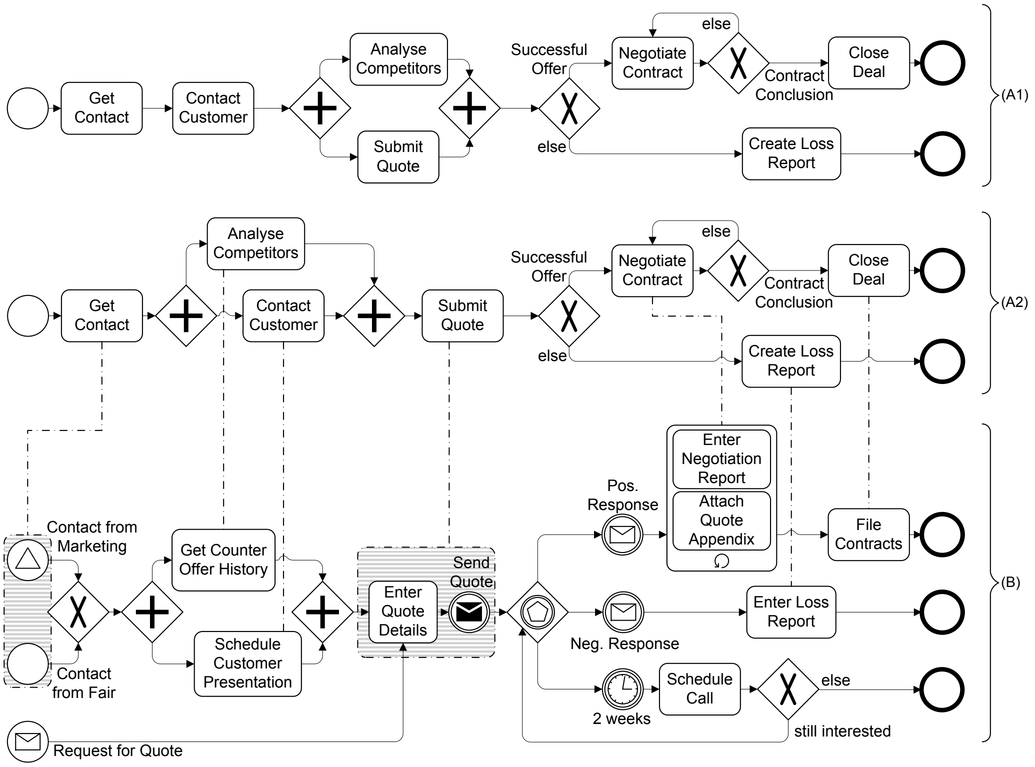


Fig. 1. Lead-to-Order scenario that is captured on two different abstraction levels

Section 5, where we define different consistency measures. We report on our findings of applying these measures to the SAP reference model in Section 6. Finally, we review related work in Section 7 and conclude in Section 8.

2 CONSISTENCY BETWEEN ALIGNED PROCESS MODELS

In general, the alignment problem addresses the question whether two process models are consistent given a set of correspondences between their elements. This problem can be further classified. Inspired by the notions of vertical and horizontal process integration [17], we also distinguish *vertical* and *horizontal* alignment depending on the assumed abstraction level of the respective process models. Business-centred and technical process models will typically assume different abstraction levels, which results in vertical alignment. Horizontal alignment, in turn, is needed between different process variants on the same level of abstraction. Such variations might be due to business strategies varying in different countries or for different products, as well as differing legal obligations or enterprise systems in use [18]. The issue of consistency for an alignment is, however, independent of the distinction of vertical and horizontal alignment. That is illustrated by Fig. 1, which depicts three process models using BPMN. All models describe a lead-to-order process, which we encountered in the course of an industry cooperation.

Both upper process models (A1) and (A2) can be seen as typical high-level process models. They depict the *major path* of the process from the initial contact with the customer to the closing of the deal. The purpose of these models can be seen in giving an intuitive overview of the major processing steps. Further on, branching points (e.g., if the offer was successful) that are of relevance for decision making or performance evaluations are modelled explicitly. Without doubt, both models (A1) and (A2) are very similar, as they contain exactly the same set of activities and control flow elements. However, we see that there is a slight semantic difference between them. While in the lead-to-order scenario in model (A1) the analysis of competitors happens while the quote is submitted in order to prepare for the negotiation phase, the corresponding activity happens while the customer is contacted in model (A2). Here, information on competitors is a preliminary step for the creation and submission of the quote. Thus, both models define different *variations* of the lead-to-order process that show a horizontal alignment.

The third process model (B), depicted in the lower part of Fig. 1, describes the same business case more fine-grained and from a different perspective. Instead of focussing on the manual tasks to be done, model (B) depicts the scenario from the perspective of a supporting IT-system. That is, the scope comprises invoked services and the functionality provided by the system that is used to realise the lead-to-order process. Obviously, process model (B) is highly related to the other two models, even though the differences are bigger than between models (A1) and (A2). Thus, we would not consider model (B) as a variant of the other processes due to the different abstraction level and assumed perspective. However, model (B) and the upper two models are aligned in a vertical manner, which is manifested in corresponding activities. These correspondences are highlighted using dash-dotted lines between model (A2) and (B). Compared to the upper models, we see that process instantiation semantics have been specialised in the course of model refinement in process model (B). That is, activity *Get Contact* of model (A2) corresponds to one of the start events, *Contact from Marketing* or *Contact from Fair*, respectively. Furthermore, the activity of submitting a quote has been refined and the possibility of receiving a requests for quote directly has been considered. Another major difference is the distinction of potential answers to the quote from the customer. Here, not only the *major path*, but also the case of a time out for the customer response is modelled.

Assessing the consistency of this scenario based on common notions of behavioural equivalence does not seem to be appropriate. Consider, for instance, the application of trace equivalence. Applying this criterion to check consistency of the horizontal alignment of the upper models (A1) and (A2) is straight-forward and yields a negative result. The submission of the quote might happen before the analysis of competitors only in model (A1) but not in model (A2). Arguably, the two models are not free of contradictions. Nevertheless, the high similarity of the two process variants is not reflected in this Boolean result. In addition, an assessment of the ratio of complete traces of one model that can be mirrored in the other model, yielding the degree of trace consistency, suffers from the same problem. First, both models show infinite sets of traces due to the loop in the process structure. Once the loop is neglected, still, various execution sequences of each model cannot be mirrored in the other model due to the different dependencies for activity *Analyse Competitors*. That, in turn, highlights the inappropriateness of the trace equivalence criterion or the degree of trace consistency in order to investigate the consistency of process variants.

Turning the focus towards the consistency of the vertical alignment between the lower model (B) and the upper two models, an application of the trace equivalence criterion (or the degree of trace consistency, respectively) raises various questions. In order to compare the set of possible traces of both process models, first and foremost, all parts that have been subject to projection (in either direction) have to be discarded. An example would be activity *Schedule Call* and its preceding timer event in the lower model. As they have no counterparts in the upper models, they are removed from the traces based on which equivalence is decided. Secondly, correspondence links between multiple activities have to be treated. It is important to notice that such correspondences might have different semantics. Consider the aforementioned refinements of activity *Get Contact* and activity *Submit Quote* of the upper models (A1) and (A2). The former corresponds to *one* of the start events, whereas the latter corresponds to *both*, activity *Enter Quote Details* and the event *Send Quote*. Obviously, these different semantics have to be taken into account when deciding trace equivalence. In previous work, we already explicated on the whole spectrum between a conjunction and a disjunction of corresponding elements that might have to be dealt with (e.g., an activity is split up into a set of activities out of which an arbitrary subset might occur in a certain trace) [19]. Even in case such refinements are handled, the different instantiation and termination semantics of both process models (caused by model projections) violate trace equivalence and impact on the degree of trace consistency. While the activity *Get Contact* is part of *every* trace of process models (A1) and (A2), none of the corresponding events has to be executed in process model (B), owing the possibility to instantiate the process via the event *Request for Quote*. Similarly, the third end state in process model (B), i.e., the potential customer has been contacted after 2 weeks, but they are no longer interested, implies that neither activity *Enter Loss Report*, nor activity *File Contracts* has to be executed in a complete trace. In contrast, one of the corresponding activities is required in every complete trace of process model (A1) and (A2), respectively. Obviously, these projections of activities (or complete process branches) are due to the different modelling perspectives. For instance, the aforementioned time out has been neglected in the high-level process models (A1) and (A2), whereas it is considered to be crucial for the IT-support depicted in model (B). Therefore, we argue that behavioural differences that stem from such projections should not impact on the consistency assessment.

We summarise that none of the exemplary process models are trace equivalent with respect to the corresponding activities. As solely a few out of all traces of one model can be mirrored in a second model, the degree of trace consistency of a pair of models will be low. Therefore, we consider these notions to be inappropriate in order to decide consistency in the context of vertical and horizontal alignment.

3 PRELIMINARIES

For investigating process model consistency, we use workflow (WF-) systems [20] as our formal grounding. On the one hand, this class of Petri nets has been applied for process modelling for over a decade [20]. On the other hand, Petri net based formalisations have been presented for (at least parts of) most common process modelling languages, such as BPEL, BPMN, EPCs, YAWL, and UML models. For instance, the Business Process Execution Language (BPEL) has been formalised using open workflow nets [21] and a subset of the Business Process Modeling Notation (BPMN) is defined using standard Petri nets in [22]. Such a Petri net might be transformed into a WF-system, if there is a dedicated set of end states. Moreover, there are various Petri net based formalisations for Event-Driven Process Chains (EPCs) (see [23]). Again, Petri nets derived from EPCs can be transformed into WF-systems, if there is an agreement on instantiation semantics for certain ambiguous cases (e.g., multiple start events lead to an OR-join). Also YAWL heavily relies on Petri net concepts [24]. In the context of the Unified Modeling Language (UML), semantics have also been defined for Sequence Diagrams based on M-nets [25]. As an M-net requires one dedicated initial marking and end marking, it might also be transformed into a WF-system. A Petri net formalisation is also available for UML Activity Diagrams [26].

A comparative survey of formalisations for BPMN, EPCs, YAWL, and BPEL can be found in [27]. According to the authors, the formalisation of OR-join constructs and exception handling are hard to handle. The former often cannot be modelled at all due to the need for non-local semantics (e.g., in BPMN). However, we will see that for the computation of a behavioural profile, block-structured OR-joins can be traced back to AND-joins, which, in turn, enables formalisation using standard Petri net constructs. Exception handling has been tackled for BPMN and BPEL (cf., [21], [22]) using standard Petri nets, whereas the formalisation of YAWL relies on reset nets [24]. Still, formalising exception handling typically does not yield WF-nets that show the free-choice property, which we require for our efficient approach of computing a degree of consistency based on behavioural profiles.

Based on [20], we recall some basic definitions. 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. The set of all preceding nodes for $x \in (P \cup T)$ is defined as $^{\bullet}x := \{y \in (P \cup T) \mid (y, x) \in F\}$, whereas $x^{\bullet} := \{y \in (P \cup T) \mid (x, y) \in F\}$ denotes its succeeding nodes. Further on, let F^{+} be the transitive closure of F, that is $xF^{+}y$ if there is a path from x to y.

The *marking* of a net is a bag over the set of places, i.e., it is a function from P to the natural numbers. We use the notation s_j for the following markings. If j is a place, s_j is the marking that puts a token in j with no tokens elsewhere. If j is a transition, s_j is the marking that puts one token in all places $^{\bullet}j$. A transition $t \in T$ is *enabled* in a marking s, denoted by $(N, s)[t\rangle$, iff $^{\bullet}t \leq s$. *Firing* of t in s leads to a new marking $s' = s -^{\bullet} t + t^{\bullet}$, denoted by $(N, s)[t\rangle(N, s')$. A *firing sequence* of length $n \in \mathbb{N}$ is a function $\sigma : \{0, \ldots, n-1\} \mapsto T$ that specifies a sequence of transitions that can be fired in sequential order (please refer to [20] for further details). The set of all sequences of arbitrary length over T is denoted by T^{*}. For a firing sequence $\sigma = \{(0, t_x), \ldots, (n-1, t_y)\}$ we use $\sigma = t_0, \ldots, t_{n-1}$ as a short hand notation. Moreover, we write $t_j \in \sigma$, if t_j is an element of the firing sequence.

A *system* $S = (N, m)$ is given by a net N and an initial marking m. The set of all reachable markings of S is denoted by $[N, m\rangle$. A *workflow (WF-) net* is a net $N = (P, T, F)$ with a dedicated input place $i \in P$ (it is the only place $p \in P$ with $^{\bullet}p = \emptyset$) and a dedicated output place $o \in P$ (it is the only place $p \in P$ with $p^{\bullet} = \emptyset$). In addition, the short-circuit net, derived by inserting a transition between the initial and the final place, $N' = (P, T \cup \{t_c\}, F \cup \{(o, t_c), (t_c, i)\})$ for N has to be strongly connected. A *workflow (WF-) system* $S = (N, [i])$ is a system with N being a WF-net.

For the remainder of this article, we always assume the net of a WF-system to be defined as $N = (P, T, F)$ without stating it explicitly. Workflow systems are means to model process models, therefore, we use the terms *activity* and *transition* synonymously.

4 BEHAVIOURAL PROFILES OF PROCESS MODELS

In this section, we introduce the concept of a *behavioural profile* and its calculation from a WF-system. The notion of a behavioural profile provides the foundation to reason about consistency of a pair of process models. As most process modelling languages such as BPMN, EPCs, or UML Activity Diagrams can be mapped to Petri nets (at least partially), we can also derive the behavioural profile for them. The general idea is to trace back the behaviour of a process model to characteristic relations, which capture dedicated behavioural aspects like exclusiveness of a pair of activities or their order of potential occurrence. These relations, in turn, yield the behavioural profile of a process model. We dedicate Section 4.1 to a formal definition of these relations. Afterwards, we investigate the relation properties in Section 4.2 and show how they can be derived efficiently in Section 4.3. Finally, we explicate the differences between equivalence of behavioural profiles and trace equivalence in Section 4.4.

4.1 Behavioural Relations

In general, we can distinguish three fundamental relations between activities of a process model. The execution of two activities might happen either in *strict order*, *exclusively*, or in *interleaving order*. These relations state *potential* dependencies. The actual execution of an activity is not explicitly enforced. With respect to our formal model, we capture these characteristic dependencies as relations between WF-system transitions that are identified based on the existence of a certain firing sequence.

A similar, but not equivalent set of relations has been proposed in the context of workflow mining [28]. We base our definitions on the notion of an *indirect* weak order dependency, whereas the ordering relations in [28] are grounded on a *direct* sequential order. Here, a direct dependencies refers to a relation that is defined for activities that succeed each other without other activities being executed in between. In the context of workflow mining, *causal* dependencies in the sense of one activity directly succeeding another are emphasized. As a result, for instance, the notion of exclusiveness is restricted to *'pairs of transitions that never follow each other directly'* [28]. In contrast, we aim at capturing exclusiveness also for activities that might occur at different stages of a firing sequence. Furthermore, we can build on more explicit information to derive the relations as we take process models as a starting point, and not a set of traces (logs).

All of our behavioural relations are grounded on the concept of weak order. Therefore, we first introduce the weak order dependency as an auxiliary relation.

Definition 1 (Weak Order Relation): Let $(N, [i])$ be a WF-system. The *weak order relation* $\succ \subseteq T \times T$ contains all pairs (x, y), such that there exists a firing sequence $\sigma = t_1, \ldots, t_n$ with $(N, [i])[\sigma\rangle$, $j \in \{1, \ldots, n-1\}$, and $j < k \leq n$ for which holds $t_j = x$ and $t_k = y$.

Based thereon, we define the first behavioural relation, that is strict order, between two transitions.

Definition 2 (Strict Order Relation): Let $(N, [i])$ be a WF-system. The *strict order relation* $\rightsquigarrow \subseteq T \times T$ contains all pairs (x, y) with $x \succ y$ and $y \not\succ x$.

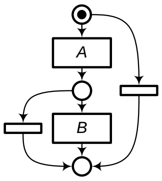


Fig. 2. *Strict order*

As mentioned before, the strict order relation enforces neither the occurrence of the first transition, nor of the second transition. Thus, for the two transitions in Fig. 2, it holds $A \rightsquigarrow B$. The second relation, which captures the exclusive occurrence of two transitions is captured as follows.

Definition 3 (Exclusiveness Relation): Let $(N, [i])$ be a WF-system. The *exclusiveness relation* $+ \subseteq T \times T$ contains all pairs (x, y) with $x \not\succ y$ and $y \not\succ x$.

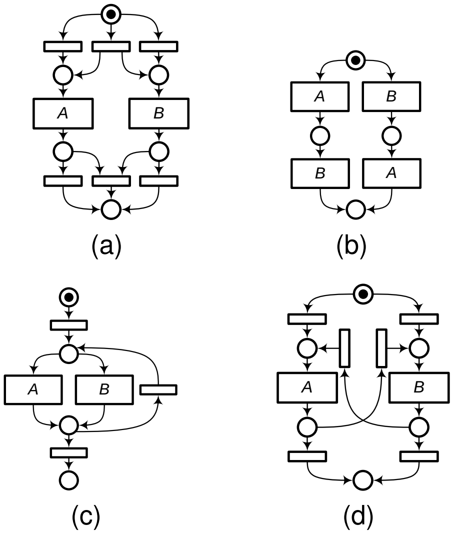
In order to capture the absence of any order between the potential occurrences of two activities, we introduce the interleaving order relation.

Definition 4 (Interleaving Order Relation): Let $(N, [i])$ be a WF-system. The *interleaving order relation* $|| \subseteq T \times T$ contains all pairs (x, y) with $x \succ y$ and $y \succ x$.

The notion of transitions in interleaving order does not necessarily imply the concurrent enabling of two transitions in a certain marking. For instance, two transitions that are preceded by an inclusive OR-split as illustrated by the (not sound) system in Fig. 3(a) will be in the interleaving order relation, i.e., $A||B$. One might also think of interleaving order based on the interleaving of labelled transitions as depicted in Fig. 3(b). However, that, in turn, requires the definition of a WF-system to contain a labelling function that assigns labels to transitions. As a result the occurrence of a label in a trace might result from firing of different transitions. Such a labelling is not part of our formal model, which excludes the systems in Fig. 3(b). Further on, interleaving order of two transitions might also result from cyclic structures. In Fig. 3(c) and Fig. 3(d), it also holds $A||B$, as the observable firing sequences contain both transitions in any order.

The three behavioural relations introduced above form the behavioural profile of a process model.

Definition 5 (Behavioural Profile): For a WF-system $(N, [i])$ the set of behavioural relations $\mathcal{BP} = \{\rightsquigarrow, +, ||\}$ is referred to as the *behavioural profile* of $(N, [i])$.



(a) (b) (c) (d)

Fig. 3. Transitions in interleaving order

4.2 Properties of Behavioural Relations

The definition of the relations of the behavioural profile implies their mutual exclusiveness. This property also holds for the similar set of relations from workflow mining [28].

Property 1: For any WF-system $(N, [i])$ holds that the behavioural relations $\rightsquigarrow$, $+$, and $||$ are mutually exclusive and $\rightsquigarrow$, $\rightsquigarrow^{-1}$, $+$, and $||$ partition $T \times T$.

Let $\succ^{-1} = \{(y, x) \in T \times T \mid x \succ y\}$ the inverse relation for $\succ$. Then, the property can be verified by the definition of the behavioural relations as $\rightsquigarrow = \succ \setminus \succ^{-1}$, $\rightsquigarrow^{-1} = \succ^{-1} \setminus \succ$, $+ = (T \times T) \setminus (\succ \cup \succ^{-1})$, and $|| = \succ \cap \succ^{-1}$.

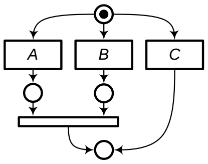


Fig. 4. Deadlocking WF-system

This property holds also in case a WF-system shows behavioural anomalies such as deadlocks (cf., Fig. 4). Even in case the initial marking is a deadlock, that is, there is not a single firing sequence, all transitions would be considered to be exclusive, i.e., $+ = (T \times T)$ owing to $\succ = \succ^{-1} = \emptyset$.

In the previous section, we saw that cyclic structures have a substantial impact on the behavioural relations. For instance, two exclusive transitions inside a cycle are considered to be in interleaving order, which is depicted in Fig. 3(c) and Fig. 3(d). Therefore, we have to address the question of how these cycles are reflected in the behavioural profile.

Property 2: For any WF-system $(N, [i])$ holds that a transition $t \in T$ is either said to be exclusive to itself $(t + t)$ or in interleaving order to itself $(t||t)$.

In order to verify this property, we have to consider two cases, $(t, t) \in \succ$ and $(t, t) \notin \succ$. For the former it holds that $(t, t) \in \succ$ implies $(t, t) \in \succ^{-1}$, which yields $t||t$. For the latter, we have the opposite implication, i.e., $(t, t) \notin \succ$ implies $(t, t) \notin \succ^{-1}$ and, therefore, $t + t$. Consequently, the behavioural relations capture, whether a transition might occur at most once ($t + t$), or might be repeated due to a control flow cycle ($t||t$). For instance, $A + A$ in Fig. 3(a) and $A||A$ in Fig. 3(c).

4.3 Deriving the Behavioural Relations

The relations of the behavioural profile follow directly from the weak order relation, which requires the existence of a certain firing sequence. Such a firing sequence is one path in the reachability graph of the system. Therefore, the whole state space of the system has to be considered, which is well-known to require exponential space and time for arbitrary Petri nets [29], [30].

However, for a dedicated class of systems, that is *sound free-choice* WF-systems, a different approach can be taken to determine the behavioural profile. Before we describe the approach, we recall basic definitions for this class of systems.

According to [31], a net $N = (P, T, F)$ is *free-choice*, if $(p, t) \in F$ implies $^\bullet t \times p^\bullet \subseteq F$ for every place p and transition t. A system $(N, [i])$ is free-choice, if N is free-choice. A system $(N, [i])$ is *live*, if for every reachable marking $s \in [N, [i]\rangle$ and $t \in T$, there is a marking $s' \in [N, s\rangle$ such that $(N, s')[t\rangle$. A system $(N, [i])$ is *bounded*, iff the set of reachable markings $[N, [i]\rangle$ is finite. It has been shown that liveness and boundedness are closely related to the *soundness* criterion, which requires a WF-system (1) to always terminate, and (2) to have no dead transitions (note that proper termination is implied for WF-systems) [32]. In fact, a system is *sound*, if and only if the corresponding short-circuit system is live and bounded [32]. Further on, there is an important dependency between the structure of a sound free-choice system and its semantics. That is, the existence of a path from a place q to a place p with s_p being a home marking (a marking which is reachable from every marking reachable from the initial state) implies the existence of a firing sequence containing all transitions on the path between q and p (Lemma 4.2 in [33]). Due to liveness of the short-circuit system $(N', [i])$, all markings reachable from the initial state $[i]$ in N are home markings in $(N', [i])$.

We start by deriving the interleaving order relation. We first define an auxiliary relation that captures concurrent enabling of two transitions.

Definition 6 (Concurrency Relation): Let $(N, [i])$ be a WF-system. The *concurrency relation* $||_c \subseteq T \times T$ contains all pairs (x, y) with $x \neq y$, such that there is a reachable marking $s \in [N, [i]\rangle$ that enables both transitions concurrently, i.e., $s \geq s_x + s_y$.

As mentioned above, interleaving order does not enforce concurrent enabling of two transitions. In contrast, the concurrency relation requires concurrent enabling, for instance $A||_c B$ in Fig. 3(a), but not in Fig. 3(c). Nevertheless, there is the following dependency between both relations.

Lemma 1: For any free-choice system holds, every pair of transitions that is concurrent is also in interleaving order.

Proof: Let $(N, [i])$ be a free-choice system, $x, y \in T$, and $x||_c y$. From the latter, we know that there is a marking $s \in [N, [i]\rangle$ with $s \geq s_x + s_y$. Therefore, there are two possible firing sequences $(N, s)[xy\rangle$ and $(N, s)[yx\rangle$, which yields $x \succ y$ and $y \succ x$, in other words interleaving order. □

Further on, we can relate structural dependencies of transitions in sound free-choice systems to the behavioural relations. For $(N, [i])$ as a sound free-choice system, we say that two transitions $x, y \in T$ are *cyclic dependent*, if xF^+y and yF^+x. They are *structurally ordered*, if xF^+y and $y \not F^+ x$, and *structurally exclusive*, if $x \not F^+ y$ and $y \not F^+ x$.

Lemma 2: For any sound free-choice system holds, for every two transitions that are not concurrent, interleaving order and cyclic dependency coincide.

Proof: Let $(N, [i])$ be a sound free-choice system, $x, y \in T$, and $x \not\|_c y$.

- $\Rightarrow$ Let $x||y$ and assume $x \not F^+ y$. From $x||y$ we know $x \succ y$, which, in turn, implies the existence of a firing sequence containing x and y. Let $s_1, s_2 \in [N, [i]\rangle$ be the markings before and after firing of x, i.e., $(N, s_1)[x\rangle(N, s_2)$. Due to $x \not F^+ y$, all places $x^\bullet$ cannot impact on the enabling of y. Therefore, there must be a marking $s_3 \in [N, s_2\rangle$ enabling y for which holds $s_3 \geq s_y + \sum_{p \in x^\bullet}(s_p)$. In other words, s_3 is reachable from s_2, marks all places $x^\bullet$, and enables y. As a consequence, there must also be a marking s_4 from which s_3 is derived by firing of x, $(N, s_4)[x\rangle(N, s_3)$. Then, $s_4 \geq s_x + s_y$. That yields a contradiction, as we required $x \not\|_c y$. The assumption of $y \not F^+ x$ results in the same contradiction, due to the mirrored argument.
- $\Leftarrow$ From xF^+y and yF^+x we know that, due to the soundness and the free-choice property, there must be a firing sequence containing both transitions in either order (cf., Lemma 4.2 in [33]), $x \succ y$ and $y \succ x$. Thus, both transitions are in interleaving order.

□

Structural exclusiveness can also be related to the behavioural relations for sound free-choice systems.

Lemma 3: For any sound free-choice system holds, for every two transitions that are not concurrent, exclusiveness and structural exclusiveness coincide.

Proof: Let $(N, [i])$ be a sound free-choice system, $x, y \in T$, and $x \not\|_c y$.

- $\Rightarrow$ Let $x + y$ and assume xF^+y. From $x + y$ we know $x \not\succ y$. As the system is sound, x must not be a dead transition. Let $s_1, s_2 \in [N, [i]\rangle$ be the markings before and after firing of x in some firing sequence, $(N, s_1)[x\rangle(N, s_2)$. Again, xF^+y implies a firing sequence containing x and y due to the soundness and the free-choice property. In other words, there is a marking $s_3 \in [N, s_2\rangle$ with $s_3 \geq s_y$, i.e., y is enabled. That yields a contradiction with $x \not\succ y$. The same holds true for the mirrored argument under the assumption of yF^+x.
- $\Leftarrow$ Let $x \not F^+ y$ and assume $x \succ y$. Again, x must not be a dead transition (soundness property). Thus, it is

contained in some firing sequence and there are two markings $s_1, s_2 \in [N, [i]\rangle$ before and after firing of x. In order to meet $x \succ y$, still, there has to be a marking $s_3 \in [N, s_2\rangle$ that enables y. As above, $x \not{F}^{+} y$ implies that all places $x^\bullet$ cannot impact on the enabling of y. Thus, the marking s_3 might enable y while marking all places $x^\bullet$ as well, $s_3 \geq s_y + \sum_{p \in x^\bullet} (s_p)$. Consequently, there is a marking s_4 from which s_3 is derived via firing of x, $(N, s_4)[x\rangle(N, s_3)$. Then, $s_4 \geq s_x + s_y$ which is not in line with $x \not{||}_c y$. Again, the argument can be mirrored for the assumption of $y \succ x$. As both assumptions, $x \succ y$ and $y \succ x$ lead to contradictions, we know that $x + y$.

□

Finally, we are able to state that the behavioural relations can be derived from the concurrency relation and the flow relation.

Theorem 1: Given a sound free-choice system and its concurrency relation, the behavioural profile can be calculated from the transitive closure of the flow relation.

Proof: Interleaving order is given by concurrency and cyclic dependency according to Lemma 1 and Lemma 2. Exclusiveness can be derived from the flow relation as well (Lemma 3). It remains to be shown how strict order can be derived from the transitive closure of the flow relation. Let $(N, [i])$ be a sound free-choice system, $x, y \in T$, $xF^{+}y$, and $y \not{F}^{+} x$. From $xF^{+}y$, we know $x \succ y$ (cf., $\Leftarrow$ of the proof of Lemma 2). Now consider $y \not{F}^{+} x$. If we assume $y \succ x$, we get that both transitions are required to be concurrent, $y ||_c x$ (cf., $\Leftarrow$ of the proof of Lemma 3). As the concurrency relation is given, we can exclude these pairs of transitions, such that for all remaining pairs $y \not{F}^{+} x$ implies $y \not\succ x$. Finally, from $x \succ y$ and $y \not\succ x$ follows $x \rightsquigarrow y$. This means that strict order can also be derived from the transitive closure of F. □

Based thereon, the behavioural profile can be calculated very efficiently for sound free-choice systems.

Corollary 1: Given a sound free-choice WF-system, the behavioural profile can be derived in $O(n^3)$ with n as the number of transitions and places of the system.

Proof: For any free-choice system the concurrency relation can be calculated in $O(n^3)$ with n as the number of transitions and places of the system according to [34]. Further on, the transitive closure for the flow relation is well-known to be computable in $O(n^3)$ as well [35]. According to Theorem 1 this suffices to derive the behavioural profile. □

The requirements for the application of our approach can also be decided in polynomial time. The free-choice property can be decided solely based on the structure of the system, i.e., the flow relation. On the other hand, soundness can be traced back to liveness and boundedness, which can also be decided in polynomial time for free-choice WF-systems [32].

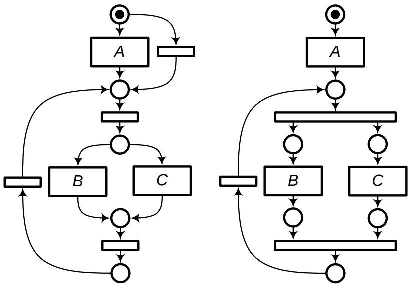


Fig. 5. Equal behavioural profiles, but not trace equivalent

4.4 Behavioural Profiles vs. Trace Equivalence

After we introduced behavioural profiles, we investigate their relation to the notion of trace equivalence. In particular, two aspects that impact on the set of possible traces are not captured in the behavioural profile.

First, the weak order relation and, therefore, all behavioural relations, is based on the *existence* of a firing sequence containing certain transitions. Whether or not a transition is obligated to occur at a certain point is, however, not captured. Of course that might result in different traces.

Consider, for instance, transition A in Fig. 5. This transition might be skipped in the system on the left-hand side, which yields traces that commence with the occurrence of B or C. These traces are not possible in the system on the right-hand side, in which all traces start with the occurrence of A.

The second aspect that is not captured by the behavioural profile, are transition cardinalities in traces. Although the behavioural profile specifies whether a transition might occur at most once or whether it might occur multiple times, dependencies between the occurrence cardinalities of two transitions are neglected. Again, this is illustrated by Fig. 5. In both systems, transitions B and C are part of a loop. However, in the system on the left-hand side, only one transition might occur per loop iteration, whereas the other system enforces the occurrence of both transitions per iteration. As a consequence, the latter allows solely for traces, in which at most two occurrences of B (or C, respectively) follow on each other directly.

Thus, the example in Fig. 5 illustrates that the equivalence of behavioural profiles does not imply trace equivalence. However, we observe the following.

Property 3: Every two WF-systems that are trace equivalent have the same behavioural profile.

This property follows directly from the definition of the behavioural relations. Two systems having the same set of traces will end up with the same weak order relation, as it requires the existence of a certain trace. That, in turn, results in equivalent relations of the behavioural profiles.

5 CONSISTENCY MEASURES FOR ALIGNED PROCESS MODELS

The previously defined concept of a behavioural profile allows us to formally discuss the notion of a degree of profile consistency between a pair of process models. Section 5.1 introduces the concept of a correspondence relation that defines an alignment between activities of two process models. We will then use the classical notion of trace equivalence, which we extend to trace consistency, as a benchmark. Accordingly, this section introduces two types of consistency notions between aligned process models. In Section 5.2, we show how the trace equivalence criterion can be applied in the alignment setting. Subsequently, Section 5.3 defines a consistency measure grounded on the notion of behavioural profiles. Finally, Section 5.4 elaborates on the interpretation of this degree.

5.1 The Notion of Correspondences

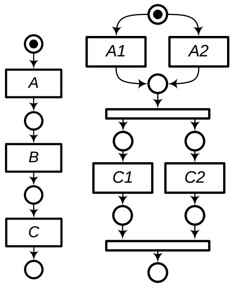
Before we define the consistency measures, first and foremost, we have to introduce the concept of an alignment according to our formal model. It is captured by a correspondence relation that specifies an alignment between transitions of two WF-systems. A transitions is said to be aligned, if it is related to some corresponding activity by a correspondence relation.

Definition 7 (Correspondence Relation): Let $(N_1, [i_1])$ and $(N_2, [i_2])$ be two WF-systems. The *correspondence relation* $\sim \subseteq T_1 \times T_2$ aligns both systems by relating corresponding transitions to each other, such that $\sim \neq \emptyset$.

Depending on the cardinality constraints that are inducted by the correspondence relation, we distinguish the following types (cf., alignment cardinalities from the field of ontology matching [36]). A correspondence relation $\sim$ between two WF-systems $S_1 = (N_1, [i_1])$ and $S_2 = (N_2, [i_2])$ is said to be

- *total* from S_1 to S_2, if $\forall\, t_1 \in T_1\ [\ \exists\, t_2 \in T_2\ [\ t_1 \sim t_2\ ]]$,
- *injective* from S_1 to S_2, if $\forall\ (t_a, t_x), (t_b, t_y) \in \sim$ holds $(t_x = t_y) \Rightarrow (t_a = t_b)$,
- *bijective*, if it is total from both S_1 and S_2, and injective from one of them.

For instance, assume that both WF-systems in Fig. 6 have been aligned as indicated by the transition labels, e.g., $A \sim A1$ and $A \sim A2$. In this case, the correspondence relation is injective from model 6(a) to model 6(b), but not from model 6(b) to model 6(a). In contrast, the relation is total from model 6(b) to model 6(a), but not vice versa.



(a) (b)

Fig. 6. Correspondence relation is not injective from 6(b) to 6(a)

The distinction between vertical and horizontal alignment as introduced in Section 2 is reflected in the cardinality properties of the correspondence relation. That is, horizontal alignment between different variations of a process results in a correspondence relation that is injective in both directions. Due to an equal abstraction level, there are solely 1:1 correspondences between activities. In contrast, vertical alignment imposes no restriction on the cardinality properties of the correspondence relation.

Any degree for alignment consistency measures the quality of correspondences between process models. Solely activities that are aligned by the correspondence relation are considered. Hence, we define the sets of aligned transitions for two aligned models as follows.

Definition 8 (Aligned Transitions): Let $(N_1, [i_1])$ and $(N_2, [i_2])$ be two WF-systems and $\sim$ a correspondence relation between them. The set of *aligned transitions* $T_1^{\sim} \subseteq T_1$ for $(N_1, [i_1])$ is defined as $T_1^{\sim} = \{t_1 \in T_1 \mid \exists\ t_2 \in T_2\ [\ t_1 \sim t_2\ ]\}$. The set $T_2^{\sim}$ for $(N_2, [i_2])$ is defined analogously.

With respect to our example in Fig. 6, the set of aligned transitions is given as $T_1^{\sim} = \{A, C\}$ for the model 6(a) due to transition B having no counterpart in the second model. For the model 6(b), all transitions are aligned, i.e., $T_2^{\sim} = \{A1, A2, C1, C2\}$.

5.2 Consistency based on Trace Equivalence

As a benchmark for our consistency analysis, we define a notion of consistency based on the trace equivalence criterion. First, we adapt the trace equivalence criterion for model alignments yielding the notion of trace consistency. Second, the degree of trace consistency is introduced based on the amount of traces of one model that have a counterpart in the other model. We already mentioned in Section 2 that the application of trace equivalence in an alignment setting requires that all parts that have been subject to projection are discarded. Therefore, we define how the projection implied by a certain correspondence relation is applied to a firing sequence of one of the aligned WF-systems. For two WF-systems $(N_1, [i_1])$ and $(N_2, [i_2])$ that are aligned by $\sim$ and a firing sequence $\sigma \subseteq T_1^*$ of $(N_1, [i_1])$ we define a short-hand notation for all aligned transitions that are part of σ up to index j, such that $T_{\sigma|j}^{\sim} = \{t_x \in \sigma \mid x < j \wedge t_x \in T_1^{\sim}\}$.

Definition 9 (Projected Firing Sequence): Let $(N_1, [i_1])$ and $(N_2, [i_2])$ be two WF-systems aligned by $\sim$ and $\sigma \subseteq T_1^*$ a firing sequence of length n. The *projected firing sequence* $\sigma^{\sim}$ for σ contains all aligned transitions, i.e., $\sigma^{\sim} = \bigcup_{i=0}^{|T_{\sigma|n}^{\sim}|}(i, t_i)$ with $t_i \in T_1^{\sim}$, such that $\exists\ j \in \mathbb{N}\ [\ (j, t_i) \in \sigma \wedge i = |T_{\sigma|j}^{\sim}|\ ]$.

Again, we illustrate the notion of projected firing sequences by the examples in Fig. 6. For instance, the model 6(a) defines a reachable firing sequence $\sigma = \{(0, A), (1, B), (2, C)\}$. Transition B is not aligned, hence, the respective projected firing sequence is defined as $\sigma^{\sim} = \{(0, A), (1, C)\}$. For any firing sequence that does not contain any aligned transition, the projected firing sequence is an empty sequence.

Based thereon, the trace equivalence criterion can be applied in an alignment setting. However, a reasonable application is possible only in case of horizontal alignment, which requires the correspondence relation to be injective in both directions. That is due to the fact that for 1:n (or even n:m) correspondences behavioural semantics of the aligned group of activities are subject to interpretation. That is illustrated by the alignment of the aforementioned models in Fig. 6. Obviously, occurrence of A in the model in Fig. 6(a) corresponds to the occurrence of *either* $A1$ or $A2$ in the model in Fig. 6(b), whereas the occurrence of C corresponds to the occurrence of *both*, $C1$ and $C2$. While this example illustrates solely the conjunction and disjunction of groups of corresponding activities, there might be more complex dependencies in the general case. That, in turn, raises the question, how these dependencies can be extracted in an efficient way. Therefore, we neglect these kinds of correspondences and focus on consistency of horizontal alignments.

A trace consistent alignment requires all reachable projected firing sequences of one model to have a corresponding sequence in the other model. Thus, two firing sequences σ_1, σ_2 of two WF-systems $(N_1, [i_1])$ and $(N_2, [i_2])$ that are aligned by $\sim$ are corresponding under $\sim$, denoted by $\sigma_1^{\sim} \simeq \sigma_2^{\sim}$, iff $\sigma_1^{\sim} = \sigma_2^{\sim} = \emptyset$ or $\forall\, t_i \in \sigma_1^{\sim}\ [\ \exists\, t_j \in \sigma_2^{\sim}\ [\ i = j \wedge t_i \sim t_j\ ]]$.

Definition 10 (Trace Consistency of Alignment): Let $(N_1, [i_1])$ and $(N_2, [i_2])$ be two WF-systems with initial places i_1 and i_2, and final places o_1 and o_2, respectively. Let they be aligned by $\sim$. The alignment specified by $\sim$ is *trace consistent*, iff for every firing sequence σ_1 with $(N_1, [i_1])[\sigma_1\rangle(N_1, [o_1])$ and σ_2 with $(N_2, [i_2])[\sigma_2\rangle(N_2, [o_2])$ there is a firing sequence σ_x with $(N_2, [i_2])[\sigma_x\rangle(N_2, [o_2])$ and σ_y with $(N_1, [i_1])[\sigma_y\rangle(N_1, [o_1])$, such that $\sigma_1^{\sim} \simeq \sigma_x^{\sim}$ and $\sigma_2^{\sim} \simeq \sigma_y^{\sim}$.

According to Definition 10 the trace consistent alignment is based on firing sequences leading from the initial state to the final state, instead of considering all reachable firing sequences. That is due to the fact the projection might result in empty sequences. Of course, preservation of such an empty sequence is reasonable solely in the case the final state has been reached by the sequence. That is illustrated in Fig. 7, which depicts three models that are aligned according to their transition labels. In model 7(a), there is a reachable firing sequence containing just the transition A. Obviously, projection would result in an empty sequence for this firing sequence. Such an empty sequence does not exist for model 7(b). However, as the firing sequence containing only the transition A does not lead to the final state it is not required to be preserved. Thus, models 7(a) and 7(b) show a trace consistent alignment. In contrast, model 7(c) is not consistent with the other two models as it allows for a firing sequence from the initial to the final state that does not contain any aligned transition. For this sequence, there is no counterpart in the other two models.

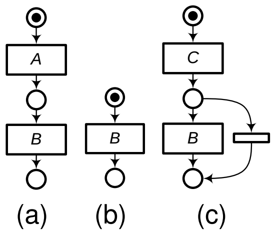


Fig. 7. Projection might result in empty sequences

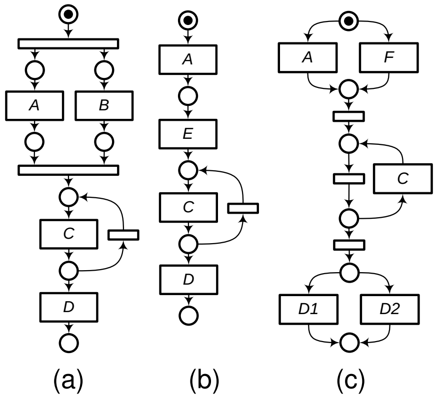


Fig. 8. Exemplary alignment setting

We further illustrate the application of trace consistency in Fig. 8. Concerning models 8(a) and 8(b) the projected firing sequences of both models are equal. Thus, both models show a trace consistent alignment. For the model pairings 8(a) and 8(c), as well as 8(b) and 8(c) the criterion is not applicable, as the correspondence relation is not injective in either direction ($D \sim D1$ and $D \sim D2$). Even if transition D would be ignored, the alignment would not be trace consistent, as model 8(c) allows for a projected firing sequence starting with transition C, whereas skipping of the aligned transition A is not possible in the other two models.

With respect to our initial example in Fig. 1, only the horizontal alignment between the model (A1) and (A2) can be evaluated. The vertical alignment between model (B) and the upper models (A1) and (A2) does not meet the requirements of a correspondence relation that is injective in both directions. In order to apply our definitions, we first have to map the BPMN models into our formal model. We do not go into the details of such a mapping, but refer the reader to [22] for more information. The processes (A1) and (A2) depicted in Fig. 1 can be represented as free-choice WF-systems, as we know all start and end events to be exclusive. Deciding trace consistency for the alignment between the model (A1) and (A2) leads to a negative result. Obviously, in model (A1) activity *Analyse Competitor* might happen after *Submit Quote*, which is not possible in model (A2).

The notion of trace consistency according to Definition 10 shows how the trace equivalence criterion can be applied to judge about consistency of aligned process models. In the same vein, the ratio of traces of one model that can be mirrored in the second model and all firing sequences can be used to quantify the consistency in a straight-forward manner under the assumption of finite sets of traces, cf., [37]. In the following definition, we assume the WF-systems to be bounded and acyclic, which guarantees a finite set of traces.

Definition 11 (Degree of Trace Consistency): Let $(N_1, [i_1])$ and $(N_2, [i_2])$ be two bounded acyclic WF-systems with initial places i_1 and i_2, and final places o_1 and o_2, respectively. Let they be aligned by $\sim$. Let $\mathcal{T}_1^{\sim}$ and $\mathcal{T}_2^{\sim}$ be the sets of projected firing sequences, such that $\sigma_1^{\sim} \in \mathcal{T}_1^{\sim}$ and $\sigma_2^{\sim} \in \mathcal{T}_2^{\sim}$, iff there are firing sequences σ_1 and σ_2 with $(N_1, [i_1])[\sigma_1\rangle(N_1, [o_1])$ and $(N_2, [i_2])[\sigma_2\rangle(N_2, [o_2])$, respectively. With $\mathcal{T}^{\sim} = (\mathcal{T}_1^{\sim} \cup \mathcal{T}_2^{\sim})$, the *degree of trace consistency* induced by $\sim$ is defined as

$$\mathcal{TC}^{\sim} = \frac{|\{\sigma_x^{\sim} \in \mathcal{T}^{\sim} \mid \exists\ \sigma_y^{\sim} \in \mathcal{T}^{\sim}\ [\ \sigma_x^{\sim} \simeq \sigma_y^{\sim}]\}|}{|\mathcal{T}_1^{\sim}| + |\mathcal{T}_2^{\sim}|}.$$

For the example models given in Fig. 7, we see that the trace consistency of models 7(a) and 7(b) implies a degree of trace consistency of $\mathcal{TC}^{\sim} = 1.0$. For models 7(a) and 7(c), in turn, the degree equals $\mathcal{TC}^{\sim} = \frac{2}{1+2} \approx 0.667$ owing to one complete trace of model 7(c) that cannot be mirrored in model 7(a). The same degree value is observed for the model pair 7(b) and 7(c).

For the models in Fig. 8, the degree of trace consistency cannot be computed. All these models have infinite sets of traces. The two models (A1) and (A2) of our initial example in Fig. 1 cannot be assessed for the same reason. When considering solely the first part of both models (until reaching the branching point on whether the offer has been successful), we see that halve of the traces of both models can be mirrored by the other model, yielding a degree of trace consistency of $\mathcal{TC}^{\sim} = \frac{2}{2+2} = 0.5$.

5.3 Consistency based on Behavioural Profiles

In general, our notion of consistency based on behavioural profiles, i.e., profile consistency, is grounded on the preservation of behavioural relations for corresponding activities. In contrast to the notion of a trace consistent alignment, it does not require the correspondence relation to be injective. Instead, it allows for 1:n (and even n:m) correspondences. Therefore, this notion can be applied to vertical as well as horizontal alignments. In addition, we are able to measure the degree of profile consistency.

In order to specify the degree of behaviour that is preserved by an alignment, we define two sets of consistently aligned transition pairs, one for each of the aligned WF-systems. These sets contain all pairs of transitions that are aligned by a correspondence relation, such that their relation of the behavioural profile is preserved. With $\mathcal{R}_1 \in \{\rightsquigarrow, \rightsquigarrow^{-1}, +, ||\}$ and $\mathcal{R}_2 \in \{\rightsquigarrow, \rightsquigarrow^{-1}, +, ||\}$ as the respective behavioural relations for two WF-systems $(N_1, [i_1])$ and $(N_2, [i_2])$, we define the sets of consistent transition pairs as follows.

Definition 12 (Consistent Transition Pairs): Let $(N_1, [i_1])$ and $(N_2, [i_2])$ be two WF-systems aligned by $\sim$. The set of *consistent transition pairs* $CT_1^{\sim} \subseteq (T_1^{\sim} \times T_1^{\sim})$ for $(N_1, [i_1])$ contains all transition pairs (t_x, t_y), such that

- if $t_x = t_y$, then $\forall\ t_s \in T_2^{\sim}$ with $t_x \sim t_s$ it holds $t_x \mathcal{R}_1 t_x \Rightarrow t_s \mathcal{R}_2 t_s$
- if $t_x \neq t_y$, then $\forall\ t_s, t_t \in T_2^{\sim}$ with $t_s \neq t_t$, $t_x \sim t_s$, and $t_y \sim t_t$ it holds either
  (1.) $t_x \mathcal{R}_1 t_y \Rightarrow t_s \mathcal{R}_2 t_t$ or
  (2.) $t_x \sim t_t$ and $t_y \sim t_s$.

The set $CT_2^{\sim}$ for $(N_2, [i_2])$ is defined analogously.

Preservation of the behavioural relation is only required in case there are no overlapping correspondences. Imagine that two transitions t_x and t_y in one model are associated with sets of transitions in the other models, e.g., $t_x \sim t_s$, $t_x \sim t_t$, $t_y \sim t_s$, and $t_y \sim t_t$. This case might be interpreted as an n:m correspondence between two sets of transitions. Imagine that t_x and t_y would be in strict order. That, in turn, would require $t_s \rightsquigarrow t_t$ and $t_t \rightsquigarrow t_s$, which cannot be satisfied. Hence, it is reasonable to neglect the behavioural relation in such cases (cf., the second condition in Definition 12).

With respect to the examples in Fig. 8 it is easy to see that all pairs of aligned transitions are also consistent with respect to their behavioural relation. For instance, the strict order relation between transitions A and D in model 8(a) is preserved for transition pair A and $D1$, as well as A and $D2$ in model 8(c). In addition, in all three models it holds $C||C$. That is, C might occur multiple times during execution.

Similar to the degree of trace consistency defined in the previous section, the ratio between the transition pairs that are aligned in a consistent manner and all aligned pairs can be used as a consistency measure.

Definition 13 (Degree of Profile Consistency of Alignment): Let $(N_1, [i_1])$ and $(N_2, [i_2])$ be two WF-systems aligned by $\sim$. The *degree of profile consistency* of $\sim$ is defined as

$$\mathcal{PC}^{\sim} = \frac{|CT_1^{\sim}| + |CT_2^{\sim}|}{|(T_1^{\sim} \times T_1^{\sim})| + |(T_2^{\sim} \times T_2^{\sim})|}.$$

The degree of profile consistency considers all relations of the behavioural profile to be equally important. While we consider this to be appropriate in our context, these relations might be weighted in a different setting. For instance, for the use case of checking compliance of process models and process logs based on behavioural profiles, the exclusiveness relation might be weighted higher than the interleaving order relation. The former completely disallows the joint occurrence of two transitions, whereas the latter does not even enforce any order of occurrence.

Using this definition we are now able to quantify the degree of profile consistency for the examples. As mentioned before, all aligned transition pairs for the models in Fig. 8 are consistent. This yields a degree of profile consistency of $\mathcal{PC}^{\sim} = 1.0$ for all model pairings.

We mentioned above that the processes (A1) and (A2) depicted in Fig. 1 can be represented as free-choice WF-systems. The same holds true for process (B), for which the deferred choice is modelled as non-determinism. This allows for efficient derivation of the behavioural profile.

Each of the upper two models (A1) and (A2) consists of 7 aligned activities, i.e., $|(T^{\sim} \times T^{\sim})| = 49$ for both models, whereas model (B) contains even 10 aligned activities, which yields $|(T^{\sim} \times T^{\sim})| = 100$. An analysis of models (A1) and (A2) reveals that for four pairs of activities the relations of the behavioural profile are not consistent. That is, the activity pairs *(Analyse Competitors - Contact Customer)* and *(Analyse Competitors - Submit Quote)* along with the reversed pairs have different behavioural relations in both models. Therefore, the degree of profile consistency for the horizontal alignment is $\mathcal{PC}^{\sim} = \frac{45+45}{49+49} \approx 0.918$. For the vertical alignment between models (A1) and (B), we see that in model (A1) the same four activity pairs are inconsistent. Out of the 100 aligned activity pairs in model (B), 6 pairs are inconsistent with respect to the model (A1), e.g., *(Get Counter Offer History - Send Quote)*. That, in turn, yields a degree of profile consistency of $\mathcal{PC}^{\sim} = \frac{45+94}{49+100} \approx 0.933$ for the models (A1) and (B). Regarding the alignment of models (A2) and (B) the degree of profile consistency is $\mathcal{PC}^{\sim} = \frac{49+100}{49+100} = 1.0$ as the behavioural relations are preserved for all aligned pairs of transitions. We conclude that the alignment is completely consistent solely for the models (A2) and (B). However, the degree of profile consistency for the other alignments indicates that there are only minor deviations for these cases.

5.4 Interpretation of Profile Consistency

As exemplified in the previous section, the degree of profile consistency ranges between 0 and 1.0 for two process models and a correspondence relation. Still, a degree of 1.0 does not imply that both models are (projected) trace equivalent. This stems from the fact that the underlying behavioural profile represents a behavioural abstraction, cf., Section 4.4. Following on the argumentation given in Section 2, we argue that the projection of activities should not impact on the consistency assessment. As a consequence, projections might result in two models having a degree of profile consistency of 1.0, even though they are not trace consistent.

Apparently, the degree of profile consistency quantifies the quality of an alignment with respect to the order of potential activity occurrences. A degree of 1.0 guarantees that all these constraints are equal for the aligned activities of two models. A degree of 0.9, in turn, indicates that the constraints on the order of potential activity occurrences are equal solely for 90% of the relations between aligned activities. As the degree of profile consistency measures the quality of the alignment, its definition is independent of the coverage of the process models by the correspondence relation (i.e., the share of activities in both models that are aligned). Based on the degree of profile consistency, consistency thresholds might be defined. However, we assume these thresholds to be highly dependent on a specific project setting.

Once a degree of profile consistency below 1.0 is observed, the question of how to locate the source of inconsistency has to be addressed. According to our approach, inconsistencies manifest themselves in different relations of the behavioural profile of two process models for a pair of aligned activities. This information can directly be provided to business analysts and system analysts in order to judge on the necessity of the inconsistency. While this kind of feedback allows for locating the inconsistency directly in case of only a few inconsistent profile relations (e.g., caused by an interchanged order of two activities in a sequence), it might be inappropriate if a big number of profile relations is inconsistent. Imagine two process models containing a set of aligned activities in sequential order and assume that one of these activities in one model would now be moved to a branch that is executed concurrently to the remaining activities. Then, all behavioural relations between this activity and the remaining activities would be inconsistent, such that feedback on the set of activities that show inconsistent relations would be of little help. Instead, we would consider the biggest subset of aligned activities that show consistent behavioural relations among each other to be valuable feedback on the observed inconsistencies. For the aforementioned case, the single activity having inconsistent relations with all other activities might be identified by this approach.

6 CASE STUDY: CONSISTENCY OF THE SAP REFERENCE MODEL

In order to evaluate the appropriateness of our consistency measure based on behavioural profiles, this section presents a case study based on SAP reference model (refer to [38] for further information). This reference model describes the functionality of the SAP R/3 system in its version 4.6 and comprises 604 process diagrams modelled as EPCs. These diagrams are expanded to 737 EPC models as some diagrams contain multiple disconnected EPCs. The EPC models capture different functional aspects of an enterprise, such as sales or accounting. However, the models are not fully orthogonal. That is, various models show an overlap, such that events and functions with identical labels occur in multiple models. A consistency analysis can help to determine those model pairs that can be easily integrated. It is well established that such redundancies can lead to similar anomality problems as known from database research [39]. Models with a high degree of consistency might therefore be good candidates for integration in order to resolve redundancies in the model collection. Partially overlapping process variants can be aligned vertically according to their labels. Based thereon, we are able to apply the aforementioned consistency measures for pairs of models. Naturally, these models should show a high alignment consistency as the different processes describe the same underlying IT-functionality. In addition, the models have been created by a rather small group of process modellers with a similar professional background and within the same organisational unit of a single company. That, in turn, eliminates various sources of inconsistencies that might be observed in different contexts.

The reference model consists of EPCs for which our measures are not directly applicable. Therefore, Section 6.1 summarises our preprocessing of the reference model, which deals with the mapping to free-choice WF-systems

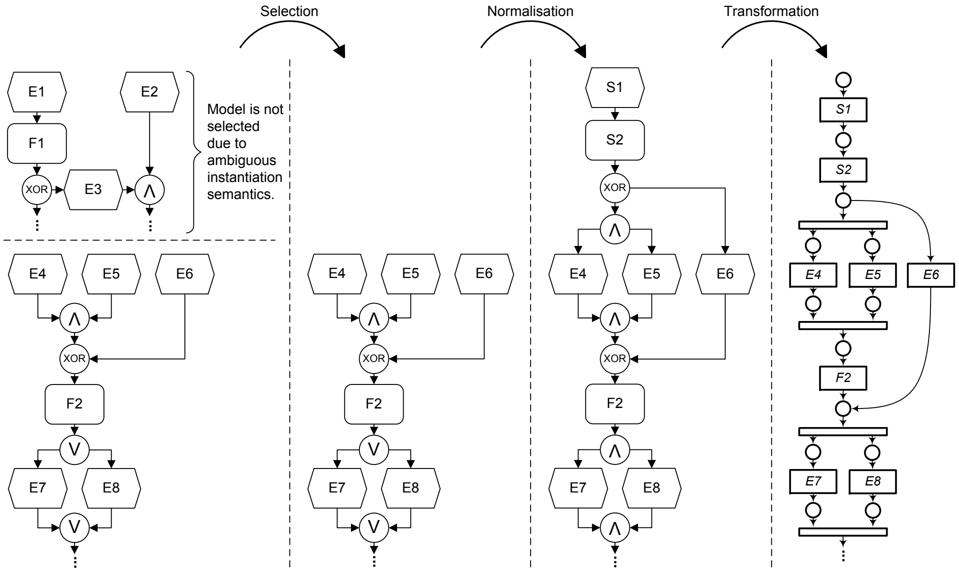


Fig. 9. Preprocessing of EPCs incorporates model selection, model normalisation, and model transformation

and exclusion of erroneous models. Subsequently, Section 6.2 introduces the results of our consistency analysis and Section 6.3 discusses the results.

6.1 Preprocessing of the Reference Model

The preprocessing of the SAP reference model for our purposes incorporates three steps: model selection to exclude incorrect models, model normalisation to match WF-system topology, and model transformation to WF-systems. These steps are illustrated in Fig. 9, which depicts the preprocessing for two exemplary EPCs. In general, EPCs consist of events (represented as hexagons) specifying the process state, functions (represented as rounded rectangles) modelling activities, and connectors (represented as circles) that are used to implement the control flow logic. That is, they are used to specify convergence or divergence of control flow.

Model selection. It is well-known that the SAP reference model contains models that are erroneous [40], [41]. These models contain deadlocks or livelocks, or even syntactical errors that preclude any reasonable interpretation. An example for the latter is an EPC with events or functions with more than one incoming or outgoing flow arc. As stated above, convergence and divergence of control flow has to be realised by connectors in EPCs. Hence, semantics for multiple arcs leading to or starting at a function or an event is not defined. However, we ignore syntax errors that do not impact on the behavioural semantics (e.g., a connector with one incoming and one outgoing flow arc). In addition, various models have ambiguous instantiation semantics [42]. That is, these models have multiple start events, but do not contain a so called *start join*. Such a start join is *'a join connector such that for every other node n in the EPC there is either a path from n to the start join or a path from it to n.'* [42]. An EPC without start join is illustrated in Fig. 9. The EPC model (taken from [42]) in the upper left corner is not selected as the question whether the event $E2$ is needed for instantiation cannot be answered at design time. Instead, it depends on the state of the process instance that, in turn, impacts on the branching at the XOR connector.

From the initial set of 737 EPCs, we removed models for the following reasons.

- Triviality. 23 models consist of solely one node and can, therefore, not be used in behavioural consistency analysis.
- Syntax errors. 4 models have syntax errors that preclude any reasonable interpretation.
- Semantical errors. 84 models contain deadlocks or livelocks.
- Ambiguous instantiation semantics. 169 models do not have a start join.

Some models meet more than one of these criteria. Consequently, our selection yields a sample of 511 models for the consistency analysis.

Model normalisation. The first normalisation step aims at deriving a single start event for all EPCs that contain multiple start events. As mentioned above, we selected solely models that contain a start join as they have unambiguous instantiation semantics. For these models, a single start event might be introduced by mirroring the control flow logic between the start events and the minimal start join. A start join is minimal, if for all paths from the join to an end event holds that the path does not contain another start join. Again, the normalisation of start events is illustrated in Fig. 9. Two auxiliary nodes $S1$ and $S2$ are introduced, such that the event $S1$ is now the unique start event of the process. The same approach is taken for multiple end events, such that all models have a dedicated start event and a dedicated end event after the first normalization step. For obvious reasons, all nodes introduced as part of the normalisation (e.g., $S1$ and $S2$ in Fig. 9) are neglected when we match equal labels in order to establish correspondences between different models.

In a second normalisation step, OR connectors in the EPCs are handled. In contrast to the first step, handling of OR connectors is applied solely for the analysis of profile consistency. The relations of the behavioural profile are all based on the weak order relation. This elementary relation captures the fact whether or not there is a trace containing two activities in a certain order. Therefore, the difference between a parallel and an inclusive OR splitting of control flow does not impact on the behavioural relations. As a consequence, we can replace all splitting OR connectors in the EPCs with splitting AND connectors. Further on, we also replaced the corresponding merging OR connectors to merging AND connectors for all block-structured regions. That is, whenever all incoming paths of a merging OR connector originate from a single splitting OR connector, both are replaced with AND connectors. Fortunately, all selected models contain only block-structured OR connectors. Thus, none of the models consists of an OR connector after the second normalisation step. Note again that the second normalisation step is not applied if we check an alignment for trace consistency, as replacing OR by AND connectors obviously impacts on the set of reachable firing sequences.

Model transformation. The transformation of EPCs to Petri nets follows on common EPC formalisations (cf., [41]) with a slightly different treatment of events. As events are also considered in the consistency analysis, we need to preserve their labels in the transformation. Therefore, an event is mapped to a transition and two places instead of a single place. Functions, AND, and XOR connectors are mapped as usual. Fig. 9 depicts the transformation for the running example, which shows that additional transitions and places might be introduced in order to derive a bipartite Petri net structure. For the transformation, we assume all EPCs to be free of OR connectors. As mentioned before, this holds for all normalised EPCs at least in the analysis of profile consistency. EPCs that contain OR connectors and should be analysed with respect to their trace consistency are not transformed, but have to be handled manually.

All Petri nets created by the transformation are *workflow* nets owing to the normalisation of start and end events of the respective EPC. Moreover, as we transform solely EPCs without OR connectors, the resulting nets are also *free-choice*. For these nets, we derive the corresponding WF-system by introducing an initial marking that marks the initial place of the WF-net. Further on, the systems are *sound*, as we removed models with behavioural anomalies already in the model selection phase.

6.2 Consistency Results

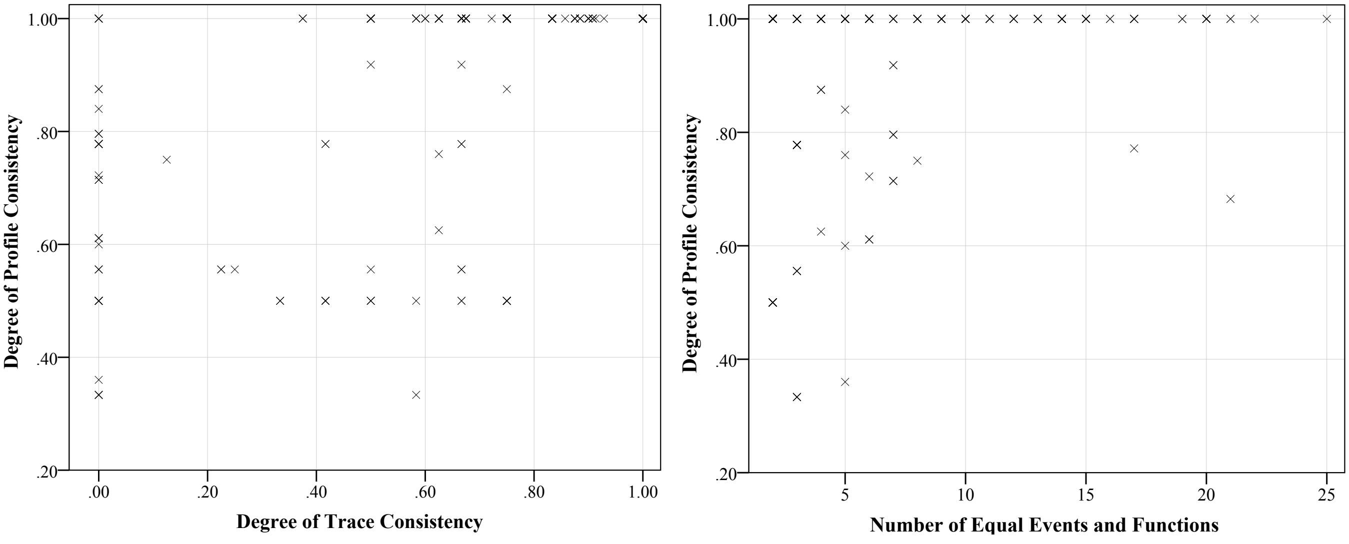
After preprocessing of the SAP reference model, we are able to analyse its consistency. As mentioned before, we establish correspondences between events and functions with equal labels. Further on, we extract all pairs of process models that are aligned by at least two correspondences. For such a pair, we then calculate the consistency measures as introduced in Section 5, that is, trace consistency, the degree of trace consistency, and the degree of profile consistency of the alignment.

In order to derive these measures we implemented the calculation of behavioural profiles for sound free-choice WF-systems as introduced in Section 4.3 in a research prototype. As mentioned before, the preprocessing results in sound free-choice WF-systems for all EPC models. Thus, the behavioural profile for a single model and the profile consistency for a pair of models is calculated within seconds. Unfortunately, the calculation of trace consistency cannot be done efficiently for all models. We implemented the derivation of all projected firing sequences for a pair of sound free-choice WF-systems, such that all transitions that are not aligned are marked to be neglected. Afterwards, we reduce the size of the systems by a *fusion of series places* [43] that are connected by such marked transitions. Based thereon, the state space of the respective system is explored. Although suffering from the state explosion problem, this approach works for the majority of models that can be transformed into sound free-choice WF-systems. For 18 model pairs, we have to rely on manual checking of trace consistency due to the size of the state space. However, all EPC models containing OR-connectors cannot be transformed into free-choice WF-systems. Therefore, we check trace consistency manually for these models. For the degree of trace consistency, a manual computation is not feasible. Hence, this degree is computed solely for pairs of models that do not show OR-connectors and for which the state space can be explored completely.

Table 1 summarises the results of our consistency analysis. We distinguish three different categories of alignments. We either align only functions, only events, or both node types. Of course, the node type is considered in the latter category as well, i.e., a correspondence holds either between two functions or two events, but not between a function and an event. The second column states the number of pairs of models with at least two correspondences. We see that we align significantly more models if events are taken into account. That is due to the structure of the reference model, in which single models contain a lot more events than functions. In addition, end events of one model might reappear as start events of another model. Independent of the category, the average profile consistency (Av. $\mathcal{PC}^{\sim}$) is rather high, which is further underpinned by the low numbers of model pairs with a profile consistency less than 1.0 ($\mathcal{PC}^{\sim} < 1.0$). In contrast, the amount of model pairs that are not trace consistent is significantly larger. When events are considered more than a half of the aligned models is not trace consistent, which yields a set of nearly 500 models in absolute terms. To put it differently, while trace equivalence suggests high inconsistencies,

TABLE 1
Consistency measures for the identified pairs of aligned models

Alignment	Pairs	Av. $\mathcal{PC}^{\sim}$	$\mathcal{PC}^{\sim} < 1.0$	Not Trace Consistent	Pairs for $\mathcal{TC}^{\sim}$	Av. $\mathcal{TC}^{\sim}$
Functions	171	0.98	7 (4.09%)	56 (32.75%)	114 (66.67%)	0.89
Events	907	0.96	74 (8.16%)	446 (49.17%)	695 (76.63%)	0.83
Functions & Events	952	0.96	86 (9.03%)	484 (50.84%)	735 (77.21%)	0.81



(a) (b)

Fig. 10. Analysis of profile consistency values

the degree of profile consistency points to very high consistency between the models. Further on, for a certain subset of model pairs (Pairs for $\mathcal{TC}^{\sim}$) that do not contain OR-connectors and for which state space exploration is possible, we also computed the average degree of trace consistency (Av. $\mathcal{TC}^{\sim}$). We see that this degree is significantly lower than the average of the degree of profile consistency for all three categories.

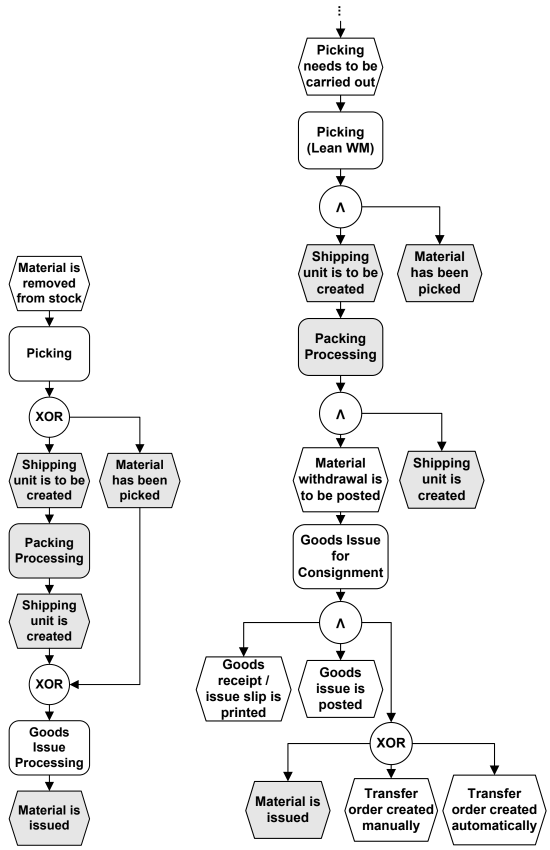
The spectrum of values for profile consistency is further explored in Fig. 10. Fig. 10(a) depicts the distribution of profile consistency values against the degree of trace consistency. Obviously, any trace consistent alignment (the degree of trace consistency equals 1.0) is also profile consistent. For the pairs of models that are not aligned in a trace consistent manner (the degree of trace consistency is lower than 1.0), we see that the spectrum of profile consistency is rather wide with numerous different consistency values. Thus, profile consistency is a fine granular measure of the alignment quality. In particular, model pairs for which the degree of trace consistency equals 0 (no trace can be mirrored) are classified in a more fine granular manner by the degree of profile consistency.

Moreover, the relation between profile consistency of a model pair and the number of aligned elements is illustrated in Fig. 10(b). We see that low values of profile consistency appear much more in settings with only a few aligned elements. We consider this result to be intuitive as a single inconsistency has a bigger impact on the profile consistency for models that are aligned by only a few correspondences than for models with a big overlap.

6.3 Discussion of the Results

In the previous section, we showed that an evaluation of consistency based on behavioural profiles yields a fine-granular measure. On the one hand, this measure goes beyond a simple search for corresponding elements as the behavioural characteristics are taken into account. On the other hand, the results in the previous section highlighted a significant difference in the share of model pairs that have a degree of profile consistency of 1.0 and those that show a trace consistent alignment. This section further underpins the appropriateness of our consistency measure by showing two alignment examples from the reference model in detail.

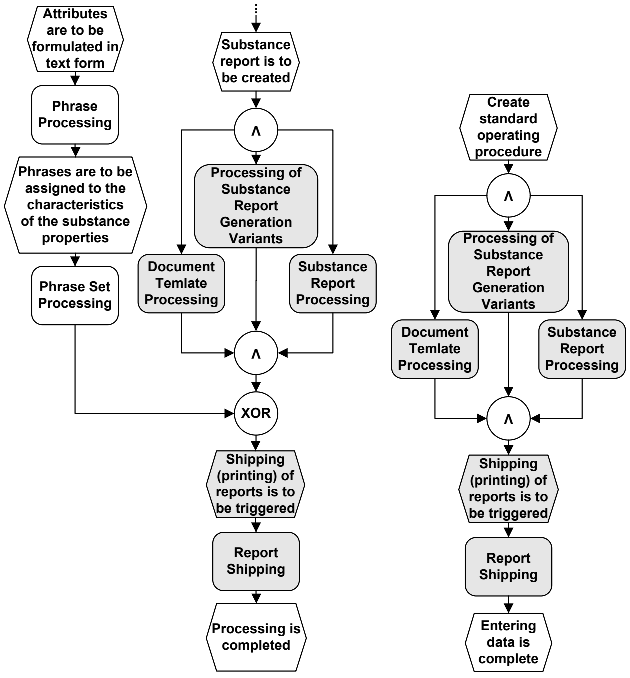
The first example illustrates the need to consider behavioural information in the alignment by means of two processes from the SAP reference model. Both processes, depicted in Fig. 11, specify a shipping procedure. While the model 11(a) is part of the procurement handling, the model 11(b) originates from the sales and distribution processing. Both models show a notable overlap, i.e., they specify how material shipping is implemented. That, in turn, is reflected by five elements with identical labels. Further elements of both processes have rather



(a) (b)

Fig. 11. EPCs from the SAP reference model that are overlapping, but not profile consistent

similar labels (e.g., *picking* and *picking (lean WM)*), which also indicates high similarity of process semantics, even though these nodes are not considered in our consistency analysis. However, an analysis of execution semantics of both models reveals various serious differences. For instance, if the process has reached the state *material has been picked*, the model 11(a) specifies that the state *material is issued* will be assumed directly. In contrast, the model 11(b) states that the state *material has been picked* implies the need for *packing processing*. What is an exclusive choice of states in the one model, corresponds to a conjunction of states in the other model. Although the *picking* functions are not aligned, that has to be regarded as a clear inconsistency. Therefore, a reasonable alignment of both models requires more than simply deriving the correspondences between equivalent model elements. Instead, the contradicting execution dependencies in both models have to be detected by a behavioural analysis. Such inconsistencies are reflected in our notion of profile consistency as well as in the notion of trace consistency. That is, application of the profile consistency measure to this example yields a value of $\mathcal{PC}^{\sim} = 0.6$, while the trace consistency criterion is also not fulfilled and the degree of profile consistency is $\mathcal{TC}^{\sim} = 0$.



(a) (b)

Fig. 12. EPCs from the SAP reference model that are profile consistent, but not trace consistent

The second example focusses on the impact of refinements of process variants on the consistency measures. Fig. 12 depicts two processes from the environment, health, and safety category of the reference model. Both describe the creation of incident reports and show several functions with identical labels. While the model 12(b) specifies the standard procedure of creating an incident report, the model 12(a) defines a specialisation of the procedure for the case of substance reports. Consequently, an alternative way of creating the report is defined. However, this difference does not preserve the trace consistency criterion. For instance, function *report shipping* is *always* preceded by function *document template processing* in model 12(b). This is not the case for model 12(a). Thus, the trace consistency criterion is not met and both models would be considered to be inconsistent. The degree of trace consistency for this example is $\mathcal{TC}^{\sim} = 0.93$. Although this degree indicates solely minor deviations, the specialisation of the incident report procedure (a model projection from model 12(a) to model 12(a)) impacts on the consistency result in a negative manner. In contrast, it is easy to see that both models have identical behavioural profiles for the aligned activities, i.e., there is no difference in the potential execution order. For instance, if function *document template processing* and function *report shipping* are part of the process execution, the former will always happen before the latter. Even though function *document template processing* is optional for execution in model 12(a), the strict order between the two functions holds in both models, i.e., function *report shipping* will never happen before function *document* *template processing*. As all behavioural relations for aligned elements are preserved, application of profile consistency yields $\mathcal{PC}^{\sim} = 1.0$. Thus, the refinement is considered to be fully consistent.

We conclude that the first example in Fig. 11 illustrates that an analysis of the pure co-occurrence of corresponding elements is not sufficient. Instead, behavioural inconsistencies have to be detected by means of behavioural equivalence notions. However, the second example showed that the projected variant of the trace equivalence criterion is too restrictive. Models that are overlapping only partially owing to the refinements and extensions, show an alignment that is not trace consistent. The degree of trace consistency is lowered by model projections. Instead, our consistency measure grounded on behavioural profiles enables detection and quantification of behavioural inconsistencies of aligned activities, while it allows for refinements and extensions that are common in practise.

7 RELATED WORK

In this section, we discuss four related areas of research, namely correspondences between process models, behavioural equivalence and inheritance, process similarity, and process variability.

For our approach we assume that *correspondences* of two process models have been identified and captured. The research area of schema integration, and in particular, schema matching investigates how such correspondences can be identified automatically. Overviews of the various techniques are provided in [36], [44], [45]. As discussed in the introduction, our notion of consistency can be related to desirable properties of schema mappings. Such properties are reported in [13]. Recently, several publications showed how these matching techniques can be applied for business process models [46], [47], [48], [49], [50]. For instance, [47] uses graph matching techniques in order to identify matching parts of related process models. Our approach is complementary to these works, as we show how consistency can be calculated once a suitable matching is available. For our experiments, we consider a simple matching based on string equivalence of labels, which can be easily replaced by more advanced matching techniques, such as those mentioned above.

The consistency of an alignment between process models closely relates to different notions of *behavioural equivalence*, such as trace equivalence and bisimulation. These notions yield a true or false answer and can, therefore, not directly be applied if models overlap partially. In particular, these notions require 1:1 correspondences, as semantics for 1:n or n:m correspondences are subject to interpretation. Good overviews of various equivalence notions are presented in [15], [51]. We illustrated the application of the trace equivalence criterion in the context of model projection. In the same vein, other equivalence criteria reported in [15], [51] might also be adapted. Nevertheless, the drawbacks of these boolean notions and measures that are based on them in the context of alignment consistency remain the same. Inspired by the notions of behavioural equivalence, *behaviour inheritance* aims at applying the idea of inheritance known from static structures to behavioural descriptions. Harel and Kupferman argued that object-oriented system design should incorporate a concept of behavioural inheritance for classes [52]. Based on the notion of object systems, they advocate that any system refinement should either preserve trace inclusion or simulation for the language build by the system's protocol. The idea to preserve the protocol of a behavioural model is also one of the basic inheritance notions by Basten et al. [53]. They define *protocol inheritance* and *projection inheritance* based on labelled transition systems and branching bisimulation. A model inherits the behaviour of a parent model, if it shows the same external behaviour when all actions that are not part of the parent model are either *blocked* (protocol inheritance) or *hidden* (projection inheritance). Similar ideas have been presented in [54], in which the authors distinguish *invocation consistency* and *observation consistency*. These notions correspond to the notions of Basten et al. mentioned above [53]. Focussing on object life cycles, Schrefl and Stumptner build upon this work and further distinguish *weak invocation consistency* and *strong invocation consistency* [55]. They argue that there is no exclusive choice between invocation consistency and observation consistency. The former implies inheritance of the interface, while the latter also enforces that added activities do not interfere with the inherited interface. The boolean characteristics of these notions have been criticized in [56] as inadequate for many process modeling scenarios. Due to their grounding on bisimulation, all existing notions of behaviour inheritance are not applicable for evaluating alignment consistency. Our case study showed that even the notion of trace equivalence, which is weaker than any bisimulation based criterion, does not match the consistency requirements of real-world processes in an alignment setting.

The question of *process similarity* has been addressed from various angles. Here, we focus on measures that consider behavioural aspects and that can be applied once correspondences have been established. The authors of [46] introduced an approach for merging statechart specifications that takes behavioural aspects into account. That is, the preservation of bisimilarity is considered in the similarity score. Also focussing on behavioural aspects, [57], [58] introduce similarity measures based on an edit distance between workflows. Such an edit distance might be based on the language of the workflow, the underlying automaton, or based on the n-gram representation of the language. A similar approach is also taken in [59], in which the authors measure similarity based on high-level change operations that are needed to transform one model into another. Inspired by work in the field of schema matching, similarity flooding has also been applied in order to measure similarity of process models [60]. Close to our behavioural abstraction of a behavioural profile are causal footprints as introduced in [50]. The authors also show how the footprints can be leveraged to determine the similarity between process models. All these similarity notions are expensive in terms of calculation. Behavioural profiles, and its related degree of profile consistency, are an important contribution to this field as they can be calculated efficiently.

Similarity measures are also at the core of process mining, which aims at constructing models from event logs. As mentioned above, similar behavioural relations, but not exactly those of behavioural profiles, are used in [28] in order to characterise the process behaviour. Further on, [56] propose a similarity measure for process mining that is based on probabilities of certain process executions. Focussing on the detection of differences between process models rather than similarities, the work presented in [61], [62] provides a systematic framework of diagnosis and resolution of such mismatches. The identification of process similarity is also an important pillar for work on process model integration [39], [63], [64], [65] and ranking results from process model queries [66], [67], [68]. The concept of a consistency notion, and its efficient calculation based on behavioural profiles, makes it possible to implement search features in process modelling tools, where users expect an immediate result to a process model query.

Closely related to process similarities are means for modelling *process variability*. Various approaches for effective management and configuration of process model variants have been presented in recent years. They are inspired by the work in the field of Software Product Line Engineering (SPLE) [69], [70], [71]. In the field of process modelling, variants might be derived from reference models via model projection [72] or explicit configuration mechanisms. Such mechanisms that extend process modelling languages with configuration capabilities have been presented for EPCs [73], [74], YAWL [18], BPEL [18], UML activity diagrams [75], [76], BPMN [76], and WF-systems [77]. Similar to the properties that are preserved when generating a member of a product line via refinement (cf., [78]), most of these process configuration approaches also define structural and behavioural properties that are preserved. For instance, a configuration of a sound WF-system will be a sound WF-system as well, using the approach specified in [77]. However, none of the aforementioned approaches investigates the degree of behavioural consistency between process variants. Therefore, consistency measures based on behavioural profiles allow for investigations about the behaviour that is preserved by configuration operations and enable quantification of behaviour preservation.

8 CONCLUSION & FUTURE WORK

Process models play an important role to bridge the gap between business requirements and system specifications. In this article, we have discussed alignment issues between related process models at different abstract levels and different perspectives. More concisely, we have addressed the research challenge of defining a notion of consistency between process models that is more adequate to this problem than existing notions of behavioural equivalence. We propose the concept of a behavioural profile that captures the essential behavioural constraints of a process model. Such behavioural profiles are used for the definition of the formal notion of profile consistency. Behavioural profiles provide three major advantages in contrast to the existing notion of trace equivalence and consistency measures that build up it. First, behavioural profiles are less sensitive to projections than trace equivalence, as behavioural profiles remain unchanged even if additional start and end branches are introduced. Second, the structure of a behavioural profile provides us with a straight-forward way to define a degree of profile consistency ranging from 0 to 1.0. Finally, the concept of a behavioural profile builds on formal properties of free-choice Petri nets. We proved that profile consistency can be checked for sound free-choice WF-systems in $O(n^3)$ time with n nodes. All concepts proposed in this article have been implemented to demonstrate the applicability of our approach. Furthermore, we utilized the process models of the SAP reference model for an thorough validation.

There are several directions for future research based on behavioural profiles. We have emphasized the fact that different interrelated process models and variants are utilized for the development of process-aware information systems. While we define methods for efficiently calculating the behavioural profile, there is currently no easy way back from the profile to a process model. We are optimistic that algorithms can be defined to synthesize a process model from a behavioural profile, as there exist synthesis techniques to build Petri nets from transition systems [79] and from traces [28]. Such algorithms might not only take one profile as input. We are currently experimenting with building integrated process models from two behavioural profiles and their alignment.

While most control flow aspects of existing process modelling languages can be expressed as free-choice nets, we already mentioned that some features of languages like BPMN can only be formalised as non free-choice constructs (cf., Section 3). Currently, we benefit from efficient calculation methods for free-choice nets. However, these non free-choice constructs might require other ways of calculation. Here, techniques for alleviating the state explosion problem known from the field of model checking (cf., [80]) might be applied in order to compute a behavioural profile. We are also currently investigating to which extent the non free-choice parts of a process model can be treated separately using graph parsing techniques such as [81]. Further on, process models typically cover aspects beyond control flow such as data and resources, such that they should also be considered in the consistency calculation. Again, model projections, e.g., a certain data object is present only in one model, should not be penalised. Instead, behavioural inconsistencies, e.g., differing data access or contradicting role assignments, have to be quantified.

For calculating the degree of consistency, we are currently using boolean matches. Identifying correspondences is directly within the scope of consistency calculation, such that there is potential for including information about the confidence of a correspondence in the consistency measure. Many matching approaches extract correspondences based on similarity scores [36], emphasizing the certain correspondences have a higher confidence than others. That, in turn, might be considered in our consistency measures. It is interesting to see that the equivalence criteria of the linear time – branching time spectrum have already been lifted from transition systems to metric transitions systems in [82]. In such a metric transitions system, states are associated with predicate valuations. Adapting this concept to our alignment setting would enable us taking confidence of correspondences into account. Correspondences established with high confidence might have a bigger impact on the consistency as correspondences that are considered to be uncertain.

REFERENCES

[1] E. Yourdon, *Modern structured analysis*. Yourdon Press Upper Saddle River, NJ, USA, 1989.

[2] J. Luftman, R. Papp, and T. Brier, "Enablers and inhibitors of business-IT alignment," *Communications of the AIS*, vol. 1, no. 3, 1999.

[3] E. Kindler, "Model-based software engineering and process-aware information systems," *LNCS Transactions on Petri Nets and Other Models of Concurrency*, vol. 2, pp. 27–45, 2009.

[4] M. Henkel, J. Zdravkovic, and P. Johannesson, "Service-based processes: Design for business and technology," in *ICSOC*, M. Aiello, M. Aoyama, F. Curbera, and M. P. Papazoglou, Eds. ACM, 2004, pp. 21–29.

[5] B. Andersson, M. Bergholtz, A. Edirisuriya, T. Ilayperuma, and P. Johannesson, "A Declarative Foundation of Process Models," in *CAiSE*, ser. Lecture Notes in Computer Science, O. Pastor and J. F. e Cunha, Eds., vol. 3520. Springer, 2005, pp. 233–247.

[6] J. Koehler, R. Hauser, J. M. Küster, K. Ryndina, J. Vanhatalo, and M. Wahler, "The role of visual modeling and model transformations in business-driven development," *Electr. Notes Theor. Comput. Sci.*, vol. 211, pp. 5–15, 2008.

[7] OMG, *Business Process Modeling Notation (BPMN) 1.2*, January 2009.

[8] Alexandre Alves et al., "Web Services Business Process Execution Language Version 2.0," OASIS, Tech. Rep., January 2007.

[9] V. Grover, K. Fiedler, and J. Teng, "Exploring the Success of Information Technology Enabled Business Process Reengineering," *IEEE Transactions on Engineering Management*, vol. 41, no. 3, pp. 276–284, August 1994.

[10] C. Rolland and N. Prakash, "Bridging the Gap Between Organisational Needs and ERP Functionality," *Requirements Engineering*, vol. 5, no. 3, pp. 180–193, October 2000.

[11] T. Kühne, "Matters of (Meta-)Modeling," *Software and System Modeling*, vol. 5, no. 4, pp. 369–385, 2006.

[12] IEEE, "Standard Glossary of Software Engineering Terminology," 1990.

[13] G. Rull, C. Farré, E. Teniente, and T. Urpí, "Validation of mappings between schemas," *Data Knowl. Eng.*, vol. 66, no. 3, pp. 414–437, 2008.

[14] R. van Glabbeek, *Handbook of Process Algebra*. Elsevier, 2001, ch. The linear time - brancing time spectrum I. The semantics of concrete, sequential processes, pp. 3–99.

[15] R. J. van Glabbeek and U. Goltz, "Refinement of actions and equivalence notions for concurrent systems," *Acta Inf.*, vol. 37, no. 4/5, pp. 229–327, 2001.

[16] M. Weidlich, M. Weske, and J. Mendling, "Change propagation in process models using behavioural profiles," in *Proceedings of the IEEE International Conference on Services Computing (SCC)*. IEEE Computer Society, 2009.

[17] G. J. Ramackers, "Integrated object modelling," Ph.D. dissertation, Leiden University, Thesis Publishers Amsterdam, 1994.

[18] F. Gottschalk, W. M. P. van der Aalst, M. H. Jansen-Vullers, and M. L. Rosa, "Configurable workflow models," *Int. J. Cooperative Inf. Syst.*, vol. 17, no. 2, pp. 177–221, 2008.

[19] M. Weidlich, A. Barros, J. Mendling, and M. Weske, "Vertical Alignment of Process Models - How can we get there?" in *CAiSE 2009 Workshop Proceedings - 10th Workshop on Business Process Modeling, Development, and Support (BPMDS)*, ser. LNBIP, S. Nurcan, R. Schmidt, P. Soffer, and R. Ukor, Eds., no. 29. Springer, 2009, pp. 71–84.

[20] W. M. P. van der Aalst, "The application of petri nets to workflow management," *Journal of Circuits, Systems, and Computers*, vol. 8, no. 1, pp. 21–66, 1998.

[21] N. Lohmann, "A feature-complete Petri net semantics for WS-BPEL 2.0," in *Web Services and Formal Methods, Forth International Workshop, WS-FM 2007 Brisbane, Australia, September 28-29, 2007, Proceedings*, ser. Lecture Notes in Computer Science, M. Dumas and R. Heckel, Eds., vol. 4937. Springer-Verlag, 2008, pp. 77–91.

[22] R. Dijkman, M. Dumas, and C. Ouyang, "Semantics and Analysis of Business Process Models in BPMN," *Information and Software Technology (IST)*, vol. 50, no. 12, pp. 1281–1294, 2009.

[23] J. Mendling, *Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness*, ser. Lecture Notes in Business Information Processing. Springer, 2008, vol. 6.

[24] A. ter Hofstede and W. van der Aalst, "Yawl: yet another workflow language," *Information Systems*, vol. 30, no. 4, pp. 245–275, 2005.

[25] C. Eichner, H. Fleischhack, R. Meyer, U. Schrimpf, and C. Stehno, "Compositional semantics for uml 2.0 sequence diagrams using petri nets," in *SDL Forum*, ser. Lecture Notes in Computer Science, A. Prinz, R. Reed, and J. Reed, Eds., vol. 3530. Springer, 2005, pp. 133–148.

[26] R. Eshuis and R. Wieringa, "Tool support for verifying uml activity diagrams," *IEEE Trans. Software Eng.*, vol. 30, no. 7, pp. 437–447, 2004.

[27] N. Lohmann, E. Verbeek, and R. M. Dijkman, "Petri net transformations for business processes - a survey," *T. Petri Nets and Other Models of Concurrency*, vol. 2, pp. 46–63, 2009.

[28] W. M. P. van der Aalst, T. Weijters, and L. Maruster, "Workflow mining: Discovering process models from event logs," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 9, pp. 1128–1142, 2004.

[29] R. Lipton, "The Reachability Problem Requires Exponential Space," Department of Computer Science, Yale University, Tech. Rep., 1975.

[30] A. Valmari, "The state explosion problem," in *Petri Nets*, ser. Lecture Notes in Computer Science, W. Reisig and G. Rozenberg, Eds., vol. 1491. Springer, 1996, pp. 429–528.

[31] J. Desel and J. Esparza, *Free choice Petri nets*. New York, NY, USA: Cambridge University Press, 1995.

[32] W. M. P. van der Aalst, "Verification of workflow nets," in *ICATPN*, ser. Lecture Notes in Computer Science, P. Azéma and G. Balbo, Eds., vol. 1248. Springer, 1997, pp. 407–426.

[33] B. Kiepuszewski, A. H. M. ter Hofstede, and W. M. P. van der Aalst, "Fundamentals of control flow in workflows," *Acta Informatica*, vol. 39, no. 3, pp. 143–209, 2003.

[34] A. Kovalyov and J. Esparza, "A polynomial algorithm to compute the concurrency relation of free-choice signal transition graphs," in *Prof. of the International Workshop on Discrete Event Systems, WODES'96*. Edinburgh: The Institution of Electrical Engineers, 1996, pp. 1–6.

[35] S. Warshall, "A theorem on boolean matrices," *Journal of the ACM (JACM)*, vol. 9, no. 1, pp. 11–12, 1962.

[36] J. Euzenat and P. Shvaiko, *Ontology Matching*. Springer-Verlag, 2007.

[37] M. Dumas, L. García-Bañuelos, and R. M. Dijkman, "Similarity search of business process models," *IEEE Data Eng. Bull.*, vol. 32, no. 3, pp. 23–28, 2009.

[38] T. A. Curran, G. Keller, and A. Ladd, *SAP R/3 Business Blueprint: Understanding the Business Process Reference Model*. Prentice-Hall, 1997.

[39] V. Pankratius and W. Stucky, "A formal foundation for workflow composition, workflow view definition, and workflow normalization based on petri nets." in *Conceptual Modelling 2005, Second Asia-*

Pacific Conference on Conceptual Modelling (APCCM2005), Newcastle, NSW, Australia, January/February 2005, ser. CRPIT, S. Hartmann and M. Stumptner, Eds., vol. 43. Australian Computer Society, 2005, pp. 79–88.

[40] J. Mendling, H. M. W. Verbeek, B. F. van Dongen, W. M. P. van der Aalst, and G. Neumann, "Detection and prediction of errors in epcs of the sap reference model," *Data Knowl. Eng.*, vol. 64, no. 1, pp. 312–329, 2008.

[41] B. van Dongen, M. Jansen-Vullers, H. Verbeek, and W. van der Aalst, "Verification of the sap reference models using epc reduction, state space analysis, and invariants," *Computers in Industry*, vol. 58, no. 6, pp. 578–601, 2007.

[42] G. Decker and J. Mendling, "Process instantiation," *Data & Knowledge Engineering (DKE)*, vol. 68, pp. 777–792, 2009.

[43] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541–580, 1989.

[44] E. Rahm and P. A. Bernstein, "A survey of approaches to automatic schema matching," *VLDB Journal*, vol. 10, no. 4, pp. 334–350, 2001.

[45] A. Doan and A. Y. Halevy, "Semantic integration research in the database community: A brief survey," *AI Magazine*, vol. 26, no. 1, pp. 83–94, 2005.

[46] S. Nejati, M. Sabetzadeh, M. Chechik, S. M. Easterbrook, and P. Zave, "Matching and merging of statecharts specifications," in *ICSE*. IEEE Computer Society, 2007, pp. 54–64.

[47] R. Dijkman, M. Dumas, L. García-Bauelos, and R. Käärik, "Aligning business process models," in *13th International IEEE Enterprise Distributed Object Computing Conference (ECOC)*, 2009.

[48] M. Weidlich, R. Dijkman, and J. Mendling, "The ICoP framework: Identification of correspondences between process models," in *CAiSE*, 2010, to appear.

[49] M. Ehrig, A. Koschmider, and A. Oberweis, "Measuring similarity between semantic business process models," in *Conceptual Modelling 2007, Proceedings of the Fourth Asia-Pacific Conference on Conceptual Modelling (APCCM 2007)*, J. Roddick and A. Hinze, Eds., vol. 67. Ballarat, Victoria, Australia: Australian Computer Science Communications, 2007, pp. 71–80.

[50] B. F. van Dongen, R. M. Dijkman, and J. Mendling, "Measuring Similarity between Business Process Models," in *CAiSE*, ser. Lecture Notes in Computer Science, Z. Bellahsene and M. Léonard, Eds., vol. 5074. Springer, 2008, pp. 450–464.

[51] J. Hidders, M. Dumas, W. M. P. van der Aalst, A. H. M. ter Hofstede, and J. Verelst, "When are two Workflows the Same?" in *CATS*, ser. CRPIT, M. D. Atkinson and F. K. H. A. Dehne, Eds., vol. 41. Australian Computer Society, 2005, pp. 3–11.

[52] D. Harel and O. Kupferman, "On object systems and behavioral inheritance," *IEEE Trans. Software Eng.*, vol. 28, no. 9, pp. 889–903, 2002.

[53] T. Basten and W. M. P. van der Aalst, "Inheritance of Behavior," *Journal of Logic and Algebraic Programming (JLAP)*, vol. 47, no. 2, pp. 47–145, 2001.

[54] J. Ebert and G. Engels, " Observable or Invocable Behaviour - You Have to Choose," Department of Computer Science, Leiden University, Technical Report 94-38, December 1994.

[55] M. Schrefl and M. Stumptner, "Behavior-consistent specialization of object life cycles," *ACM Trans. Softw. Eng. Methodol.*, vol. 11, no. 1, pp. 92–148, 2002.

[56] A. K. A. de Medeiros, W. M. P. van der Aalst, and A. J. M. M. Weijters, "Quantifying process equivalence based on observed behavior," *Data Knowl. Eng.*, vol. 64, no. 1, pp. 55–74, 2008.

[57] A. Wombacher, "Evaluation of technical measures for workflow similarity based on a pilot study," in *OTM Conferences (1)*, ser. Lecture Notes in Computer Science, R. Meersman and Z. Tari, Eds., vol. 4275. Springer, 2006, pp. 255–272.

[58] A. Wombacher and M. Rozie, "Evaluation of workflow similarity measures in service discovery," in *Service Oriented Electronic Commerce*, ser. LNI, M. Schoop, C. Huemer, M. Rebstock, and M. Bichler, Eds., vol. 80. GI, 2006, pp. 51–71.

[59] C. Li, M. Reichert, and A. Wombacher, "On measuring process model similarity based on high-level change operations," in *ER*, ser. Lecture Notes in Computer Science, Q. Li, S. Spaccapietra, E. S. K. Yu, and A. Olivé, Eds., vol. 5231. Springer, 2008, pp. 248–264.

[60] T. Madhusudan, J. L. Zhao, and B. Marshall, "A case-based reasoning framework for workflow model management," *Data Knowl. Eng.*, vol. 50, no. 1, pp. 87–115, 2004.

[61] J. M. Küster, C. Gerth, A. Förster, and G. Engels, "Detecting and resolving process model differences in the absence of a change log," in *BPM*, ser. Lecture Notes in Computer Science, M. Dumas, M. Reichert, and M.-C. Shan, Eds., vol. 5240. Springer, 2008, pp. 244–260.

[62] R. M. Dijkman, "Diagnosing differences between business process models," in *BPM*, ser. Lecture Notes in Computer Science, M. Dumas, M. Reichert, and M.-C. Shan, Eds., vol. 5240. Springer, 2008, pp. 261–277.

[63] G. Preuner, S. Conrad, and M. Schrefl, "View integration of behavior in object-oriented databases." *Data & Knowledge Engineering*, vol. 36, no. 2, pp. 153–183, 2001.

[64] G. Grossmann, Y. Ren, M. Schrefl, and M. Stumptner, "Behavior based integration of composite business processes." in *BPM 2005, Proceedings*, ser. Lecture Notes in Computer Science, W. van der Aalst, B. Benatallah, F. Casati, and F. Curbera, Eds., vol. 3649. Springer, 2005, pp. 186–204.

[65] J. Mendling and C. Simon, "Business Process Design by View Integration," in *Proceedings of BPM Workshops 2006*, ser. Lecture Notes in Computer Science, J. Eder and S. Dustdar, Eds., vol. 4103. Vienna, Austria: Springer-Verlag, 2006, pp. 55–64.

[66] M. Klein and A. Bernstein, "Towards high-precision service retrieval," *IEEE Internet Computing*, vol. 8, no. 1, pp. 30–36, 2004.

[67] M. Momotko and K. Subieta, "Process query language: A way to make workflow processes more flexible." in *ADBIS 2004*, ser. Lecture Notes in Computer Science, G. Gottlob, A. Benczúr, and J. Demetrovics, Eds., vol. 3255. Springer, 2004, pp. 306–321.

[68] A. Awad, G. Decker, and M. Weske, "Efficient compliance checking using bpmn-q and temporal logic," in *BPM*, ser. Lecture Notes in Computer Science, M. Dumas, M. Reichert, and M.-C. Shan, Eds., vol. 5240. Springer, 2008, pp. 326–341.

[69] K. Pohl, G. Böckle, and F. van der Linden, *Software Product Line Engineering. Foundations, Principles, and Techniques*. Springer, 2005.

[70] D. S. Batory, C. Johnson, B. MacDonald, and D. von Heeder, "Achieving extensibility through product-lines and domain-specific languages: A case study," in *ICSR*, ser. Lecture Notes in Computer Science, W. B. Frakes, Ed., vol. 1844. Springer, 2000, pp. 117–136.

[71] M. L. Griss, "Implementing product-line features with component reuse," in *ICSR*, ser. Lecture Notes in Computer Science, W. B. Frakes, Ed., vol. 1844. Springer, 2000, pp. 137–152.

[72] J. Becker, P. Delfmann, and R. Knackstedt, *Adaptive Reference Modeling. Integrating Configurative and Generic Adaptation Techniques for Information Models*. Physica, 2007, pp. 23–49.

[73] M. Rosemann and W. Aalst, "A Configurable Reference Modelling Language," *Information Systems*, vol. 32, pp. 1–23, 2007.

[74] H. A. Reijers, R. S. Mans, and R. A. van der Toorn, "Improved model management with aggregated business process models," *Data Knowl. Eng.*, vol. 68, no. 2, pp. 221–243, 2009.

[75] M. Razavian and R. Khosravi, "Modeling variability in business process models using uml," in *ITNG*. IEEE Computer Society, 2008, pp. 82–87.

[76] A. Schnieders and F. Puhlmann, "Variability mechanisms in e-business process families," in *BIS*, ser. LNI, W. Abramowicz and H. C. Mayr, Eds., vol. 85. GI, 2006, pp. 583–601.

[77] W. van der Aalst, M. Dumas, F. Gottschalk, A. ter Hofstede, M. L. Rosa, and J. Mendling, "Correctness-preserving configuration of business process models," in *Proc. of Fundamental Approaches to Software Engineering (FASE 2008)*, ser. Lecture Notes in Computer Science, J. L. Fiadeiro and P. Inverardi, Eds., vol. 4961. Springer, 2008.

[78] D. S. Batory, J. N. Sarvela, and A. Rauschmayer, "Scaling step-wise refinement," *IEEE Trans. Software Eng.*, vol. 30, no. 6, pp. 355–371, 2004.

[79] J. Cortadella, M. Kishinevsky, L. Lavagno, and A. Yakovlev, "Deriving Petri Nets from Finite Transition Systems," *IEEE Transactions on Computers*, vol. 47, no. 8, pp. 859–882, 1998.

[80] J. Esparza and K. Heljanko, *Unfoldings: a partial-order approach to model checking*. Springer, 2008.

[81] J. Vanhatalo, H. Völzer, and J. Koehler, "The refined process structure tree," *Data & Knowledge Engineering (DKE)*, vol. 68, pp. 793–818, 2009.

[82] L. de Alfaro, M. Faella, and M. Stoelinga, "Linear and branching system metrics," *IEEE Trans. Software Eng.*, vol. 35, no. 2, pp. 258–273, 2009.

[83] M. Dumas, M. Reichert, and M.-C. Shan, Eds., *Business Process Management, 6th International Conference, BPM 2008, Milan, Italy, September 2-4, 2008. Proceedings*, ser. Lecture Notes in Computer Science, vol. 5240. Springer, 2008.

[84] W. B. Frakes, Ed., *Software Reuse: Advances in Software Reusability, 6th International Conerence, ICSR-6, Vienna, Austria, June 27-29, 2000, Proceedings*, ser. Lecture Notes in Computer Science, vol. 1844. Springer, 2000.

[85] K. Jensen and W. M. P. van der Aalst, Eds., *Transactions on Petri Nets and Other Models of Concurrency II, Special Issue on Concurrency in Process-Aware Information Systems*, ser. Lecture Notes in Computer Science, vol. 5460. Springer, 2009.

PLACE
PHOTO
HERE

Matthias Weidlich received the MSc degree in computer science from the Hasso Plattner Institute (HPI), University of Potsdam, Germany, in 2008. He has been a research assistant at the HPI since 2008, working in the Business Process Technology research group. His research focuses on the study of relations between process models. His research interests include process modelling languages, modelling methodologies, the application of formal methods for behavioural analysis.

PLACE
PHOTO
HERE

Jan Mendling is a Junior-Professor with the Institute of Information Systems at Humboldt-Universit́at zu Berlin, Germany. His research areas include Business Process Management, Conceptual Modelling and Enterprise Systems. He has published more than 100 research papers and articles, among others in ACM Transactions on Software Engineering and Methodology, Information Systems, Data & Knowledge Engineering, Formal Aspects of Computing, and Information and Software Technology. He is member of the editorial board of two international journals. His Ph.D. thesis has won the Heinz-Zemanek-Award of the Austrian Computer Society and the German Targion-Award for dissertations in the area of strategic information management. He is one of the founders and chair of the Berlin BPM Community of Practice (http://www.bpmb.de) and organizer of several academic events on process management. He is program co-chair of the International Conference on Business Process Management 2010.

PLACE
PHOTO
HERE

Mathias Weske is a professor at the Hasso Plattner Institute, University of Potsdam, Germany, where he leads the Business Process Technology research group. His research interests include business process management, process choreographies, process modelling methodologies, and software architectures for process-oriented information systems. He leads Oryx, an open source project on business process management. Dr. Weske has published twelve books and over 90 scientific papers in journals and conferences. He is on the steering committee of the BPM conference series and a member of ACM, IEEE, and GI.