

Change Propagation in Process Models using Behavioural Profiles

Matthias Weidlich, Mathias Weske
Hasso-Plattner-Institute, Potsdam, Germany
Email: {matthias.weidlich,weske}@hpi.uni-potsdam.de

Jan Mendling
Humboldt-Universität zu Berlin, Germany
Email: jan.mendling@wiwi.hu-berlin.de

Abstract—Business process change is at the very core of business process management, which aims at enabling flexible adaptation to changing business needs. However, the wide variety of drivers for business process modelling initiatives, reaching from business evolution to process enactment, results in multiple models that overlap in content due to serving different purposes. That, in turn, imposes serious challenges for the propagation of changes between these process models. Given a change in one model, this paper introduces an approach to determine a change region in another model by exploiting the behavioural profile of corresponding activities. It, therefore, supports the process of change propagation and eases the synchronisation of process models significantly. As a major contribution, our approach can handle changes in pairs of models, even if they are not defined in terms of a hierarchical refinement.

Keywords—Change propagation, behaviour equivalence, process adaptation, model alignment

I. INTRODUCTION

Nowadays, Business Process Management (BPM) has a broad field of application, reaching from process evolution to process enactment. That, in turn, results in manifold requirements for BPM methods and techniques. The *purpose* guides the creation of every particular process model. It is a consequence of this observation that companies create different models for the same process. These models reside on different levels of abstraction and assume different modelling perspectives depending on what is appropriate with respect to the modelling goal. These differences, for instance, lead to the widely known ‘Business-IT Gap’ (cf., [1], [2]) between models describing the operational processes in an organisation and the actual implementation in an IT-landscape. The key challenge in this area relates to the fact that the differences between such models cannot be traced back to hierarchical refinements or customised views on a core process. The reason for this non-trivial relationship stems, among others, from the constraints that a particular process execution engine imposed on *how* the goals of a process have to be achieved. On the other hand, a business analyst can simply focus on *what* needs to be done when executing the process. These differences are of fundamental nature and need to be bridged by an explicit alignment.

The flexibility to adapt business processes in order to respond to changing business needs is at the very heart of BPM. Therefore, the *propagation of changes* between

several related process models is a major use case for model alignment [3]. According to Gartner, change is of high relevance to the key elements of the BPM discipline, which are ‘*keeping the business process model in sync with process execution [and] enabling rapid iteration of processes and underlying systems for continuous process improvement and optimisation*’ [4]. The importance of change in BPM is also illustrated by the impressive numbers published by Logica Consulting in a recent white paper. They state that ‘*the majority of European companies spend 10% to 55% of the average profit margin on business process change*’ [5].

This paper presents a novel approach to change propagation between business process models. Its central contribution is the definition and application of a technique for dealing with overlapping process models that are not defined in terms of a hierarchical refinement. This technique is based on the notion of a behavioural profile which captures a set of dedicated behavioural aspects of a process model. If we know which activities correspond to each other in the two process models, we show how their behavioural profiles can be leveraged for change propagation. Given a change in the source model, our approach isolates a potential *change region* in the target model grounded on the behavioural profile of corresponding activities. In this way, process modellers can quickly assess the necessity to propagate the change. If change propagation seems to be appropriate, the change region spots the position where to extend the model.

The remainder of this paper is structured as follows. Section II introduces an exemplary setting for change propagation. Afterwards, Section III introduces preliminaries, such as our formal model. Section IV defines our approach for change propagation. Based thereon, we provide a discussion of our results in Section V and review related work in Section VI. Finally, Section VII concludes the paper.

II. PROCESS MODEL ALIGNMENT

In order to illustrate the context of process model alignment, Figure 1 shows an alignment setting consisting of two process models depicting a lead-to-order process using the Business Process Modeling Notation (BPMN) [6]. The upper model (A) gives an intuitive overview of the major processing steps, from the first contact with the customer to the closing of a deal. The lower model (B) provides a more fine-grained view and illustrates the support for

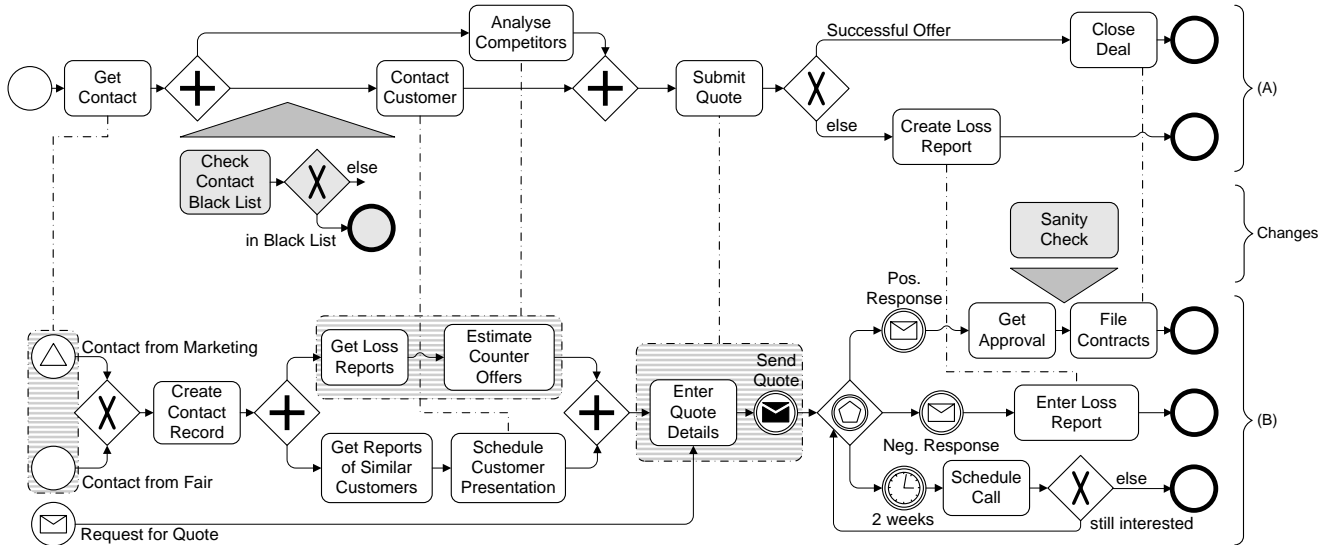


Figure 1. Two process models aligned by model correspondences along with two model changes

the high-level process in a specific CRM system. Here, the assumed perspective is different. The lower model focusses at different stages solely on that part of the activities of the upper model that is supported by the CRM directly, rather than specifying the whole task to be done. Further on, the lower model specifies additional entry points, e.g., the possibility of receiving requests for quote directly is taken into account.

Clearly, both models are closely related, which is manifested in correspondences between elements of both models. In Figure 1, these correspondences are highlighted as dashed lines between elements or groups of elements, respectively. We assume these correspondences to be defined always between two elements of two models. We use the grouping in Figure 1 in this context for illustration purposes, e.g., activity *Submit Quote* in the middle of the upper model corresponds to the activity *Enter Quote Details* and the event *Send Quote* in the lower model. However, this does not imply the interpretation of both elements in the lower model *jointly* corresponding to the activity in the upper model (cf., the correspondences for the activity *Get Contact* of model (A)). For our approach, we assume these correspondences to be given. Obviously, correspondences cannot be defined manually for real world scenarios, owing to the pure number of model elements. Techniques that allow for automatic or at least semi-automatic definition of correspondences need to be exploited, which build on linguistic analysis of element naming, domain specific ontologies, or analysis of data dependencies (see [7] for an overview). A discussion of these techniques is, however, beyond the scope of this paper.

Besides the aligned process models, Figure 1 also depicts two change operations, which illustrates that changes can originate from all abstraction levels. On the left-hand side, a

high-level management decision requires the insertion of an additional fragment in model (A), i.e., the check whether the black list contains the current contact. On the right-hand side, the CRM process is changed and now features a sanity check before the actual contracts are filed. Both changes might not be propagated under all circumstances. Nevertheless, any decision of whether and how to propagate these changes has to be guided by illustrating the change location in the model under consideration. For real-world scenarios the intrinsic complexity of process models prevents a manual change analysis by just looking at the respective models.

III. PRELIMINARIES

This section introduces our formal model. First, Section III-A defines the notion of a process model. Second, we define behavioural profiles that capture behavioural characteristics of a process model in Section III-B. Finally, Section III-C introduces basic notions of model alignment.

A. Process Models

Our notion of a process model is based on a graph containing activity nodes and control nodes, which, in turn, captures the commonalities of process description languages. Thus, the subset of BPMN used in our initial example can be traced back to the following definition of a process model.

Definition 1: (Process Model) A *process model* is a tuple $P = (A, I, C, F, T)$ with

- A as a non-empty set of activity nodes, and C as a set of control nodes, A and C are disjoint,
- $I \subseteq A$ as a set of initial activities,
- $F \subseteq (A \cup C) \times (A \setminus I \cup C)$ as the flow relation, and
- $T : C \mapsto \{and, or, xor\}$ as a function that assigns a type to control nodes.

In the remainder, the set of all nodes is denoted by $N = A \cup C$. Further on, we require all process models to be connected, that is $\forall n \in N [\exists i \in I [i F^* n]]$ with F^* as the transitive reflexive closure of F .

Our approach of propagating changes is based on the execution traces of a process model. We do not formalise the behavioural semantics of a process model, but assume an interpretation of the model following on common execution semantics. Such semantics are inherent for Petri net-based modelling languages (cf., [8]) or have been presented for informal modelling languages such as BPMN, EPCs, or UML activity diagrams [9], [10], [11]. Please note that our approach is independent of the concrete definition of behavioural semantics. In particular, we do not assume a certain definition of semantics for the inclusive OR construct, which raises serious issues in cyclic structures. We solely expect the definition of such a semantics and require the process traces to fulfil a syntactical requirement. That is, we expect a trace to be given not only as a list of nodes, but to take the flow arcs between two nodes into account.

Definition 2: (Process Traces) The set of *process traces* \mathcal{T}_P for a process model $P = (A, I, C, F, T)$ is a set of lists of the form $I \cdot (F \cdot N)^*$, such that a list entry contains the execution order of nodes along with the flow arcs that led to an execution.

B. Behavioural Profile

The behavioural profile aims at capturing behavioural aspects of a process in a fine-grained manner. That is, it consists of three relations between nodes of a process graph. These relations are based on the notion of *weak order*. Two nodes or flow arcs of a process model are in weak order, if there exists a trace in which one node occurs after the other. Note that we require only the *existence* of such a trace. Thus, weak order does not have to hold for all traces of the model.

Definition 3 (Weak Order Relation): Let $P = (A, I, C, F, T)$ be a process model and \mathcal{T}_P its set of traces. The *weak order relation* $\succ_P \subseteq ((N \cup F) \times (N \cup F))$ contains all pairs (x, y) , such that there is a trace $\sigma = n_1, \dots, n_m$ in \mathcal{T}_P with $j \in \{1, \dots, m-1\}$ and $j < k \leq m$ for which holds $n_j = x$ and $n_k = y$.

Two nodes or flow arcs of a process model might be related by weak order in three different ways, which, in turn, defines the characteristic relations of the behavioural profile.

Definition 4 (Behavioural Profile): Let $P = (A, I, C, F, T)$ be a process model. A pair $(x, y) \in ((N \cup F) \times (N \cup F))$ is in one of the following relations:

- The *strict order relation* \rightsquigarrow_P , if $x \succ_P y$ and $y \not\succ_P x$.
- The *exclusiveness relation* $+_P$, if $x \not\succ_P y$ and $y \not\succ_P x$.
- The *observation concurrency relation* $||_P$, if $x \succ_P y$ and $y \succ_P x$.

The set of all three relations is the *behavioural profile* of P .

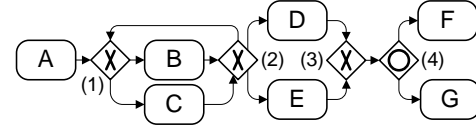


Figure 2. An example for which, among others, these relations hold: $A \rightsquigarrow D$, $((2), D) \rightsquigarrow G$, $D + E$, $B || C$, and $((4), F) || G$

Figure 2 illustrates the relations of the behavioural profile by means of an example. Here, it holds $A \rightsquigarrow D$ as the strict order does not imply the actual occurrence. As the relations are also defined between nodes and flow arcs, it holds also $A \rightsquigarrow ((2), D)$. Further on, $D + E$ as both activities will never occur in a single trace. Note that $B || C$ due to the potential repetition of the choice between both activities. It is also important to notice that all three relations of the behavioural profile are mutually exclusive. That is, any pair of nodes is solely in one of the three relations. Further on, a node is either said to be *exclusive to itself* (e.g., $D + D$) or *observation concurrent to itself* (e.g., $B || B$). The former holds, when a node cannot be repeated, whereas the latter implies that the node is part of a control flow cycle.

C. Changes and Model Alignment

Different kinds of changes have been classified for process models by Weber et al. [12]. Albeit defined in a different context, i.e., instance adaptation, these *change patterns* illustrate the variety of changes for process models. For our approach, we assume that a change can be localised as a dedicated node, the so called *change node*, in the changed process model. This captures not only insertion of activities or process fragments (as in our example in Section II), but is also applicable for other changes. For instance, removal of an activity or process fragment can also be localised by a dedicated node. As we exploit behavioural characteristics for change propagation, however, a change has to be consistent.

Definition 5: (Consistent Change) A change in a process model P is represented by a change node X , such that the behavioural profile for all other nodes of P is not changed.

Note that we might even consider behavioural changes, as for instance the parallelisation of activities. Such a change solely requires to encapsulate the region of activity parallelisation by means of change node, such that behavioural profile does not change for the remaining activities.

Further on, we already mentioned our assumption of correspondences between two process models. In addition, we require both models to be consistent with respect to their behavioural profile.

Definition 6: (Correspondence Relation) The *Correspondence relation* $\sim \subseteq A_1 \times A_2$ contains corresponding activity nodes of two process models $P_1 = (A_1, I_1, C_1, F_1, T_1)$ and $P_2 = (A_2, I_2, C_2, F_2, T_2)$.

Definition 7: (Model Consistency) Two process models P_1 and P_2 are *consistent* w.r.t. a correspondence relation \sim ,

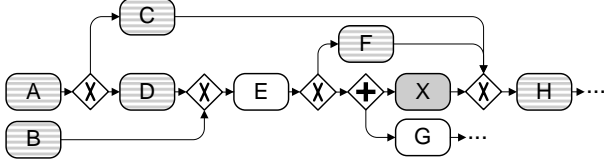


Figure 3. B , D , and H are boundary nodes for the change X , if E is not part of the correspondence relation

iff $\forall (a_1, a_2), (b_1, b_2) \in \sim [(a_1 \succ_{P_1} b_1) \Rightarrow (a_2 \succ_{P_2} b_2)]$.

In the remainder, we use the term *aligned nodes* in order to refer to these nodes of a process model, for which there exists a corresponding node in a second process model that is related by a certain correspondence relation. For two models $P_1 = (A_1, I_1, C_1, F_1, T_1)$ and $P_2 = (A_2, I_2, C_2, F_2, T_2)$ aligned by \sim , we use $A_1^\sim = \{a_1 \in A_1 \mid \exists a_2 \in A_2 [a_1 \sim a_2]\}$ as a short-hand notation.

IV. CHANGE PROPAGATION BASED ON CHARACTERISTIC RELATIONS

This section shows how the behavioural profile can be leveraged in order to propagate changes between aligned process models. First, we give an overview of the approach in Section IV-A. Subsequently, the two fundamental steps of our approach are explained in Section IV-B and IV-C. Afterwards, we discuss the actual algorithm and interpret potential outcomes in Section IV-D.

A. Overview

The general idea of our approach can be summarised as follows. A change operation in the source process model is reflected in its behavioural profile, i.e., its characteristic relations. Therefore, these relations *localise* the change in the source model. Under the assumption that the source and the target model have been aligned to some extent, we are able to exploit the characteristic relations for the corresponding activities in the target model in order to localise a *change region* in the target model. This region, in turn, identifies a part of the target model in which the change of the source model has to be realised.

Definition 8: (Change Region) A *change region* is a subset $C_P \subseteq F$ of flow arcs of a process model $P = (A, I, C, F, T)$.

Initially, the change region contains all flow arcs of the target model. Based thereon, the change region is narrowed using two reductions. On the one hand, the change region is reduced based on *boundary nodes* that directly precede or succeed the change in strict order. On the other hand, reduction is based on *inter-boundary nodes* that are exclusive or observation concurrent to the change.

B. Reduction based on Boundary Nodes

The first kind of reduction requires the identification of the *closest* aligned nodes that are in strict order with the actual change in the source model, the so called *preceding* or

succeeding boundary nodes. The term *closest* relates to the fact that there are no other aligned nodes between a boundary node and the change location, which are also in strict order with the change. We illustrate the notion of boundary nodes by means of the example in Figure 3. Assume that node X represents the change in the model and all activity nodes highlighted in grey are aligned, i.e., they have corresponding nodes in the second model. Then, there are two preceding boundary nodes, namely B and D , whereas H is the only boundary node succeeding the change.

Boundary nodes are captured in our formal model as follows. Note that we distinguish two sets of boundary nodes depending on whether they precede or succeed the change in the source model.

Definition 9 (Boundary Nodes): Let $P_1 = (A_1, I_1, C_1, F_1, T_1)$ be the source process model with a change node X , P_2 a target process model, and \sim a correspondence relation for P_1 and P_2 . The set of *preceding boundary nodes* $PB_{P_1} \subseteq A_1^\sim$ contains all aligned activity nodes a_1 that directly precede the change in strict order, i.e., $(a_1 \rightsquigarrow_{P_1} X)$ and $(\nexists b_1 \in A_1^\sim [a_1 \rightsquigarrow_{P_1} b_1 \rightsquigarrow_{P_1} X])$. The set of *succeeding boundary nodes* SB_{P_1} is defined accordingly.

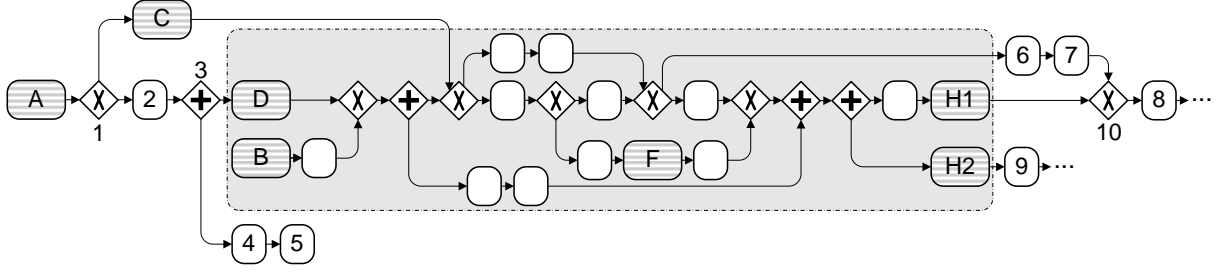
The boundary nodes enable a first reduction of the change region, which is illustrated using the process model in Figure 4(a) as an example for the target model. Further, let the model in Figure 3 to be the source model with X being inserted. The aligned nodes are highlighted in grey and have similar labels in both models (e.g., $H \sim H1$ and $H \sim H2$).

First and foremost, flow arcs of the target model are removed based on their relations to the corresponding boundary nodes $B, D, H1$, and $H2$. That is due to the fact that strict order between boundary nodes and the change in the source model has to be preserved for the corresponding boundary nodes and the change region in the target model. A flow arc is removed, if it meets one of the following requirements.

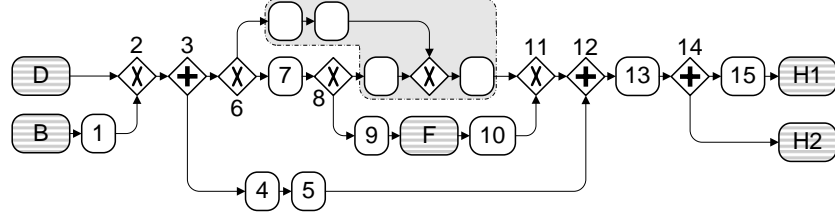
- It precedes a corresponding preceding boundary node in strict order (e.g., flow arc $(3, D)$).
- It succeeds a corresponding succeeding boundary node in strict order (e.g., flow arc $(H2, 9)$).
- It is exclusive to a corresponding boundary node (e.g., flow arc $(1, C)$).
- It is observation concurrent to a corresponding boundary node (e.g., flow arc $(4, 5)$).

Regarding our example, this reduction yields the set of all flow arcs that are fully contained in the region highlighted in grey in Figure 4(a).

In order to define the reduction formally, we introduce a short-hand notation for the corresponding boundary nodes in the target model. For $P_1 = (A_1, I_1, C_1, F_1, T_1)$ as the source process model, $P_2 = (A_2, I_2, C_2, F_2, T_2)$ as the target process model, and \sim as a correspondence relation, we



(a) The grey part depicts the change region after a reduction based on the *boundary nodes*



(b) The grey part is depicts the change region after a reduction based on the *inter-boundary nodes*

Figure 4. The change region in the target model is determined via step-wise reduction

define $PB_{P_2} = \{a_2 \in A_2 \mid \exists a_1 \in PB_{P_1} [a_1 \sim a_2]\}$ as the set containing all nodes that correspond to preceding boundary nodes of the source model. Again, the set SB_{P_2} for succeeding corresponding boundary nodes is derived in the same way and $B_{P_2} = PB_{P_2} \cup SB_{P_2}$ denotes all boundary nodes. Based thereon, we specify the reduction.

Definition 10 (Boundary Node Reduction): Let P_1 be the source process model, $P_2 = (A_2, I_2, C_2, F_2, T_2)$ the target process model, and \sim a correspondence relation for P_1 and P_2 . The *boundary node reduction* creates a change region \mathcal{C}_{P_2} , such that

$$\begin{aligned} \mathcal{C}_{P_2} &= F_2 \setminus PF_2 \setminus SF_2 \setminus RF_2 \quad \text{with} \\ PF_2 &= \{(x, y) \in F_2 \mid \exists b \in PB_{P_2} [(x, y) \rightsquigarrow_{P_2} b]\} \\ SF_2 &= \{(x, y) \in F_2 \mid \exists b \in SB_{P_2} [b \rightsquigarrow_{P_2} (x, y)]\} \\ RF_2 &= ((B_{P_2} \times F_2) \cap +_{P_2}) \setminus ((B_{P_2} \times F_2) \cap ||_{P_2}) \end{aligned}$$

C. Inter-Boundary Nodes Reduction

The second kind of reduction exploits the behavioural profile for aligned nodes that are in strict order with preceding and succeeding boundary nodes in source model (the so called *inter-boundary nodes*), but not with the change node. For illustration purposes, again, we assume the model in Figure 3 to be the source model. Further on, we consider the model in Figure 4(b) to be the target model. Note that this model equals the part of the original target model that is covered by the change region derived via boundary node reduction. We see that the source model contains an aligned activity node, namely F , that is *between* the boundary nodes in terms of strict order (i.e., it is an inter-boundary node), but which is exclusive to the change. In order to enforce this exclusive relation between the corresponding node F

and the change region in the target model, all flow arcs that are not exclusive to F are removed from the change region. This yields a change region, which is highlighted in grey in Figure 4(b). This example illustrates the reduction based on inter-boundary nodes for the case of an exclusive relation. It can equally be applied for inter-boundary nodes that are observation concurrent to the change in the source model.

With respect to our formal model, we define inter-boundary nodes and the respective reduction as follows.

Definition 11 (Inter-Boundary Nodes): Let $P_1 = (A_1, I_1, C_1, F_1, T_1)$ be the source process model, P_2 a target process model, and \sim a correspondence relation for P_1 and P_2 . The set of *inter-boundary nodes* IB_{P_1} contains all aligned activity nodes that precede and succeed the respective boundary nodes in strict order, i.e., $IB_{P_1} = \{a \in A_1 \mid \forall b_p \in PB_{P_2}, b_s \in SB_{P_2} [b_p \rightsquigarrow_{P_1} a \rightsquigarrow_{P_1} b_s]\}$.

Definition 12 (Inter-Boundary Node Reduction): Let P_1 be the source process model, $P_2 = (A_2, I_2, C_2, F_2, T_2)$ the target process model, and \sim a correspondence relation for P_1 and P_2 . The *inter-boundary node reduction* creates a change region \mathcal{C}_{P_2} , such that

$$\begin{aligned} \mathcal{C}_{P_2} &= F_2 \setminus EF_2 \setminus CF_2 \quad \text{with} \\ EF_2 &= \{(x, y) \in F_2 \mid \exists b \in B_{P_2} [(x, y) +_{P_2} b]\} \\ CF_2 &= \{(x, y) \in F_2 \mid \exists b \in B_{P_2} [(x, y) ||_{P_2} b]\} \end{aligned}$$

D. Determining the Change Region

After we discussed the elementary steps, the whole algorithm to derive the change region is shown in Algorithm 1. We see that both reductions potentially narrow the set of flow arcs of the target model, such that the actual change region is the intersection of their results. Although both reductions

Algorithm 1: Determining the change region

Input: $P_1 = (A_1, I_1, C_1, F_1, T_1)$, the source process model
 $P_2 = (A_2, I_2, C_2, F_2, T_2)$, the target process model
 \sim , the correspondence relation between P_1 and P_2
 X , the change node in P_1

Output: C_{P_2} , the change region

```
/* Boundary Node Reduction */
PBP1 := GetPrecedingBoundaryNodes(P1, ~, X);
SBP1 := GetSucceedingBoundaryNodes(P1, ~, X);
R1 := BoundaryNodeReduction(P2, PBP1, SBP1);

/* Inter-Boundary Node Reduction */
IBP1 := GetInterBoundaryNodes(P1, ~, PBP1, SBP1);
R2 := InterBoundaryNodeReduction(P2, IBP1);

/* Deriving the result */
CP2 := R1 ∩ R2
```

are conceptually independent of each other, the result of one reduction might also be used as the input for the second reduction. In fact, this procedure has been illustrated by the examples given in Figure 4(a) and Figure 4(b), as we treated the change region in the former model as the input for the second reduction step. Albeit potentially overlapping, both reductions, however, are not redundant. The final change region in Figure 4(b) cannot be derived by applying only the inter-boundary node reduction. For instance, the flow arc (6, 7) of the original target model is only removed by the boundary node reduction.

The result of our algorithm to determine the change region might either be a set of flow arcs of the target model or an empty set. The former indicates that there are already flow arcs in the model that fulfil the requirements of the behavioural profile with respect to the potential change. Therefore, all these flow arcs qualify for representing the change of the source model in the target model. Note that these flow arcs are not necessarily part of a connected subgraph of the process model. Instead, there might be multiple areas in the target model that qualify for the realisation of the change. In case the set is empty, the target model does not yet contain a flow arc satisfying the behavioural requirements. However, it is always possible to insert flow arcs according behavioural requirements, due to our requirement of a consistent model alignment (cf., Definition 7). In this case, the boundary nodes and inter-boundary nodes guide the adaptation of the target process model. The former impose requirements regarding the strict order relation, whereas the exclusiveness and observation concurrency relation have to be considered for the latter.

V. EVALUATION OF THE APPROACH

This section elaborates on the applicability of our approach by reviewing the initial example, presenting an implementation, and discussing the computation complexity.

Initial Example & Tool Support: Application of our approach to the initial example depicted in Figure 1 yields

the following result. For the change in the lower model (B), i.e., insertion of activity *Sanity Check*, we derive a change region containing solely one flow arc in the upper model (A). That is, a corresponding activity might have to be inserted between the XOR gateway and the activity *Close Deal*. For the change in the upper model (A), i.e., the insertion of a fragment realising the black list handling, the change region in the lower model (B) contains two flow arcs, preceding and succeeding activity *Get Reports of Similar Customers*.

We implemented our approach prototypically. Currently, we are integrating it into Oryx¹, an open-source modelling framework. Given two BPMN process models, a definition of the correspondence relation, and one activity marked as the change node, the change region is highlighted in the corresponding model as illustrated in Figure 5. Currently, our implementation is restricted to BPMN process models that can be mapped to free-choice Petri-nets, as we leveraged an existing mapping from BPMN to Petri-net in order to determine the behavioural profiles of the process models.

Complexity Considerations: To estimate the complexity of our approach, we first consider the identification of boundary nodes and inter-boundary nodes. Starting from the change node, we determine all aligned nodes preceding or succeeding the change in strict order and identify boundary nodes. After that, we check each aligned node whether it is an inter-boundary node. Here, it suffices to check exclusiveness or observation concurrency to the change node, and strict order with one preceding and one succeeding boundary node. Both these computations can be performed in low polynomial time since they essentially build on search techniques.

More costly is the derivation of the behavioural profiles from the source and target model. Calculating the profiles from the reachability graph is not an efficient option. The reachability problem, even in live and safe free-choice Petri nets, is known to be NP-complete in the general case, even though for important subclasses, such as cyclic extended free-choice nets, it is polynomial [13]. Fortunately, there are several other efficient techniques. The calculation of a strict concurrency relation on free-choice Petri nets (a subset of our concurrency relation) can be performed in $O(n^3)$ [14]. We are currently investigating this and other techniques in order to find a low complexity class for the profile calculation.

VI. RELATED WORK

The work presented in this paper can be related to three major areas of research, namely process adaptation, process model similarity, and schema matching.

Process adaptation deals with correctness issues in the situation when a process model becomes obsolete and has to be replaced by a new model. This change on the schema level requires the migration of process instances in a process engine. The notion of trace equivalence places an important

¹<http://www.oryx-project.org>

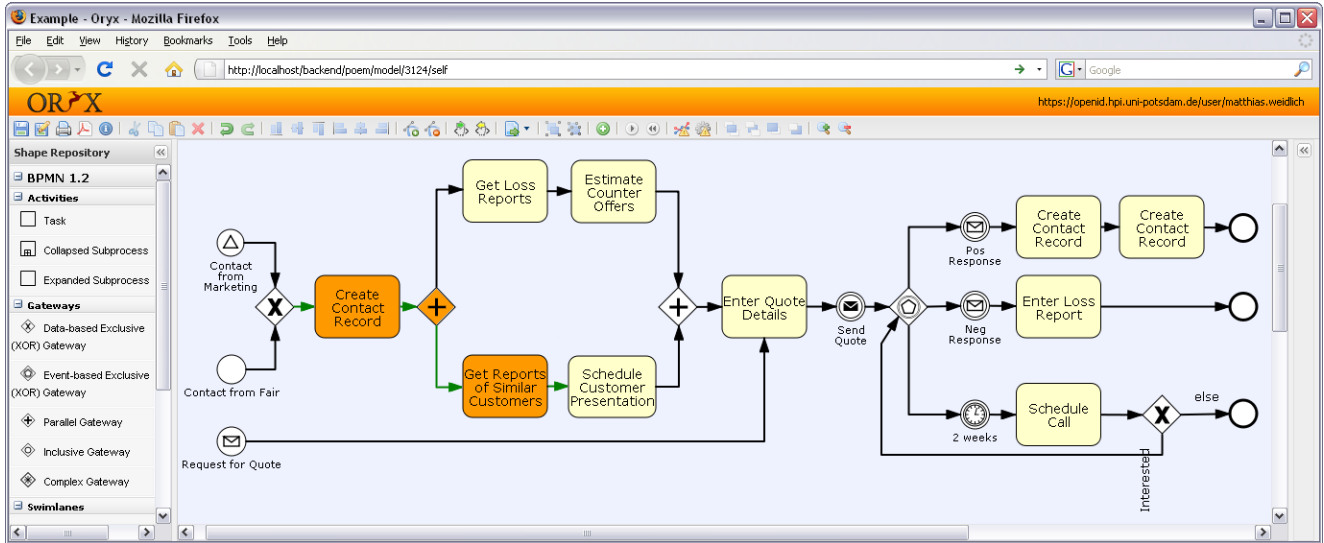


Figure 5. The change region for the change illustrated in Figure 1 is highlighted

role in this context for determining whether an instance can be migrated to the new schema without modifications. Different correctness criteria are nicely summarized in [15]. The concept of a behavioural profile and the notion of consistency are less strict than trace equivalence, but provide straight-forward information where a change propagation is allowed. Similar relations, but not exactly the same as used in behavioural profiles, are used in process mining as input for constructing process models [16].

The similarity of process models closely relates to different notions of behavioral equivalence such as trace equivalence and bisimulation. Since these notions yield a true or false answer, and can therefore not directly be applied if models overlap. Our notion of consistency that is based on behavioural profiles can be regarded as a relaxed version of trace equivalence that can also handle overlapping models. A good overview of various equivalence notions is presented in [17]. The strict true or false nature of these notions has been criticized in [18] who propose a similarity measure for process mining. Other work has used causal footprints as a behavioural abstraction for determining the degree of similarity between process models [19], [20].

For our approach we assume that correspondences of two process models have been identified and captured as so-called links. The research area of schema integration, and in particular, schema matching investigates how such correspondences can be identified automatically. A good overview is provided in [7]. It also has to be mentioned that our notion of consistency can be related to desirable properties of schema mappings. For instance, the notion of answerability as discussed in [21] can be interpreted as a reformulation of process model consistency for the domain of data schemas. For business process models, it is almost

of equivalent importance to detect mismatches. The work presented in [22], [23] provides a systematic framework of diagnosis and resolution of such mismatches.

VII. CONCLUSION

In this paper we discussed the problem of change propagation between related process models at different levels of abstraction serving different purposes. Non-hierarchical refinement is the major conceptual challenge for any approach to propagate changes between such models. Our contribution in this area is a formal approach to change propagation based on the concept of behavioural profiles. We use the behavioural profiles of the source model (where the change is introduced) and of the target model along with the correspondences between them, to determine the change region. The change region includes all flow arcs where the change can be propagated in the target model without violating the consistency of the models.

In future research, we aim at extending our approach with data related aspects by taking data access relations into account. That, in turn, might qualify our approach of change propagation for data-centric process models. Such models can be observed, for instance, in the area of scientific workflows. Further on, we plan to investigate how behavioural profiles can be applied in other use cases such as validation and process integration.

REFERENCES

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

- [2] 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.
- [3] 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.
- [4] J. B. Hill, M. Cantara, M. Kerremans, and D. C. Plummer, "Magic Quadrant for Business Process Management Suites," *Gartner Research*, vol. G00164485, February 2009.
- [5] Logica Management Consulting, "Securing the Value of Business Process Change," Whitepaper, October 2008.
- [6] OMG, *Business Process Modeling Notation (BPMN) 1.2*, January 2009.
- [7] E. Rahm and P. A. Bernstein, "A Survey of Approaches to Automatic Schema Matching," *VLDB Journal*, vol. 10, no. 4, pp. 334–350, 2001.
- [8] 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.
- [9] 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.
- [10] R. Eshuis and R. Wieringa, "Tool Support for Verifying UML Activity Diagrams," *IEEE Trans. Software Eng.*, vol. 30, no. 7, pp. 437–447, 2004.
- [11] 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.
- [12] B. Weber, S. Rinderle, and M. Reichert, "Change Patterns and Change Support Features in Process-Aware Information Systems," in *CAiSE*, ser. Lecture Notes in Computer Science, J. Krogstie, A. L. Opdahl, and G. Sindre, Eds., vol. 4495. Springer, 2007, pp. 574–588.
- [13] J. Esparza, "Reachability in Live and Safe Free-Choice Petri Nets is NP-Complete," *Theor. Comput. Sci.*, vol. 198, no. 1-2, pp. 211–224, 1998.
- [14] 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, 1996, pp. 1–6.
- [15] S. Rinderle, M. Reichert, and P. Dadam, "Correctness Criteria for Dynamic Changes in Workflow Systems - A Survey," *Data Knowl. Eng.*, vol. 50, no. 1, pp. 9–34, 2004.
- [16] W. Aalst, A. Weijters, and L. Maruster, "Workflow Mining: Discovering Process Models from Event Logs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 9, pp. 1128–1142, 2004.
- [17] 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.
- [18] 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.
- [19] 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.
- [20] B. F. van Dongen, J. Mendling, and W. M. P. van der Aalst, "Structural Patterns for Soundness of Business Process Models," in *Tenth IEEE International Enterprise Distributed Object Computing Conference (EDOC 2006), 16-20 October 2006, Hong Kong, China*. IEEE Computer Society, 2006, pp. 116–128.
- [21] 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.
- [22] 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.
- [23] 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.
- [24] 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.