A Formal Framework for Data-Aware Process Interaction Models

David Knuplesch, Rüdiger Pryss, Manfred Reichert
A Formal Framework for
Data-Aware Process Interaction Models

David Knuplesch and Rüdiger Pryss and Manfred Reichert
Institute of Databases and Information Systems
Ulm University, Germany
Email: {david.knuplesch,ruediger.pryss,manfred.reichert}@uni-ulm.de

Abstract
IT support for distributed and collaborative workflows as well as related interactions between business partners are becoming increasingly important. For modeling such partner interactions as flow of message exchanges, different top-down approaches, covered under the term interaction modeling, are provided. Like for workflow models, correctness constitutes a fundamental challenge for interaction models; e.g., to ensure the boundedness and absence of deadlocks and lifelocks. Due to their distributed execution, in addition, interaction models should be message-deterministic and realizable, i.e., the same conversation (i.e. sequence of messages) should always lead to the same result, and it should be ensured that partners always have enough information about the messages they must or may send in a given context. So far, most existing approaches have addressed correctness of interaction models without explicitly considering the data exchanged through messages and used for routing decisions. However, data support is crucial for collaborative workflows and interaction models respectively. This technical report enriches interaction models with the data perspective. In particular, it defines the behavior of data-aware interaction models based on Data-Aware Interaction Nets, which use elements of both Interaction Petri Nets and Workflow Nets with Data. Finally, formal correctness criteria for Data-Aware Interaction Nets are derived, guaranteeing the boundedness and absence of deadlocks and lifelocks, and ensuring message-determinism as well as realizability.

Index Terms

I. INTRODUCTION
Workflow management is of utmost importance for companies that want to efficiently handle their workflows as well as their interactions with partners and customers [1]. Despite the varying issues relevant for the IT support of distributed and collaborative workflows [2], common aspects to be considered include the support of appropriate modeling techniques as well as the definition of a formal execution semantics, ensuring proper and correct partner interactions (i.e., message exchanges).

Workflow management methods and techniques tackling these challenges consider a choreography as a specification of message exchanges between the partners of a collaborative workflow. Respective approaches provide a global view on distributed workflows and support partners in correctly defining their private processes (partner processes for short). The latter can be transformed into distributed, executable workflows. When executing these workflows, their interplay is coordinated in terms of a conversation (i.e., a sequence of exchanged messages) that follows the global behavior specified by the choreography.

Currently, there exist two different paradigms for modeling choreographies: interconnection modeling and interaction modeling. Interconnection modeling uses message exchange as link between partner processes or public views on them. In particular, this paradigm does not allow modeling the message exchange separately from the partner processes. Hence, it is considered as a bottom-up approach. Approaches enabling interconnection modeling include BPMN Collaboration Diagram [3], BPEL4Chor [4], and Compositions of Open Nets [5]. By contrast, interaction modeling provides a top-down approach. An Interaction Model specifies the flow of message exchanges
without having any knowledge about the partner processes. Moreover, the models of the partner processes are created taking the interaction model into account. Nevertheless, common interaction models use the same patterns as workflow models (e.g. parallel and conditional branchings), but instead of tasks they refer to the messages exchanged. Approaches enabling interaction modeling include iBPMN [6], BPMN Choreography Diagrams [3], Service Interaction Patterns [7], and WSCDL [8].

This technical report focuses on the correctness of interaction models. Related issues discussed in the literature include boundedness and absence of deadlocks and lifelocks, as well as the realizability of interaction models [5], [9]–[11]. Realizability postulates that partners always can compute which messages they must or may send in a given execution context. Fig. 1 (1) outlines a simple example of a non-realizable choreography with four partners $A, B, C,$ and $D$, and two messages $m_1$ and $m_2$. This interaction model specifies that after sending message $m_1$ from $A$ to $B$, message $m_2$ must be sent from $C$ to $D$. Obviously, only $A$ or $B$ knows when $C$ must send message $m_2$, but $C$ does not have this knowledge. Consequently, this interaction model is not realizable. A necessary precondition for realizability is message-deterministic behavior, i.e. the same conversation (i.e. sequence of messages) should always lead to the same result. An example of an interaction model, which is not message-deterministic, is shown in Fig. 1 (2); this interaction model comprises partners $A, B,$ and $C$, and messages $m_1, m_2, m_3,$ and $m_4$. After sending the first message $m_1$, either the upper or the bottom branch shall be chosen. In any case, the next message $m_2$ must be sent from $B$ to $C$. Depending on the branch chosen, however, then $C$ either must send $m_3$ to $B$ or $m_4$ to $A$. From the perspective of $C$, it cannot be determined, which of the two interpretations shall be applied. By contrast, $B$ knows the chosen branch (e.g., the upper one). Hence, $C$ might send $m_4$ to $A$, while $B$ waits for $m_3$, or vice versa.

A property similar to realizability is clear termination. It requires that a partner always can compute, whether he will be sender or receiver of any messages in the sequel. An example of an interaction model, which is not clearly terminating, is shown in Fig. 1 (3). This interaction model comprises partners $A, B,$ and $C$, and messages $m_1, m_2, m_3,$ and $m_4$. After sending the first message $m_1$ from $A$ to $B$, $B$ can either send message $m_2$ to $A$ or message $m_4$ to $C$. When choosing the first option (i.e. $B$ sends $m_2$ to $A$), $A$ must send $m_3$ to $C$ afterwards. In turn, when choosing the second option (i.e. $B$ sends $m_4$ to $C$), the execution is terminated, although $A$ may still wait for the arrival of message $m_2$. Note, that from the perspective of $A$ nothing has changed since $m_1$ was sent. 

Existing approaches for interaction modeling do not adequately support the data perspective. Either related execution semantics completely ignore the data perspective or there is a lack of appropriate correctness criteria, especially if routing decisions are based on message data. 

This technical report deals with fundamental correctness issues when making interaction models data-aware. Section II provides an example from the healthcare domain to emphasize the need of data-awareness in interaction models. Section II further discusses the challenges to be tackled when considering the data perspective. Section III then introduces our formal framework for data-aware interaction modeling. First, an interaction meta-model is provided in terms of the Data-Aware Choreography (DAChor). The behavior of a DAChor is described by a transformation to Data-Aware Interaction Nets (DAI Nets). These combine Interaction Petri Nets [9] and Workflow

![Fig. 1. Violating realizability, message-determinism, clear termination [9]](image_url)
Nets with Data [12]. Based on Data-Aware Interaction Nets, the set of allowed conversations (i.e., message exchanges) is derived and used to introduce formal correctness criteria for DAI Nets and DAChor respectively. These criteria guarantee for the boundedness and absence of deadlocks and lifelocks, and ensure message-determinism, realizability, and clear termination. Section IV discusses related work and Section V concludes with a summary and outlook.

II. EXAMPLE, CHALLENGES, CONTRIBUTION

This section introduces a simplified real-world scenario, which we elaborated in the context of case studies conducted in the healthcare domain. These case studies highlighted the relevance of the data perspective in interaction models. Thus, the scenario we select emphasizes the challenges arising from the support of data-awareness in interaction models. It describes the transport of a patient to and from a unit performing a Positron Emission Tomography (PET) scan. A PET scan is a kind of nuclear medicine imaging not performed by the respective hospital itself in our scenario. Thus, if a PET scan is ordered for a patient, patient transportation to the respective provider is required. In this context, the hospital must inform the provider of the PET scan about the patient’s status, such that he can decide on the preparations required. Furthermore, we require a patient to be examined just before the transport to exclude potential risks (e.g., the patient being in a critical condition).

The scenario involves three partners, i.e., the Hospital responsible for the patient and ordering the PET scan, the Transportation (Transp.) Provider transporting the patient, and the PET provider performing the PET scan. The interaction starts with the Hospital requesting the PET scan (Request PET). In the context of this request, the Hospital informs the PET Provider about the status of the patient. In turn, the PET provider confirms the time for which the scan is scheduled (Confirm), and then requests the Transp. Provider to perform the transport (Request Transp.).

- If the patient is in a critical condition, the Transp. Provider requests the Hospital to examine him to check whether he is transportable (Request Exam.). Based on the Result of this examination, the Hospital informs the Transp. Provider on whether to continue or abort the interaction.
- If the interaction is continued or the patient is not in a critical condition, Transp. Provider informs the PET provider after picking up the patient and arriving at the PET unit (Arrival). After the PET scan is performed, the PET provider requests retransport of the Transp. Provider (Retransport).

Finally, the Transp. Provider informs the Hospital about the return of the patient (Return).

Obviously, properly modeling the interactions of this scenario requires support for routing decisions based on the data of the messages exchanged. More precisely, in the given scenario, there is a decision referring to data of the first message exchanged (i.e. whether or not the patient is in a critical condition). Another decision refers to the message sent by the hospital and indicating whether the request shall be canceled. Hence, we use a notation based on BPMN 2.0 [3] and iBPMN [9], but enrich it with so-called virtual data objects. We denote this notation as Data-Aware Choreography (DAChor) and use it to model our scenario in Fig. 2. Virtual data objects have a data domain and can be used as variables when defining conditions for routing decisions. However, these virtual data objects are not used for modeling information flow. Thus, the data assignment relation denotes which data of an interaction is assigned to any virtual data object. Note that such a data assignment relation can only lead from an interaction to a virtual data object, but not vice versa. Furthermore, an interaction is assigned to a message class denoting the message type. From the message class, the sender, receivers, and data domain are inherited (e.g., boolean). Finally, when executing a choreography, messages of the related message class correspond to interactions.

Having a closer look at our scenario, one can recognize that it neither ensures realizability nor clear termination. If the Hospital requests canceling the PET scan, the PET provider is not informed accordingly and hence may still wait for the message; i.e., no clear termination is ensured. However, if Alternative 2 (cf. Fig. 2) is applied, the PET provider will be informed and clear termination can be ensured. In turn, realizability is violated for the given interaction model, since Transp. Provider does not know whether the patient is in a critical condition. Thus, Transp. Provider cannot determine whether an examination must be requested. To ensure realizability, it is not sufficient to only check whether this information was directly sent to Transp. Provider. Consider
Alternative 1, which ensures realizability by also sending the confirmation to Transp. Provider, if the patient is in a critical condition. Obviously, implicit knowledge of Transp. Provider about the value of virtual data object Status is sufficient to ensure realizability. This makes the definition of proper correctness criteria for data-aware interaction models Section III very challenging.

Before defining correctness criteria for DACHors, their behavior has to be formalized. In [9], Decker et al. define the behavior of iBPMN choreographies based on their transformation to Interaction Petri Nets (IP Nets). However, IP Nets are unaware of data. This raises the challenge to first enrich IP Nets as well as their behavior with data, i.e., to design Data-Aware Interaction Nets (DAI Nets). Following this, an appropriate transformation is presented.

The main contribution of this technical report is to introduce a formal framework for data-aware interaction models putting emphasis on correctness. Especially, this framework comprises specific correctness criteria for interaction models (e.g. realizability, clear termination). Note, that the latter exceed traditional correctness and soundness criteria that are known from workflows and interconnection models [5], [13], [14]. Further contributions include the introduction of DACHors and DAI Nets as well as the transformation from DACHors to DAI Nets with well defined behavior.

III. Formal Framework

This section introduces our formal framework for ensuring correctness of data-aware interaction models. First, the scope of an interaction model is described as interaction domain and in terms of messages (cf. Def. 1 and 2 in Section III-A). Second, Data-Aware Choreographies (DACHors) are introduced as formal meta-model for data-aware interaction modeling (cf. Def. 3 in Section III-B). In Section III-D, the semantics of DACHors is described based on their transformation to Data-Aware Interaction Nets (DAI Nets). DAI Nets combine Interaction Petri Nets (IP Nets) [9] and Workflow Nets with Data (WFD Nets) [12] (cf. Def. 5 in Section III-C). Their behavior is described in terms of markings and execution traces (cf. Def. 8–10 in Section III-E). Def. 12 introduces conversations representing the observable parts of an execution trace (i.e., exchanged messages). Finally, partner views are defined (cf. Def. 14). Based on traces, conversations, and views, we then introduce correctness criteria for DAI Nets and DACHors respectively (cf. Def. 11, 13, and 15). Fig. 3 provides an overview of the main elements of our formal framework and their relations.
A. Interaction Domains and Messages

This section defines the basic elements of data-aware interaction modeling in terms of an *interaction domain*. The latter contains *roles* to differentiate the partners as well as *message classes* and related *data domains*. Furthermore, the notion of *message* (cf. Def. 1 and 2 and Example 1).

**Definition 1** (Interaction Domain).

An interaction domain is a tuple \( I = (\mathcal{R}, \mathcal{D}, C, \text{dom}_C, s_C, r_C, \epsilon) \), with

- \( \mathcal{R} \) is a set of roles,
- \( \mathcal{D} \) is a set of data domains; each \( D \in \mathcal{D} \) represents a finite set of values,
- \( C \) is a set of message classes,
- \( \text{dom}_C : C \rightarrow \mathcal{D} \) is a function assigning to each message class a data domain,
- \( s_C : C \rightarrow \mathcal{R} \) assigns the sender to each message class,
- \( r_C : C \rightarrow 2^\mathcal{R} \) assigns the set of receivers to each message class,
- \( \epsilon \) is the empty value.

Further, we define \( \Omega_I := \{\epsilon\} \cup \bigcup_{D \in \mathcal{D}} D \) as the set of all values.
Furthermore, we introduce several sets of messages.

Definition 2 (Messages).
Let $\mathcal{I} = (\mathcal{R}, \mathcal{D}, \mathcal{C}, \text{dom}_C, \text{sc}, \text{rc}, \epsilon)$ be an interaction domain. Then: A message in $\mathcal{I}$ is a tuple $\mu = (c, x) \in C \times \Omega_\mathcal{I}$, with
- $c \in C$ is the corresponding message class, and
- $x \in \text{dom}_\mathcal{C}(c)$ is the message content transferred.

Furthermore, we define:
- $\Sigma_c := \{(c', x) \in C \times \Omega_\mathcal{I} \mid c' = c \land x \in \text{dom}_\mathcal{C}(c')\}$ as set of all messages corresponding to message class $c \in C$,
- $\Sigma_\mathcal{I} := \bigcup_{c \in C} \Sigma_c$ as set of all messages corresponding to interaction domain $\mathcal{I}$,
- $\Sigma_{\mathcal{R}_R} := \{(c, v) \in \Sigma_\mathcal{I} \mid \text{dom}_\mathcal{C}(c) = R\}$ as set of all messages sent by role $R \in \mathcal{R}$,
- $\Sigma_{\mathcal{R}_R} := \Sigma_{\mathcal{R}_R} \cup \Sigma_{\mathcal{R}_R}$ as set of all messages corresponding to role $R$, i.e. sent or received by $R$.

Example 1 (Basic Notions).
Consider the interaction model of the patient transportation scenario from Fig. 2. Its interaction domain is $\mathcal{I} = (\mathcal{R}, \mathcal{D}, \mathcal{C}, \text{dom}_C, \text{sc}, \text{rc}, \epsilon)$ with:

$\mathcal{R} = \{\text{Hospital, PET Provider, Transp. Provider}\}$
$\mathcal{D} = \{D_1 = \{\}, D_{\text{status}} = \{\text{uncritical, critical}\}, D_{\text{Order}} = \{\text{abort, continue}\}, D_{\text{Date}} = \{1.1.1900, \ldots, 31.12.2099\}\}$
$\mathcal{C} = \{\text{Request PET, Confirmation, Request Trans., Request Exam., Result, Arrival, Retransport, Return, Confirmation+, Cancel PET}\}$

$\text{sc}(\text{Request PET}) = \text{Hospital}$
$\text{sc}(\text{Confirmation}) = \text{PET provider}$
$\text{sc}(\text{Request Trans.}) = \text{PET provider}$
$\text{sc}(\text{Result}) = \text{Hospital}$
$\text{sc}(\text{Arrival}) = \text{Transp. Provider}$
$\text{sc}(\text{Retransport}) = \text{PET provider}$
$\text{sc}(\text{Confirmation+}) = \text{PET provider}$
$\text{sc}(\text{Cancel PET}) = \text{Transp. Provider}$
$\text{rc}(\text{Request PET}) = \{\text{PET provider}\}$
$\text{rc}(\text{Confirmation}) = \{\text{Hospital}\}$
$\text{rc}(\text{Request Trans.}) = \{\text{Transp. Provider}\}$
$\text{rc}(\text{Result}) = \{\text{Transp. Provider}\}$
$\text{rc}(\text{Arrival}) = \{\text{PET provider}\}$
$\text{rc}(\text{Retransport}) = \{\text{Transp. Provider}\}$
$\text{rc}(\text{Confirmation+}) = \{\text{Hospital, Transp. Provider}\}$
$\text{rc}(\text{Cancel PET}) = \{\text{PET provider}\}$

$\text{dom}_\mathcal{C}(\text{Request PET}) = D_{\text{Status}}$
$\text{dom}_\mathcal{C}(\text{Confirmation}) = D_{\text{Date}}$
$\text{dom}_\mathcal{C}(\text{Request Trans.}) = D_{\text{Date}}$
$\text{dom}_\mathcal{C}(\text{Result}) = D_\epsilon$
$\text{dom}_\mathcal{C}(\text{Retransport}) = D_\epsilon$
$\text{dom}_\mathcal{C}(\text{Confirmation+}) = D_{\text{Date}}$

$\Sigma_\mathcal{I} = \{(\text{Request PET, uncritical}), (\text{Request PET, critical}), (\text{Result, abort}), (\text{Result, continue}), (\text{Request Exam., } c), (\text{Arrival, } c), (\text{Confirmation, } 1.1.1900), \ldots, (\text{Confirmation, } 31.12.2099), (\text{Confirmation+}, 1.1.1900), \ldots, (\text{Confirmation+}, 31.12.2099), (\text{Retransport, } c), (\text{Return, } c), (\text{Request Trans., } 1.1.1900), \ldots, (\text{Request Trans., } 31.12.2099), (\text{Cancel PET, } c)\}$
B. Data-Aware Choreography

Based on the interaction domain from Def. 1, we define the notion of data-aware choreography (DAChor). DAChor enriches BPMN choreography models with virtual data objects, a data assignment relation, and guards.

**Definition 3 (Data-Aware Choreography; DAChor).**
Let \( \mathcal{I} = (R, D, C, \text{dom}_C, s_C, r_C, \epsilon) \) be an interaction domain. Then: A Data-Aware Choreography (DAChor) over \( \mathcal{I} \) is a tuple \( \text{DAC} = (N, I, G, e_s, E_e, G_0^s, G_0^m, G_d^s, G_d^m, G_x^s, G_x^m, V, \text{class}, \rightarrow, \rightarrow^+, \text{dom}_V, \text{grd}) \), with

- \( N \) is the set of nodes being the disjoint conjunction of the set of interactions \( I \) and the set of gateways and events \( G \). In turn, the latter is the disjoint conjunction of the start event \( \{e_s\} \), the set of end events \( E_e \), the set of AND-splits \( G_s^a \), the set of AND-mergers \( G_m^a \), the set of data-based XOR-splits \( G_s^d \), the set of event-based XOR-splits \( G_m^d \), and the set of XOR-mergers \( G_x^a \).
- \( V \) is the set of virtual data objects.
- \( \text{class} : I \rightarrow C \) assigns a message class to each interaction,
- \( \rightarrow \subseteq (N - E_e) \times (N - \{e_s\}) \) is the interaction flow relation,
- \( \rightarrow^+ \subseteq \{e_s\} \times V \) is the data assignment relation,
- \( \text{dom}_V : V \rightarrow D \) is a function assigning a domain to each virtual data object,
- \( \text{grd} : (\rightarrow) \rightarrow \mathcal{G}_V \) is a function assigning a guard to each interaction flow.

The set of guards \( \mathcal{G}_V \) is defined as the set of propositional logic formulas over propositions of the form \( v = s \) or the form \( v \in \{s_1, s_2, \ldots, s_n\} \). Thereby, \( v \in V \) is a virtual data object and \( s, s_1, s_2, \ldots, s_n \in \text{dom}_V(v) \) are values of the related data domain. If a guard \( g \in \mathcal{G}_V \) uses a virtual data object \( v \in V \), we denote this as \( v \sim g \). Note that a guard can be constantly true. In this case, we omit it in the graphical representation of the DAChor (cf. Fig 2).

In the following, we introduce the well-formedness of DAChors. Example 2 then provides a formal description of our scenario from Fig. 2.

**Definition 4 (Well-Formed DAChor).**
A DAChor is well-formed, iff the following properties hold:
- the start event, each interaction, and each merge node have exactly one successor, i.e.,
  \[ \forall n \in \{e_s\} \cup I \cup G_m^s \cup G_m^m : \left| \{n' \in N | n \rightarrow n'\} \right| = 1 \]
- each split node has at least one successor, i.e.,
  \[ \forall g^s \in G_s^a \cup G_d^s \cup G_x^s : \left| \{n \in N | g^s \rightarrow n\} \right| \geq 1 \]
- each end event, each interaction, and each split node have exactly one predecessor, i.e.,
  \[ \forall n \in E_e \cup I \cup G_s^s \cup G_d^s \cup G_x^s : \left| \{n' \in N | n' \rightarrow n\} \right| = 1 \]
- each merge node has at least one predecessor, i.e.,
  \[ \forall g^m \in G_s^m \cup G_m^m : \left| \{n \in N | n \rightarrow g^m\} \right| \geq 1 \]
- each event-based XOR-split is solely followed by interactions, i.e.,
  \[ \forall g^s_{ex} \in G_s^s : \{n \in N | g^s_{ex} \rightarrow n\} \subseteq I \]
- guards of interaction flows are constantly true unless the source of an interaction flow is a data-based XOR-split, i.e.,
  \[ \text{grd}((n_1, n_2)) \neq \text{true} \iff n_1 \in G_d^s \]
- the data of an interaction is solely assigned to variables of the same data domain, i.e.,
  \[ \forall i \in I, \forall v \in V : i \rightarrow v \Rightarrow \text{dom}_C(\text{class}(i)) = \text{dom}_V(v) \]
- there is no cycle solely consisting of gateways, i.e.,
  \[ \not\exists g_0, g_1, \ldots, g_n \in G : g_0 \rightarrow g_1 \rightarrow \cdots \rightarrow g_n \rightarrow g_0 \]
Example 2 (DAChor).
Consider the scenario from Fig. 2. Basing its interaction domain $I$ (cf. Example 1) we can describe the given scenario as DAI Net $DAC = (N, I, G, e_s, E_c, G^a_x, G^m_x, G^a_d, G^m_d, G^a_{ex}, G^m_{ex}, V, class, \to, \to, dom_V, grd)$:

$I=\{i_1, \ldots, i_8\}, E_c=\{e_c, e_2\}, V=\{\text{Status, Order}\}$

$G^a_d = \{g_{d_x}^1, g_{d_x}^2\}, G^m_d = \{g_{m_d}^1, g_{m_d}^2\}, G^a_{ex} = G^m_{ex} = \emptyset$.

Further, we define

- $\text{class}(i_1) = \text{Request PET}$
- $\text{class}(i_2) = \text{Confirmation}$
- $\text{class}(i_3) = \text{Request Trans.}$
- $\text{class}(i_4) = \text{Exam.}$
- $\text{class}(i_5) = \text{Result}$
- $\text{class}(i_6) = \text{Arrival}$
- $\text{class}(i_7) = \text{Retransport}$
- $\text{class}(i_8) = \text{Return}$

$\text{dom}_V(\text{Status}) = D_{\text{status}}$
$\text{dom}_V(\text{Order}) = D_{\text{order}}$
$\text{dom}_V(\text{Request Exam.}) = \text{Status = critical}$
$\text{dom}_V(\text{Request PET}) = \text{Status = uncritical}$
$\text{dom}_V(\text{Order = abort})$
$\text{dom}_V(\text{Order = continue})$

C. Data-Aware Interaction Net

We introduce the notion of Data-Aware Interaction Net (DAI Net). It combines IP Nets [9] and WFD Nets [12]: Hence, the main elements of a DAI Net are places and transitions. To add data, these elements are enriched with variables and guards on transitions as known from WFD Nets. Furthermore, DAI Nets allow assigning message classes to transitions. Like in IP Nets, respective transitions are denoted as interaction transitions. Finally, all other transitions are called silent transitions.

Definition 5 (Data-Aware Interaction Net; DAI Net).
Let $I = (R, D, C, dom_C, s_C, t_C, e)$ be an interaction domain. Then, a Data-Aware Interaction Net (DAI Net) over $I$ is a tuple $\# = (P, p_{in}, P_o, P_f, T, T_S, T_I, V, class, \to, \to, dom_V, grd)$, where

- $P$ is the set of places; $P$ can be partitioned into the initial place $p_{in}$, the set of ordinary places $P_o$, and the set of final places $P_f$.
- $T$ is the set of transitions; $T$ can be partitioned into the sets of silent transitions $T_S$ and the set of interaction transitions $T_I$.
- $V$ is the set of variables.
- $\text{class}: T_I \to C$ is a function assigning a message class to each interaction transition,
- $\to \subseteq ((P - P_f) \times T) \cup (T \times (P - \{p_{in}\}))$ is the flow relation,
- $\to \subseteq T_I \times V$ is the data assignment relation. It expresses that the data of an interaction transition is assigned to the related variable,
- $\text{dom}_V: V \to D$ is a function assigning a data domain to each variable,
- $\text{grd}: T \to G_V$ is a function assigning a guard to each interaction flow relation.

Further, we define

- $\Sigma_{\#} := \bigcup_{t \in T_I} \Sigma_{\text{class}(t)}$ as the set of all messages corresponding to $\#$
- $P^t := \{p \in P | p \to t\}$ as the set of all places preceding $t$
- $P^t := \{p \in P | t \to p\}$ as the set of all places succeeding $t$
- $P^{\rightarrow t} := \{p \in P | \exists t \in P \to p\}$ as the set of the faraway places of $t$
As below Def. 3, the set of guards $\mathcal{G}_V$ is defined the set of propositional logic formulas over propositions of the form $v = s$ or the form $v \in \{s_1, s_2, \ldots, s_n\}$. Thereby, $v \in V$ is a variable and $s, s_1, s_2, \ldots, s_n \in \text{dom}_V(v)$ are values of the related data domain. If a guard $g \in \mathcal{G}_V$ uses a variable $v \in V$, we denote this as $v \sim g$. Note that a guard can be constantly true. In this case, we omit it in the graphical representation of the DAI Net (cf. Fig 4).

In the following, we introduce the well-formedness of DAI Nets. Then, we introduce a mapping from DACHors to DAI Nets and show that this mapping preserves the property of well-formedness.

**Definition 6 (Well-Formed DAI Net).**
A DAI Net is well-formed, iff the following properties hold:

- each transition has at least one preceding and one succeeding place, i.e.,
  $\forall t \in T : \exists p_1, p_2 \in P : p_1 \rightarrow t \rightarrow p_2$
- the content of an interaction transition is solely assigned to variables of the same data domain, i.e.,
  $\forall t_i \in T_i, \forall v \in V : t_i \rightarrow v \Rightarrow \text{dom}_V(\text{class}(t_i)) = \text{dom}_V(v)$
- there exists no cycle solely consisting of places and silent transitions, i.e.,
  $\exists p_0, p_1, \ldots, p_n \in P, t_0, t_1, \ldots, t_n \in T_S : p_0 \rightarrow t_0 \rightarrow p_1 \rightarrow t_1 \rightarrow \cdots \rightarrow p_n \rightarrow t_n \rightarrow p_0$.

**D. Mapping DACHors to DAI Nets**

In Section III-C, we introduced DAI Nets to define the behavior of DACHors. Based on this we can now define a mapping from data-aware choreographies to DAI Nets. This mapping is based on the approach proposed by Decker et al. [9] who define the behavior of iBPMN Choreographies through their transformation to IP Nets.

**Definition 7 (Mapping DACHors to DAI Nets).**
Let $\text{DAC} = (N, I, G, e_s, E_v, G^s_\exists, G^s_\lor, G^s_\land, G^s_\rightarrow, \rightarrow, \text{dom}_V, \text{grd})$ be a DACHor (cf. Def. 3). Then, $\text{DAC}$ can be mapped to a DAI Net defined as $\#: = (P, p_{in}, p_o, P_{fi}, T, T_S, T_I, \text{class'}, \rightarrow', \sim', \text{dom}_V, \text{grd}')$, with

- $P := \{(p_{in}, n_2) | n_1 \rightarrow n_2 \land n_1 \notin G^s_\exists \land n_2 \notin I \lor G^s_\lor \lor G^s_\land \land G^s_\rightarrow \land \text{dom}_V(n_1) = \text{dom}_V(n_2)\}$ interaction flow
  - $p_{in} := p_{(e_s, n)} \in P$, whereby $e_s \rightarrow n \in N$
  - $P_{fi} := \{(p_{(e_v, e)} | p \in P \land e_v \in E_v) \in P\}$ start events
  - $P_o := P - \{(p_{in}) \cup P_{fi}\}$ end events

- $T_s := \{t_g, g_x \in G^s_\lor \lor G^s_\land\}$ AND gateways
- $T^s_x := \{t_{(g_x)} | g_x \in G^s_\lor \land n \in N \land g_x \rightarrow n\}$ data-based XOR-split gateways
- $T^m_x := \{t_{(g^m_x)} | g^m_x \in G^m_\lor \land n \in N \land n \rightarrow g^m_x\}$ XOR-merge gateways
- $T_I := \{t_I \mid i \in I\}$, $T_S := T_s \cup T^s_x \cup T^m_x$, $T := T_S \cup T_I$ interactions

- $\text{class'}(t_i) := \text{class}(t)$ message class assignment

$\rightarrow' := \{(p_{(n_1, n_2)}, t_{n_2}) | n_1 \rightarrow n_2 \land n_1 \notin G^s_\lor \lor n_2 \notin I \lor G^s_\lor \lor G^s_\land \land G^s_\rightarrow \land \text{dom}_V(n_1) = \text{dom}_V(n_2)\}$ interactions/AND-gateways in

$\cup \{(t_{n_1}, P_{(n_1, n_2)}) | n_1 \rightarrow n_2 \land n_1 \in I \lor G^s_\lor \lor G^s_\land \land G^s_\rightarrow \land \text{dom}_V(n_1) = \text{dom}_V(n_2)\}$ interactions/AND-gateways out

$\cup \{(t^{\lor}_{n_1, n_2}), P_{(n_1, n_2)} | n_1 \rightarrow n_2 \land n_1 \in G^s_\lor \land n_2 \notin G^s_\lor \land G^s_\land \land G^s_\rightarrow \land \text{dom}_V(n_1) = \text{dom}_V(n_2)\}$ XORMerge-in

$\cup \{(t^{\land}_{n_1, n_2}), P_{(n_1, n_2)} | n_1 \rightarrow n_2 \land n_1 \in G^s_\land \land n_2 \notin G^s_\lor \land G^s_\land \land G^s_\rightarrow \land \text{dom}_V(n_1) = \text{dom}_V(n_2)\}$ XORMerge-out

$\cup \{(t^{\rightarrow}_{n_1, n_2}), P_{(n_1, n_2)} | n_1 \rightarrow n_2 \land n_1 \in G^s_\rightarrow \land n_2 \notin G^s_\lor \land G^s_\land \land G^s_\rightarrow \land \text{dom}_V(n_1) = \text{dom}_V(n_2)\}$ data-based XOR-split in

$\cup \{(t^{\rightarrow}_{n_1, n_2}), P_{(n_1, n_2)} | n_1 \rightarrow n_2 \land n_1 \in G^s_\rightarrow \land n_2 \notin G^s_\lor \land G^s_\land \land G^s_\rightarrow \land \text{dom}_V(n_1) = \text{dom}_V(n_2)\}$ data-based XOR-split out

$\cup \{(t^{\rightarrow}_{n_1, n_2}), P_{(n_1, n_2)} | n_1 \rightarrow n_2 \land n_1 \in G^s_\rightarrow \land n_2 \notin G^s_\lor \land G^s_\land \land G^s_\rightarrow \land \text{dom}_V(n_1) = \text{dom}_V(n_2)\}$ event-based XOR-split

$\cup \{(t^{\rightarrow}_{n_1, n_2}), P_{(n_1, n_2)} | n_1 \rightarrow n_2 \land n_1 \in G^s_\lor \land n_2 \notin G^s_\lor \land G^s_\land \land G^s_\rightarrow \land \text{dom}_V(n_1) = \text{dom}_V(n_2)\}$ data assignment relation

$\sim' := \{(t, v) | (t, v) \in \rightarrow\}$ guard assignment

$\text{grd}'(t) := \{\text{grd}((g^s_x, n)) \mid t'_{(g^s_x, n)} \in T^s_x\}$ guard assignment

Theorem 1 states that the mapping from DACHors to DAI Nets preserves well-formedness. The application to our example is shown in Example 3 and Fig 4.
Theorem 1 (Preservation of Well-Formedness).
Let DAC be a DAC hor that is mapped to a DAI Net #. If DAC is well-formed, # is well-formed as well.

We now prove Theorem 1. Our proof consists of three parts that correspond to the three properties of well-formedness for DAI Nets. First, we prove the first property, i.e., each transition has at least one preceding and one succeeding place:

Proof 1 (Preservation of Well-Formedness (Property 1)).
Let DAC = (N, I, G, e, E, G_s, G^m_s, G^m_s, G^m, G_s, G^m, V, class, →, →, dom_V, dom_G) be a well-formed DAC hor. DAC is mapped to the DAI Net #. The latter is defined as # := (P, p_in, P_o, P_f, T, T_S, T_I, V, class', →', →', dom_V, dom_G').

We now prove Theorem 1. Our proof consists of three parts that correspond to the three properties of well-formedness for DAI Nets. First, we prove the first property, i.e., each transition has at least one preceding and one succeeding place:

Proof 2 (Preservation of Well-Formedness (Property 2)).
Let DAC = (N, I, G, e, E, G_s, G^m_s, G^m_s, G^m, G_s, G^m, V, class, →, →, dom_V, dom_G) be a well-formed DAC hor. DAC is mapped to a DAI Net #. The latter is defined as # := (P, p_in, P_o, P_f, T, T_S, T_I, V, class', →', →', dom_V, dom_G').

Thus, a transition has at least one preceding and one succeeding place, consequently the first property holds. □

Second, we prove that the data of an interaction transition is solely assigned to variables of the same data domain:

Proof 2 (Preservation of Well-Formedness (Property 2)).
Let DAC = (N, I, G, e, E, G_s, G^m_s, G^m_s, G^m, G_s, G^m, V, class, →, →, dom_V, dom_G) be a well-formed DAC hor. DAC is mapped to a DAI Net #. The latter is defined as # := (P, p_in, P_o, P_f, T, T_S, T_I, V, class', →', →', dom_V, dom_G').

Let t_i ∈ T_I be an interaction transition and v ∈ V a virtual data object of #. Then:

(t_i, v) ∈ →' ⇒ (i, v) ∈ →

⇒ dom_C(class(t_i)) = dom_C(class(i)) = dom_V(v)

Thus, data assignments are correct, consequently the second property holds. □
Finally, we prove the third property, i.e., there exists no cycle solely consisting of places and silent transitions. For this purpose, we define a function \( \text{gate} \), which assigns to each silent transition \( \# \) a gateway in DAC. Furthermore, we show that the gates of two silent transitions are connected if the silent transitions are connected by a place in Lemma 1. trans

**Definition (Gate of a Silent Transition).**
Let DAC = \( (N,I,G,e_\text{s},E_\text{e},G^m_\text{s},G^m_\text{dx},G^m_\text{ex},G^m_\text{x},V,\text{class},\rightarrow,\rightarrow,\rightarrow,\rightarrow,\rightarrow,\text{dom}_V,\text{grd}) \) be a well-formed DAChor. DAC is mapped to a DAi Net \(# = (P,p(T_i,P_j,T,S,T,F,T,S,T,\text{class},\rightarrow,\rightarrow,\rightarrow,\text{dom}_V,\text{grd}) \). Then:

The function \( \text{gate} : T_S \rightarrow (G - G^m_\text{s}) : t_g \rightarrow \text{gate}(t_g) \) assigns to each silent transition its gate, with

\[
gate(t_g) = \begin{cases} 
g^+_s, & \text{iff } t = t_{g_1} \in T_+ = \{t_{g_1} | g^+_s \in G^+_s \cup G^m_\text{dx} \} 
g^+_x, & \text{iff } t = t_{g_2} \in T^+_x = \{t_{g_2} | g^+_x \in G^+_x \} 
g^{m}_x, & \text{iff } t = t^{m}_{(n_1,g^m_\text{x})} \in T^{m}_x = \{t^{m}_{(n_1,g^m_\text{x})} | g^{m}_x \in G^{m}_x \} \end{cases}
\]

**Lemma 1 (Connected Silent Transitions imply Connected Gateways).**
Let DAC = \( (N,I,G,e_\text{s},E_\text{e},G^m_\text{s},G^m_\text{dx},G^m_\text{ex},G^m_\text{x},V,\text{class},\rightarrow,\rightarrow,\rightarrow,\rightarrow,\rightarrow,\text{dom}_V,\text{grd}) \) be a well-formed DAChor. DAC is mapped to a DAi Net \(# = (P,p(T_i,P_j,T,S,T,F,T,S,T,\text{class},\rightarrow,\rightarrow,\rightarrow,\text{dom}_V,\text{grd}) \). If there are two silent transitions \( t^1_g, t^2_g \in T_S \) that are connected by a place \( p_{(n_1,n_2)} \in P \) (cf. Def.7), i.e., \( t^1_g \rightarrow t^2_g \). Then holds \( \text{gate}(t^1_g) \rightarrow \text{gate}(t^2_g) \).

**Proof 3 (Connected Silent Transitions imply Connected Gateways (Lemma 1)).**
Let DAC = \( (N,I,G,e_\text{s},E_\text{e},G^m_\text{s},G^m_\text{dx},G^m_\text{ex},G^m_\text{x},V,\text{class},\rightarrow,\rightarrow,\rightarrow,\rightarrow,\rightarrow,\text{dom}_V,\text{grd}) \) be a (well-formed) DAChor that is mapped to a DAi Net \(# \). The latter is defined as \(# = (P,p(T_i,P_j,T,S,T,F,T,S,T,\text{class},\rightarrow,\rightarrow,\rightarrow,\text{dom}_V,\text{grd}) \).
Thus, for all cases \( \text{gate}(t_3^a) \rightarrow \text{gate}(t_3^m) \) holds and Lemma 1 is proven. \( \Box \)

We use function \( \text{gate} \) and Lemma 1 to prove the third property of well-formedness of DAI Nets by contradiction:

**Proof 4 (Preservation of Well-Formedness (Property 3)).**

Assume property 3 of Theorem 1 is violated. Then: There exists a well-formed DAC = \((N, I, G, e, E, G_w, G_{wx}, G_{dwx}, G_{dxw}, G_{dwx}, G_{dxw}, G_{wx}, V, \text{class}, \rightarrow, \rightarrow, \text{dom}_{V, \text{grd}})\) that is mapped to a non-well-formed DAI Net #, with # = \((P, p_m, P_o, P_i, T, T_S, T_I, V, \text{class'}, \rightarrow', \rightarrow', \text{dom}_{V, \text{grd}'}\)

\[ \Rightarrow \exists p_0, p_1, \ldots, p_n \in P, t_0, t_1, \ldots, t_n \in T_S : p_0 \rightarrow t_0 \rightarrow t_1 \rightarrow \cdots \rightarrow p_n \rightarrow t_n \rightarrow p_0 \]

\[ \Rightarrow \exists g_0 := \text{gate}(t_0), g_1 := \text{gate}(t_1), \ldots, g_n := \text{gate}(t_n) : g_0 \rightarrow g_1 \rightarrow \cdots \rightarrow g_n \rightarrow g_0 \]

This contradicts our assumption. Thus, the third property holds. \( \Box \)

According to Proofs 1-4, the mapping from DAC to DAI Net (cf. Def. def:mapping) preserves all three properties of well-formedness. Thus, Theorem 1 holds.

**Example 3 (Transformation).**

The DAC hor DAI = \((N, I, G, e, E, G_w, G_{wx}, G_{dwx}, G_{dxw}, G_{dwx}, G_{dxw}, G_{wx}, V, \text{class}, \rightarrow, \rightarrow, \text{dom}_{V, \text{grd}})\) from Example 2 is mapped to the DAI Net # = \((P, p_m, P_o, P_i, T, T_S, T_I, V, \text{class'}, \rightarrow', \rightarrow', \text{dom}_{V, \text{grd}'}\) as follows (cf. Fig. 4):

\[
\begin{align*}
P & = \{ p_m = p(e_{s}, i_{s}), p(i_{s}, i_{s}), p(i_{s}, i_{s}), p(i_{s}, i_{s}), p(i_{s}, i_{s}), p(i_{s}, i_{s}), p(i_{s}, i_{s}), p(i_{s}, i_{s}), p(i_{s}, i_{s}) \} \\
P_o & = \{ p(e_{s}, i_{s}) \} \\
P_i & = \{ p(i_{s}, i_{s}) \} \\
T_S & = T_I \times T_I \times T_I \\
V & = \{ \text{Status, Order} \} \\
\rightarrow' & = \{ (i_{s}, i_{s}), (i_{s}, i_{s}), (i_{s}, i_{s}), (i_{s}, i_{s}), (i_{s}, i_{s}), (i_{s}, i_{s}), (i_{s}, i_{s}), (i_{s}, i_{s}), (i_{s}, i_{s}) \}
\end{align*}
\]

\[
\begin{align*}
\text{class'}(t_1) & = \text{Request Pet} & \text{class'}(t_2) & = \text{Confirmation} & \text{class'}(t_3) & = \text{Request Trans.} \\
\text{class'}(t_4) & = \text{Request Exam.} & \text{class'}(t_5) & = \text{Result} & \text{class'}(t_6) & = \text{Arrival} \\
\text{class'}(t_7) & = \text{Retransport} & \text{class'}(t_8) & = \text{Return} & \text{class'}(t_9) & = \text{Approval} \\
\end{align*}
\]

\[
\begin{align*}
\text{dom}_{V, \text{Status}} & = \text{DStatus} & \text{dom}_{V, \text{Order}} & = \text{DOrder} \\
\text{\text{grd}(t_1^s, r_1^s)} & = \text{Status = critical} & \text{\text{grd}(t_1^s, r_1^s)} & = \text{Status = uncritical} \\
\end{align*}
\]

E. Behavior of DAI Nets

Since DAI Nets are based on both WFD Nets and IP Nets, we use token semantics (i.e., tokens assigned to places and token changes) to define their behavior. Together with the values of variables, tokens define the marking
of a DAI Net. Each Interaction Net starts with an initial marking, with exactly one token placed in the initial place $p_{in}$ and each variable having the empty value $\epsilon$. A marking is called final, if all tokens belong to final places of $P_{fi}$. A transition $t$ is activated under marking $m$, iff all directly preceding places of $t$ contain at least one token, and the guard of $t$ is evaluable and evaluates to true.

**Definition 8** (DAI Net Markings and Activated Transitions). Let $\# = (P, P_{in}, P_{o}, P_{fi}, T, T_{S}, T_{I}, V, \rightarrow, \rightarrow^{*}, dom_{V}, \text{grd})$ be a DAI Net. Then: A marking of $\#$ is a tuple $m = (\otimes, \text{val})$ with

- $\otimes: P \rightarrow \mathbb{N}_{0}$ assigns to each place the number of corresponding tokens,
- $\text{val}: V \rightarrow \Omega_{I}$ assigns to each variable its current value; $\text{val}(v)$ is either the empty value $\epsilon$ or an element of the variable’s domain, i.e., $\text{val}(v) \in \text{dom}_{V}(v) \cup \{\epsilon\}$.

Additionally, for each DAI Net $\#$ we define the

- set of all markings $\mathcal{M}_{\#}$, whereby $\mathcal{M}_{\#} := \{m = (\otimes, \text{val}) \mid m \text{ is a marking of } \# \}$
- initial marking $m_{in} := (\otimes_{in}, \text{val}_{in}) \in \mathcal{M}_{\#}$, whereby $\otimes_{in}(p) := \begin{cases} 1, & \text{if } p = p_{in} \\ 0, & \text{else} \end{cases}$ and $\forall v \in V: \text{val}_{in}(v) := \epsilon$
- set of all final markings $\mathcal{F}_{\#}$, whereby $\mathcal{F}_{\#} := \{(\otimes, \text{val}) \in \mathcal{M}_{\#} \mid \forall p \in P: \otimes(p) \neq 0 \iff p \in P_{fi}\}$
- transition activation relation $\rightarrow \subseteq \mathcal{M}_{\#} \times T$. $m \rightarrow t$ denotes that marking $m \in \mathcal{M}_{\#}$ activates transition $t \in T$, iff the following conditions hold:
  1) $\forall p \in P_{\rightarrow^{*}}: \otimes(p) \geq 1$,
  2) $\forall v \in \text{grd}(t): \text{val}(v) \neq \epsilon$,
  3) $\text{grd}(t)$ is satisfied for marking $m$

If a transition is activated, it may fire and lead from the current marking to a subsequent one. More precisely, one token is taken from each preceding place and one is added to each succeeding place. Silent transitions fire immediately when they become activated. Activated interaction transitions fire, if and only if a message of the corresponding message class is sent. In this case, the value of the message is assigned to virtual data objects as
expressed by the data assignment relation. Note that a message can only be sent if an interaction transition of the related message class is activated and no silent transition is activated (cf. Def. 9).

**Definition 9 (Options and Subsequent Markings of DAI Nets).**
Let $\# = (P, p_{in}, P_o, P_{fi}, T, T_S, T_I, V, \text{class}, \rightarrow, \rightarrow_*, \text{dom}_V, \text{grd})$ be a DAI Net, an $m = (\varnothing, \text{val}), m' = (\varnothing', \text{val}') \in M_\#$ be two related markings. Then:

- $O_\# := T_S \cup \Sigma_\#$ is the set of all options on $\#$.
- $\text{opt}_\# : M_\# \rightarrow 2^{O_\#} : m \rightarrow \{o \in O_\# | \exists m' \wedge m \stackrel{o}{\rightarrow} m'\}$ maps each marking $m$ to the options available under $m$.
- $m \rightarrow m'$ expresses that $m$ leads to $m'$ by applying option $o \in \text{opt}_\#(m)$ with:
  
  **Case 1:** $o = t_o \in T_S$ is a silent transition. Then: $m \stackrel{t_o}{\rightarrow} m'$ holds, iff each of the following conditions is met:
  
  1. $m \Rightarrow s_t o$
  2. $\forall p \in P^{-t_o} : \varnothing'(p) = \varnothing(p) - 1$,
  3. $\forall p \in P^{-t_o} : \varnothing'(p) = \varnothing(p) + 1$,
  4. $\forall v \in V : \text{val}'(v) = \text{val}(v)$.
  
  **Case 2:** $o = \mu = (c, x) \in \Sigma_\#$ is a message. Then: $m \mu \rightarrow m'$ holds, iff the following conditions are met:
  
  1. $\forall t_o \in T_S : m \Rightarrow s_t o$
  2. $\exists t_i \in T_I : m \Rightarrow s_t o \wedge \mu \in \Sigma_{\text{class}(t_i)}$
  3. $\forall p \in P^{-t_i} : \varnothing'(p) = \varnothing(p) - 1$,
  4. $\forall p \in P^{-t_i} : \varnothing'(p) = \varnothing(p) + 1$,
  5. $\forall v \in V : \text{val}'(v) = x$,
  6. $\forall v \in V : t_i \stackrel{\mu}{\rightarrow} v : \text{val}'(v) = \text{val}(v)$.

Based on Def. 9, the following two theorems can be derived.

**Theorem 2 (Separation of Options).** Let $\#$ be a DAI Net. Then: For each marking, the set of options either contains solely silent transitions or messages or it is empty, i.e.,

$$\forall m \in M_\# : \text{opt}_\#(m) \neq \varnothing \Rightarrow \text{opt}_\#(m) \subseteq T_S \oplus \text{opt}_\#(m) \subseteq \Sigma_\#$$

**Theorem 3 (Termination of final markings).** Let $\#$ be a DAI Net. Then: For each final marking, the set of options is empty, i.e.,

$$\forall m \in F_\# : \text{opt}_\#(m) = \varnothing$$

We prove Theorem 2 and Theorem 3 by contradiction:

**Proof 5 (Separation of Options).** Let $\# = (P, p_{in}, P_o, P_{fi}, T, T_S, T_I, V, \text{class}, \rightarrow, \rightarrow_*, \text{dom}_V, \text{grd})$ be a DAI Net. Then: Assume, Theorem 2 is violated for $\#$:

- $\exists m \in M_\# : \text{opt}_\#(m) \neq \varnothing$ and $\exists t_o \in T_S, \mu \in \Sigma_\# : t_o, \mu \in \text{opt}_\#(m)$
- $\exists m' : m \mu \rightarrow m' \Rightarrow \forall t_o \in T_S : m \Rightarrow s_t o \Rightarrow m \Rightarrow s_t o \Rightarrow t_o \notin \text{opt}_\#(m)$

This contradicts our assumption. Thus, Theorem 2 is proven. □
**Proof 6** (Termination of final markings).
Let \( \# = (P, p_o, P_f, T, T_S, T_I, V, \text{class}, \rightarrow, \rightarrow^*, \text{dom}_V, \text{grad}) \) be a DAI Net. Then: Assume, Theorem 3 is violated for \( \#: \)
\[
\Rightarrow \exists m = (\circ, \text{val}) \in F_\#: \text{opt}_\#(m) \neq \emptyset \\
\Rightarrow \exists t \in T: m \sim t \Rightarrow 3p \in P: (\circ(p) \geq 1 \land p \rightarrow t) \\
\Rightarrow p \notin P_f \Rightarrow m \notin F_\#
\]
This contradicts our assumption. Thus, Theorem 3 is proven. \( \square \)

Based on Def. 9, we define traces on DAI Nets as sequences of options. To be more precise, a trace corresponds to a related sequence of markings that starts with the initial marking. If this related sequence of markings ends with a final marking, we denote the trace as completed.

**Definition 10** (Traces, Prefixes, and Extensions).

(A) Let \( \#: (P, p_o, P_f, T, T_S, T_I, V, \text{class}, \rightarrow, \rightarrow^*, \text{dom}_V, \text{grad}) \) be a DAI Net and \( \tau = (\tau_k)_{k=1..n} \in O_\#^* \) be a finite sequence of options (i.e. silent transitions and messages) with length \(|\tau| = n \in \mathbb{N}\). Let further \( m = (m_k)_{k=1..n+1} \in M_\#^* \) be a finite sequence of markings with length \( n + 1 \). Then:
- \( \tau \sim m \) denotes that \( \tau \) and \( m \) are related sequences, iff \( \forall l \in [1..n] : m_l \rightarrow \tau_{l+1} \) and \( m_1 = m_\#^* \).
- last : \( M_\#^* \rightarrow M_\# \) with \( (m_k)_{k=1..n} \rightarrow m_n \) is a function mapping a sequence of markings to its last marking.
- \( \tau \in O_\#^* \) is a trace, iff \( \exists m \in M_\#^* \) and \( \tau \sim m \). If last\((m)\in F_\#, \) we denote \( \tau \) as completed trace.
- \( T_\# \) denotes the set of all traces on \( \# \).
- \( T^c_\# \) denotes the set of all completed traces on \( \# \).

(B) Let \( L \subseteq M^* \) be a set of finite sequences over a set \( M \) and let \( a = (a_k)_{k=1..l}, b = (b_k)_{k=1..l}, c = (c_m)_{m=1..m} \in L \) be elements of \( L \), i.e. sequences over \( M \). Then:
- \( a \sqsubseteq b \) (\( a \sqsubseteq b \)) denotes \( a \) is prefix (real prefix) of \( b \) and \( b \) an extension (real extension) of \( a \), iff \( n \leq l \) (\( n < l \)) and \( \forall i \in [1..n] : a_i = b_i \).
- \( a + c = b \) denotes that \( a \) is extended by \( c \) to \( b \), iff \( m + n = l \), and \( a \) is prefix of \( b \), and \( \forall i \in [1..m] : c_i = b_{n+i} \).
- \( L^{ab} := \{ a \in L | a \leq b \} \) \( (L^{ab} := \{ a \in L | a < b \}) \) denotes the subset of \( L \) that contains all prefixes (real prefixes) of \( b \) in \( L \), and \( L^{ab} := \{ a \in L | b \leq a \} \) \( (L^{ab} := \{ a \in L | b < a \}) \) denotes the subset of \( L \) that contains all extensions (real extensions) of \( b \) in \( L \).

We described the behavior of a DAI Net by means of its traces. We can also use traces to characterize the desired behavioral properties of DAI Nets. The first one is determinism. It expresses that a trace is unique in terms of its related markings, i.e., replaying a trace will always lead to the same marking. The second fundamental property is soundness in terms of boundedness as well as the absence of deadlocks and lifelocks [15].

**Definition 11** (Determinism and Soundness).

(A) We call a DAI Net \( \# \) deterministic, iff for each trace \( \tau \) on \( \# \) there exists exactly one related sequence of markings, i.e., \( \forall \tau \in T_\#: |\{ m \in M_\#^* | m \sim \tau \}| = 1. \)

Let \( \# \) be a deterministic DAI Net. Then:
- \( \text{mark}_\#: T_\# \rightarrow M_\#: \tau \mapsto \text{mark}_\#(\tau) := \text{last}(m) \), whereby \( m \) is defined by \( \tau \sim m \in M_\#^* \).

Since \( \# \) is deterministic, the definition of \( m \) is unique. Thus, \( \text{mark}_\# \) is well defined.
**Definition 12 (Projections and Conversations).**

Let $A, B$ be two sets with $B \subseteq A$, and $\# = (P, p_{in}, P_o, P_{fi}, T, T_S, T_I, V, class, \to, \to', dom_V, grd)$ be a DAI Net and $\tau \in T_\#$ be a trace on $\#$. Then:

- $\Pi_B : A^* \to B^* : a \mapsto \Pi_B(a)$ is the projection function that restricts a sequence $a \in A^*$ to its elements of $B$,
- $\eta \in \Sigma^*_\#$ denotes a conversation on $\#$, iff it is the projection of a trace on $\#$ to its messages on $\#$, i.e., $\exists (\tau) \in T_\#: \Pi_{\Sigma_\#}(\tau) = \eta$. $\eta$ denotes a completed conversation on $\#$, iff it is the projection of a completed trace on $\#$.
- $C_\#$ denotes the set of all conversations on $\#$.
- $C^c_\#$ denotes the set of all completed conversations on $\#$.
- $\text{con}_{\#} : T_\# \to C_\# : \tau \mapsto \text{con}_{\#}(\tau) := \Pi_{\Sigma_\#}(\tau)$ maps each trace to the related conversation.

**Example 4 (Traces and Conversations).**

Consider the DAI Net $\#$ from Example 3. Its set of completed traces $T^c_\#$ consists of traces $\tau_1, \tau_2,$ and $\tau_3$. Projecting them to their messages leads to the conversations $\eta_1, \eta_2,$ and $\eta_3$, which build $C^c_\#$:

$\tau_1 = \langle \text{Request PET, uncritical}, \text{, Confirmation, }_1\rangle, \langle \text{Request Trans, }_1\rangle, t\rhd^m_{(g_2, g_2)}, t\rhd_{(g_2, g_2)}\rangle,$
\hspace{1em} (Arrival,$\epsilon$), (Retransport,$\epsilon$), (Return,$\epsilon$) >

$\tau_2 = \langle \text{Request PET, critical}, \text{, Confirmation, }_1\rangle, \langle \text{Request Trans, }_1\rangle, t\rhd^m_{(g_2, g_2)}, t\rhd_{(g_2, g_2)}\rangle,$
\hspace{1em} (Result,$\text{abort}$), (Request Exam,$\epsilon$), (Return,$\epsilon$) >

$\tau_3 = \langle \text{Request PET, critical}, \text{, Confirmation, }_1\rangle, \langle \text{Request Trans, }_1\rangle, t\rhd^m_{(g_2, g_2)}, t\rhd_{(g_2, g_2)}\rangle,$
\hspace{1em} (Result,$\text{continue}$), (Request Exam,$\epsilon$), (Return,$\epsilon$) >

$\eta_1 = \text{con}_{\#} (\tau_1) \Rightarrow \Pi_{\Sigma_\#}(\tau_1) = \langle \text{Request PET, uncritical}, \text{, Confirmation, }_1\rangle, \langle \text{Request Trans, }_1\rangle,$
\hspace{1em} (Arrival,$\epsilon$), (Retransport,$\epsilon$), (Return,$\epsilon$) >

$\eta_2 = \text{con}_{\#} (\tau_2) \Rightarrow \Pi_{\Sigma_\#}(\tau_2) = \langle \text{Request PET, critical}, \text{, Confirmation, }_1\rangle, \langle \text{Request Trans, }_1\rangle,$
\hspace{1em} (Request Exam,$\epsilon$), (Result,$\text{abort}$) >

$\eta_3 = \text{con}_{\#} (\tau_3) \Rightarrow \Pi_{\Sigma_\#}(\tau_3) = \langle \text{Request PET, critical}, \text{, Confirmation, }_1\rangle, \langle \text{Request Trans, }_1\rangle,$
\hspace{1em} (Request Exam,$\epsilon$), (Result,$\text{continue}$), (Arrival,$\epsilon$), (Retransport,$\epsilon$), (Return,$\epsilon$) >

As aforementioned, the behavior of silent transitions is not observable. Hence, to ensure compatible behavior of participating roles, silent transitions must behave deterministically. In other words, it must be possible to determine the behavior of a DAI Net solely based on the messages exchanged, i.e., message-determinism. First, this requires,
that firing of silent transitions always terminates, i.e., it is impossible to solely execute silent transitions infinitely (cf. Theorem 4). Second, when silent transitions terminate, the set of activated options may only depend on the messages exchanged before, i.e., it should be independent from the order in which the silent transitions were fired.

**Theorem 4 (Termination of silent subtraces).** On a well-formed DAI Net $\#$, any trace cannot infinitely be continued by silent transitions, i.e.

$$\forall \tau \in T_\#: \exists N \in \mathbb{N} \text{ such that } \forall v \in T_\#: \tau \Rightarrow |\tau| + N < |v| \Rightarrow \text{con}_{\#}(\tau) \neq \text{con}_{\#}(v).$$

To proof Theorem 4, we introduce silent ways, which solely consist of places and silent transitions. With the use of those, we define a ranking function that decreases each time a silent transition is fired.

**Definition (Silent Ways).**

Let $\# = (P, p_{in}, P_o, P_f, T, T_S, T_I, V, \text{class}, \to, \to, \text{dom}_V, \text{grd})$ be a deterministic and sound DAI Net. Then:

- A silent way $\omega$ from a place $p \in P$ to a silent transition $t \in T_S$ is a sequence of alternating places and silent transitions $\omega = \langle p_0, t_0, p_1, t_1, \ldots, p_n, t_n \rangle \in (PT_S)^*$ with $p = p_0$ and $t = t_n$, whereby holds $p_0 \rightarrow t_0 \rightarrow p_1 \rightarrow t_1 \cdots \rightarrow p_n \rightarrow t_n$.
- $\mathcal{W}_#$ as the set of all silent ways on $\#$, and
- $\mathcal{W}_{p \rightarrow t}^# := \{ \langle p_0, \ldots, t_n \rangle \in \mathcal{W}_# | p = p \land t_n = t \}$ as the set of all silent ways on $\#$ from $p \in P$ to $t \in T_S$.

Consider that Def. 6 prohibits cycles of silent transitions. Thus, each place and each transition can occur at least once in a silent way. Consequently, $\mathcal{W}_#$ is finite. Obviously, the same applies to each $\mathcal{W}_{p \rightarrow t}^# \subseteq \mathcal{W}_#$

**Lemma 2.**

Let $\# = (P, p_{in}, P_o, P_f, T, T_S, T_I, V, \text{class}, \to, \to, \text{dom}_V, \text{grd})$ be a deterministic and sound DAI Net and $t, t' \in T_S$ be silent transitions. Then:

$$\forall p \in P^{t \rightarrow t'} : \sum_{q \in P^{t \rightarrow t'}} |\mathcal{W}_{q \rightarrow t'}^#| \geq |\mathcal{W}_{p \rightarrow t'}^#| \geq \sum_{q \in P^{t \rightarrow t'}} |\mathcal{W}_{q \rightarrow t'}^#|$$

**Proof 7.**

The right inequality $|\mathcal{W}_{p \rightarrow t'}^#| \geq \sum_{q \in P^{t \rightarrow t'}} |\mathcal{W}_{q \rightarrow t'}^#|$ holds because of:

$$\forall q \in P^{t \rightarrow t'} : p \rightarrow t \rightarrow q \Rightarrow \forall q \in P^{t \rightarrow t'} : \forall \omega = \langle q, t_0, p_1, t_1, \ldots, p_n, t' \rangle \in \mathcal{W}_{q \rightarrow t'}^# \Rightarrow \exists! \omega' = \langle p, t, q, t_0, p_1, t_1, \ldots, p_n, t' \rangle \in \mathcal{W}_{p \rightarrow t'}^#$$

The left inequality $\forall p \in P^{t \rightarrow t'} : \sum_{q \in P^{t \rightarrow t'}} |\mathcal{W}_{q \rightarrow t'}^#| \geq |\mathcal{W}_{p \rightarrow t'}^#|$ holds because of:

$$p \in P^{t \rightarrow t'} \Rightarrow \bigcup_{q \in P^{t \rightarrow t'}} \mathcal{W}_{q \rightarrow t'}^# \supseteq \mathcal{W}_{p \rightarrow t'}^# \Rightarrow \sum_{q \in P^{t \rightarrow t'}} |\mathcal{W}_{q \rightarrow t'}^#| \geq |\bigcup_{q \in P^{t \rightarrow t'}} \mathcal{W}_{q \rightarrow t'}^#| \geq |\mathcal{W}_{p \rightarrow t'}^#|$$

Thus, Lemma 2 is proven. □

---

\(^1\)For reasons of simplification, we abstract from irrelevant message content in Example 4
Based on Lemma 2 we prove Theorem 4. For this purpose, first, we introduce a transition ranking function $\zeta$ that maps a marking and a silent transition to a natural number. Second, based on $\zeta$, we define a net ranking function $\xi$ that bases on maps each marking of a $\#$ to a natural number. Finally, we show, that $\xi$ is decreased each time a silent transition fires (cf. Lemma 3). Thus, $\xi$ is an upper bound to the number of steps (i.e., firings of silent transitions) that can be done in $\#$ until the net terminates.

**Definition (Ranking Functions).** Let $\# = (P, p_{in}, P_o, P_{fi}, T, T_S, T_I, V, \text{class}, \rightarrow, \rightarrow^*, \text{dom}_V, \text{grd})$ be a deterministic and sound DAI Net. Then, the transition ranking function $\zeta$ is defined as below:

- $\zeta_# : M_# \times T_S \rightarrow \mathbb{N} : ((\circ, \text{val}), t) \mapsto \zeta((\circ, \text{val}), t) := \sum_{p \in P} \circ(p) \ast |\mathcal{W}_#|_{\rightarrow^t}$.

- $\xi_# : M_# \rightarrow \mathbb{N} : m \mapsto \xi(m)$ is the net ranking function that maps a marking to a natural number, with $\xi(m) := \sum_{t \in T_S} \zeta(m, t)$.

**Lemma 3 (The Rank of a Net Decreases when a Silent Transition Fires).** Let $\# = (P, p_{in}, P_o, P_{fi}, T, T_S, T_I, V, \text{class}, \rightarrow, \rightarrow^*, \text{dom}_V, \text{grd})$ be a deterministic and sound DAI Net. Further, $m, m' \in M_#$ be two markings of $\#$, with $m = (\circ, \text{val}), m' = (\circ', \text{val}')$. Finally, $t_0 \in T_S$ be a silent transition of $\#$ with $m \xrightarrow{t_0} m'$. Then:

$$\xi(m) > \xi(m')$$

To prove Lemma 3, we partition $\xi$ based on its definition for each $t_0 \in T_S$ that may be fired:

$$\xi(m) = \sum_{t \in T_S} \zeta(m, t) = \zeta(m, t_0) + \sum_{t \in T_S - \{t_0\}} \zeta(m, t)$$

We consider both parts (i.e. $\zeta(m, t_0)$ and $\sum_{t \in T_S - \{t_0\}} \zeta(m, t)$) on its own to show that $\xi$ decreases, whenever a silent transition $t_0$ is fired.

**Lemma 4 (The Rank of a Fired Silent Transition Decreases).**

Let $\# = (P, p_{in}, P_o, P_{fi}, T, T_S, T_I, V, \text{class}, \rightarrow, \rightarrow^*, \text{dom}_V, \text{grd})$ be a deterministic and sound DAI Net. Further, $m, m' \in M_#$ be two markings of $\#$, with $m = (\circ, \text{val}), m' = (\circ', \text{val}')$. Finally, $t_0 \in T_S$ be a silent transition of $\#$ with $m \xrightarrow{t_0} m'$. Then:

$$\zeta(m, t_0) > \zeta(m', t_0)$$

**Proof 8 (The Rank of a Fired Silent Transition Decreases (Lemma 4)).**
Lemma 5 (The Rank-Sum of all Unfired Silent Transitions does not Increase).
Let $\# = (P, p_m, P_o, P_f, T, T_S, T_I, V, \text{class}, \rightarrow, +, \text{dom}, V, \text{grd})$ be a deterministic and sound DAI Net. Further, $m, m' \in M_{\#}$ be two markings of $\#$, with $m = (\circ, \text{val}), m' = (\circ', \text{val}')$. Finally, $t_0 \in T_S$ be a silent transition of $\#$ with $m \rightarrow m'$. Then:

$$
\sum_{t \in T_S - \{t_0\}} \zeta(m, t) \geq \sum_{t \in T_S - \{t_0\}} \zeta(m', t)
$$

Proof 9 (The Rank-Sum of all Unfired Silent Transitions does not Increase (Lemma 5)).

$$
\zeta(m, t_0) = \sum_{p \in P} (\circ(p) \cdot |W_{\#}^p|) = \sum_{p \in P_{\#}} (\circ(p) \cdot |W_{\#}^p|) + \sum_{p \in P_{\#}} (\circ(p) \cdot |W_{\#}^p|) + \sum_{p \in P_{\#}} (\circ(p) \cdot |W_{\#}^p|)
$$

$$
= \sum_{p \in P_{\#}} ((\circ'(p) + 1) \cdot |W_{\#}^p|) + \sum_{p \in P_{\#}} (\circ(p) \cdot |W_{\#}^p|) + \sum_{p \in P_{\#}} (\circ'(p) \cdot |W_{\#}^p|)
$$

$$
\geq \sum_{p \in P_{\#}} (\circ'(p) \cdot |W_{\#}^p|) + \sum_{p \in P_{\#}} (\circ'(p) \cdot |W_{\#}^p|)
$$

Thus, $\zeta(m, t_0) > \zeta(m', t_0)$ holds and Lemma 4 is proven. \(\square\)
Thus, \( \sum_{t \in T - \{t_0\}} \zeta(m, t) \geq \sum_{t \in T - \{t_0\}} \zeta(m', t) \) holds and Lemma 5 is proven. □

Having proven that Lemma 4 and Lemma 5 hold, we now can prove Lemma 3:

**Proof 10** (The Rank of a Net Decreases when a Silent Transition Fires (Lemma 3)).

\[
\xi(m) = \sum_{t \in T} \zeta(m, t) = \zeta(m, t_0) + \sum_{t \in T - \{t_0\}} \zeta(m, t) > \zeta(m', t_0) + \sum_{t \in T - \{t_0\}} \zeta(m', t) = \sum_{t \in T} \zeta(m', t) = \xi(m')
\]

Thus, \( \xi(m) > \xi(m') \) holds and Lemma 3 is proven. □

The termination of silent subtraces (i.e, Theorem 4) is a direct consequence of Lemma 3:

**Proof 11** (Termination of Silent Subtraces).

From Lemma 3 results:

\[
\forall \tau \in T : \exists N := \xi(\text{mark}_\#(\tau)) \in \mathbb{N} : \forall v \in T^\tau : |\tau| + N < |v| \Rightarrow \text{con}_\#(\tau) \neq \text{con}_\#(v).
\]

Thus, Theorem 4 is proven. □

According to Theorem 4, a DAI Net is message-deterministic, if the set of activated messages solely depends on the messages exchanged before (cf. Def. 13).

**Definition 13** (Message-Determinism).

We call a deterministic and sound DAI Net \( \# \) message-deterministic, iff the same sequence of messages always activates the same messages, i.e., the set of activated messages solely depends on the messages exchanged before, i.e.,

\[
\forall \tau, v \in T_\# : \left( \text{opt}_\#(\text{mark}_\#(\tau)), \text{opt}_\#(\text{mark}_\#(v)) \in \Sigma_\# \land \Pi_{\Sigma_\#}(\tau) = \Pi_{\Sigma_\#}(v) \right) \Rightarrow \text{opt}_\#(\text{mark}_\#(\tau)) = \text{opt}_\#(\text{mark}_\#(v))
\]

Let \( \# \) be a deterministic, sound and message-deterministic DAI Net. Then:

\[
\text{mo}_\# : C_\# \rightarrow 2^{\Sigma_\#} : \eta \mapsto \text{mo}_\#(\eta) \text{ maps each conversation to the set of messages it activates, with } \text{mo}_\#(\eta) := \text{opt}_\#(\text{mark}_\#(\tau)), \tau \in O^\#_\# \text{ is defined by } \eta = \text{con}_\#(\tau) \text{ and } \text{opt}_\#(\text{mark}_\#(\tau)) \in \Sigma_\#.
\]

Since \( \# \) is message-deterministic, the definition is unique. Thus, \( \text{mo}_\# \) is well defined.

Until now, we solely considered DAI Nets and conversations from a global perspective. However, a role solely knows those messages of a conversation it sends or receives. Thus, in Def. 14 the view of a role on the messages of a conversation is introduced. Further, for each role the set of activated options is defined.

**Definition 14** (Views on Conversations and Options).

Let \( \mathcal{I} = (\mathcal{R}, \mathcal{D}, C, \text{dom}_C, s_C, r_C, \epsilon) \) be an interaction domain and let the tuple \( \# = (\mathcal{P}, \pi_m, P_o, P_{f_1}, T, T_\#, T_I, V, \text{class}, \rightarrow, \cdot, \text{dom}_V, \text{grd}) \) be a sound, deterministic, and message-deterministic DAI Net. Let further \( R \in \mathcal{R} \) be a role. Then we can define the following views.
Based on Def. 14, we can define realizability. It denotes DAI Nets to be deterministic from the viewpoint of a role. Further, clear termination is defined, which indicates that a role can determine when it sent or received its last message.

**Definition 15 (Realizability, Clear Termination).**  
Let \( \# \) be a deterministic, sound, and message-deterministic DAI Net. Then, for a role \( R \in \mathcal{R} \):

- \( \# \) is realizable, iff the messages role \( R \) may send solely depend on the messages \( R \) has sent and received before, i.e.,
  \[
  \forall R \in \mathcal{R} : \forall \eta, \kappa \in C_\#: vC_\#^R(\eta) = vC_\#^R(\kappa) \Rightarrow vO_\#^R(mo_\#(\eta)) = vO_\#^R(mo_\#(\kappa))
  \]

- \( \# \) clearly terminates, iff it solely depends on the messages \( R \) has sent and received before whether further interaction with \( R \) will occur, i.e.,
  \[
  \forall R \in \mathcal{R} : \forall \eta \in C_\#, \kappa \in C_\#: vC_\#^R(\eta) < vC_\#^R(\kappa)
  \]

An important issue concerns decidability of the introduced properties of DAI Nets and DACHors; i.e., determinism, soundness, message-determinism, realizability, and clear termination (cf. Def. 11-15). Basically, these properties are decidable. Due to lack of space, we omit a discussion in this technical report.

**IV. RELATED WORK**

In the context of workflows [1], [16] and SOA [17], correctness has been discussed for a long time [15]. The approaches presented [12], [18] consider data as well. The two paradigms for modeling choreographies (i.e. interconnection and interaction models) are compared in [19]. Examples of interconnection models are BPMN 2.0 Collaborations [3] and BPEL4Chor [4]. There are several approaches that discuss the verification classic soundness criteria (i.e. boundedness, absence of deadlocks, absence and lifelocks) of distributed and collaborative workflows and service orchestrations [5], [13], [14], [20]–[24]. Some object-aware [25], [26] and data-driven approaches [27], [28] use data dependencies to interconnect processes and to define process interactions. Examples of interaction models (i.e., the paradigm we apply) include Service Interaction Patterns [7], WSCDL [8], iBPMN Choreographies [9], and BPMN 2.0 Choreographies [3]. Our approach has been mainly inspired by [9], which defines the behavior of iBPMN Choreographies through their transformation to Interaction Petri Nets and further discusses correctness and realizability. Realizability of interaction models is also discussed in [10], [29]. Furthermore, [11] provides a tool for checking realizability of BPMN 2.0 Choreographies. However, all these approaches do not explicitly consider the data exchanged by messages and used for routing decisions.

In [30], [31], state-based conversation protocols are introduced, which are aware of message contents. The messages (and data) exchanged trigger state transitions. Thus, different data may trigger different transitions. However, conversation protocols do not support the modeling of parallelism since they are state-based. Furthermore, realizability of conversation protocols requires that at every state each partner is either able to send or receive a
message or to terminate (autonomy condition). This condition strongly restricts parallelism. For example, consider a choreography solely consisting of two parallel branches: In the upper branch partner $A$ sends a message $m_1$ to partner $B$ and partner $B$ sends message $m_2$ to $A$ in the lower branch. Obviously, the autonomy condition is violated although the choreography is realizable (cf. Def. 15). Hence, conversation protocols do not constitute interaction models in our point of view. Thus, to our best knowledge the framework presented within this technical report is the first one that considers realizability and clear termination of data-aware interaction models.

V. SUMMARY AND OUTLOOK

Our vision is to provide sophisticated support for distributed and collaborative workflows. To foster this vision, we base our work on the analysis of scenarios from different domains. In essence, we learned that data support is practically relevant for interaction models from a variety of domains.

Further, this technical report introduced a formal framework for data-aware interaction models and described how correctness can be ensured. The main parts of our framework include DACHors and DAI Nets as well as the transformation of DACHors to DAI Nets. Further, the behavior of DAI Nets is defined. Other fundamental contributions are the definitions of correctness criteria for data-aware interaction models. The latter include message-determinism, realizability, and clear termination. In future work, we will extend our framework to support asynchronous message exchange and related correctness properties. Finally, we will develop algorithms for efficiently checking correctness of data-aware interaction models. In this context, we plan to apply abstraction strategies to large data domains similar to [32].

Considering the data perspective is important but may be not sufficient to enable sophisticated support for distributed and collaborative workflows. The time perspective [33]–[35] and the resource perspective [36], [37] should be considered as well in the context of interaction modeling.

However, correctness criteria discussed in this technical report solely address structural and behavioral correctness. As outlined in [38] semantic correctness (i.e., business process compliance) is challenging for distributed and collaborative processes as well. Thus, we will try to transfer the results of our previous work about business process compliance [32], [39]–[41] to distributed and collaborative processes.

Fields of application of our research may be domains with collaborative and heavily interacting processes, e.g., healthcare domain [42] and automotive domain [28], [43].

ACKNOWLEDGMENT

This work was done within the research project C³Pro funded by the German Research Foundation (DFG), Project number: RE 1402/2-1. Parts of this technical report will be published in [44].

REFERENCES


Liste der bisher erschienenen Ulmer Informatik-Berichte

Einige davon sind per FTP von ftp.informatik.uni-ulm.de erhältlich
Die mit * markierten Berichte sind vergriffen

List of technical reports published by the University of Ulm
Some of them are available by FTP from ftp.informatik.uni-ulm.de
Reports marked with * are out of print

91-01  Ker-I Ko, P. Orponen, U. Schöning, O. Watanabe
       Instance Complexity

91-02* K. Gladitz, H. Fassbender, H. Vogler
       Compiler-Based Implementation of Syntax-Directed Functional Programming

91-03* Alfons Geser
       Relative Termination

91-04* J. Köbler, U. Schöning, J. Toran
       Graph Isomorphism is low for PP

91-05  Johannes Köbler, Thomas Thierauf
       Complexity Restricted Advice Functions

91-06* Uwe Schöning
       Recent Highlights in Structural Complexity Theory

91-07* F. Green, J. Köbler, J. Toran
       The Power of Middle Bit

91-08* V. Arvind, Y. Han, L. Hamachandra, J. Köbler, A. Lozano, M. Mundhenk, A. Ogiwara,
       U. Schöning, R. Silvestri, T. Thierauf
       Reductions for Sets of Low Information Content

92-01* Vikraman Arvind, Johannes Köbler, Martin Mundhenk
       On Bounded Truth-Table and Conjunctive Reductions to Sparse and Tally Sets

92-02* Thomas Noll, Heiko Vogler
       Top-down Parsing with Simulataneous Evaluation of Noncircular Attribute Grammars

92-03  Fakultät für Informatik
       17. Workshop über Komplexitätstheorie, effiziente Algorithmen und Datenstrukturen

92-04* V. Arvind, J. Köbler, M. Mundhenk
       Lowness and the Complexity of Sparse and Tally Descriptions

92-05* Johannes Köbler
       Locating P/poly Optimally in the Extended Low Hierarchy

92-06* Armin Kühnemann, Heiko Vogler
       Synthesized and inherited functions -a new computational model for syntax-directed
       semantics

92-07* Heinz Fassbender, Heiko Vogler
       A Universal Unification Algorithm Based on Unification-Driven Leftmost Outermost
       Narrowing
92-08* Uwe Schöning
On Random Reductions from Sparse Sets to Tally Sets

92-09* Hermann von Hasseln, Laura Martignon
Consistency in Stochastic Network

92-10 Michael Schmitt
A Slightly Improved Upper Bound on the Size of Weights Sufficient to Represent Any Linearly Separable Boolean Function

92-11 Johannes Köbler, Seinosuke Toda
On the Power of Generalized MOD-Classes

92-12 V. Arvind, J. Köbler, M. Mundhenk
Reliable Reductions, High Sets and Low Sets

92-13 Alfons Geser
On a monotonic semantic path ordering

92-14* Joost Engelfriet, Heiko Vogler
The Translation Power of Top-Down Tree-To-Graph Transducers

93-01 Alfred Lupper, Konrad Froitzheim
AppleTalk Link Access Protocol basierend auf dem Abstract Personal Communications Manager

The COCOON Object Model

93-03 Thomas Thierauf, Seinosuke Toda, Osamu Watanabe
On Sets Bounded Truth-Table Reducible to P-selective Sets

93-04 Jin-Yi Cai, Frederic Green, Thomas Thierauf
On the Correlation of Symmetric Functions

93-05 K. Kuhn, M. Reichert, M. Nathe, T. Beuter, C. Heinlein, P. Dadam
A Conceptual Approach to an Open Hospital Information System

93-06 Klaus Gaßner
Rechnerunterstützung für die konzeptuelle Modellierung

93-07 Ullrich Keßler, Peter Dadam
Towards Customizable, Flexible Storage Structures for Complex Objects

94-01 Michael Schmitt
On the Complexity of Consistency Problems for Neurons with Binary Weights

94-02 Armin Kühnemann, Heiko Vogler
A Pumping Lemma for Output Languages of Attributed Tree Transducers

94-03 Harry Buhrman, Jim Kadin, Thomas Thierauf
On Functions Computable with Nonadaptive Queries to NP

94-04 Heinz Faßbender, Heiko Vogler, Andrea Wedel
Implementation of a Deterministic Partial E-Unification Algorithm for Macro Tree Transducers
V. Arvind, J. Köbler, R. Schuler
On Helping and Interactive Proof Systems

Christian Kalus, Peter Dadam
Incorporating record subtyping into a relational data model

Markus Tresch, Marc H. Scholl
A Classification of Multi-Database Languages

Friedrich von Henke, Harald Rueß
Arbeitstreffen Typtheorie: Zusammenfassung der Beiträge

Construction and Deduction Methods for the Formal Development of Software

Axel Dold
Formalisierung schematischer Algorithmen

Johannes Köbler, Osamu Watanabe
New Collapse Consequences of NP Having Small Circuits

Rainer Schuler
On Average Polynomial Time

Rainer Schuler, Osamu Watanabe
Towards Average-Case Complexity Analysis of NP Optimization Problems

Wolfram Schulte, Ton Vullinghs
Linking Reactive Software to the X-Window System

Alfred Lupper
Namensverwaltung und Adressierung in Distributed Shared Memory-Systemen

Robert Regn
Verteilte Unix-Betriebssysteme

Helmuth Partsch
Again on Recognition and Parsing of Context-Free Grammars: Two Exercises in Transformational Programming

Helmuth Partsch
Transformational Development of Data-Parallel Algorithms: an Example

Oleg Verbitsky
On the Largest Common Subgraph Problem

Uwe Schöning
Complexity of Presburger Arithmetic with Fixed Quantifier Dimension

Harry Buhrman, Thomas Thierauf
The Complexity of Generating and Checking Proofs of Membership

Rainer Schuler, Tomoyuki Yamakami
Structural Average Case Complexity

Klaus Achatz, Wolfram Schulte
Architecture Independent Massive Parallelization of Divide-And-Conquer Algorithms
95-06  Christoph Karg, Rainer Schuler
Structure in Average Case Complexity

95-07  P. Dadam, K. Kuhn, M. Reichert, T. Beuter, M. Nathe
ADEPT: Ein integrierender Ansatz zur Entwicklung flexibler, zuverlässiger
kooperierender Assistenzsysteme in klinischen Anwendungsumgebungen

95-08  Jürgen Kehrer, Peter Schulthess
Aufbereitung von gescannten Röntgenbildern zur filmlosen Diagnostik

95-09  Hans-Jörg Burtschick, Wolfgang Lindner
On Sets Turing Reducible to P-Selective Sets

95-10  Boris Hartmann
Berücksichtigung lokaler Randbedingung bei globaler Zieloptimierung mit neuronalen
Netzen am Beispiel Truck Backer-Upper

95-11  Thomas Beuter, Peter Dadam:
Prinzipien der Replikationskontrolle in verteilten Systemen

95-12  Klaus Achatz, Wolfram Schulte
Massive Parallelization of Divide-and-Conquer Algorithms over Powerlists

95-13  Andrea Mößle, Heiko Vogler
Efficient Call-by-value Evaluation Strategy of Primitive Recursive Program Schemes

95-14  Axel Dold, Friedrich W. von Henke, Holger Pfeifer, Harald Rueß
A Generic Specification for Verifying Peephole Optimizations

96-01  Ercüment Canver, Jan-Tecker Gayen, Adam Moik
Formale Entwicklung der Steuerungssoftware für eine elektrisch ortsbediente Weiche
mit VSE

96-02  Bernhard Nebel
Solving Hard Qualitative Temporal Reasoning Problems: Evaluating the Efficiency of
Using the ORD-Horn Class

96-03  Ton Vullinghs, Wolfram Schulte, Thilo Schwinn
An Introduction to TkGofer

96-04  Thomas Beuter, Peter Dadam
Anwendungsspezifische Anforderungen an Workflow-Mangement-Systeme am
Beispiel der Domäne Concurrent-Engineering

96-05  Gerhard Schellhorn, Wolfgang Ahrendt
Verification of a Prolog Compiler - First Steps with KIV

96-06  Manindra Agrawal, Thomas Thierauf
Satisfiability Problems

96-07  Vikraman Arvind, Jacobo Torán
A nonadaptive NC Checker for Permutation Group Intersection

96-08  David Cyrluk, Oliver Möller, Harald Rueß
An Efficient Decision Procedure for a Theory of Fix-Sized Bitvectors with
Composition and Extraction
96-09 Bernd Biechele, Dietmar Ernst, Frank Houdek, Joachim Schmid, Wolfram Schulte
Erfahrungen bei der Modellierung eingebetteter Systeme mit verschiedenen SA/RT-Ansätzen

96-10 Falk Bartels, Axel Dold, Friedrich W. von Henke, Holger Pfeifer, Harald Rueß
Formalizing Fixed-Point Theory in PVS

96-11 Axel Dold, Friedrich W. von Henke, Holger Pfeifer, Harald Rueß
Mechanized Semantics of Simple Imperative Programming Constructs

96-12 Axel Dold, Friedrich W. von Henke, Holger Pfeifer, Harald Rueß
Generic Compilation Schemes for Simple Programming Constructs

96-13 Klaus Achatz, Helmut Partsch
From Descriptive Specifications to Operational ones: A Powerful Transformation Rule, its Applications and Variants

97-01 Jochen Messner
Pattern Matching in Trace Monoids

97-02 Wolfgang Lindner, Rainer Schuler
A Small Span Theorem within P

97-03 Thomas Bauer, Peter Dadam
A Distributed Execution Environment for Large-Scale Workflow Management Systems with Subnets and Server Migration

97-04 Christian Heinlein, Peter Dadam
Interaction Expressions - A Powerful Formalism for Describing Inter-Workflow Dependencies

97-05 Vikraman Arvind, Johannes Köbler
On Pseudorandomness and Resource-Bounded Measure

97-06 Gerhard Partsch
Punkt-zu-Punkt- und Mehrpunkt-basierende LAN-Integrationsstrategien für den digitalen Mobilfunkstandard DECT

97-07 Manfred Reichert, Peter Dadam
ADEPTflex - Supporting Dynamic Changes of Workflows Without Loosing Control

97-08 Hans Braxmeier, Dietmar Ernst, Andrea Mößle, Heiko Vogler
The Project NoName - A functional programming language with its development environment

97-09 Christian Heinlein
Grundlagen von Interaktionsausdrücken

97-10 Christian Heinlein
Graphische Repräsentation von Interaktionsausdrücken

97-11 Christian Heinlein
Sprachtheoretische Semantik von Interaktionsausdrücken
97-12 Gerhard Schellhorn, Wolfgang Reif
Proving Properties of Finite Enumerations: A Problem Set for Automated Theorem Provers

97-13 Dietmar Ernst, Frank Houdek, Wolfram Schulte, Thilo Schwinn
Experimenteller Vergleich statischer und dynamischer Softwareprüfung für eingebettete Systeme

97-14 Wolfgang Reif, Gerhard Schellhorn
Theorem Proving in Large Theories

97-15 Thomas Wennekers
Asymptotik rekurrenter neuronaler Netze mit zufälligen Kopplungen

97-16 Peter Dadam, Klaus Kuhn, Manfred Reichert
Clinical Workflows - The Killer Application for Process-oriented Information Systems?

97-17 Mohammad Ali Livani, Jörg Kaiser
EDF Consensus on CAN Bus Access in Dynamic Real-Time Applications

97-18 Johannes Köbler, Rainer Schuler
Using Efficient Average-Case Algorithms to Collapse Worst-Case Complexity Classes

98-01 Daniela Damm, Lutz Claes, Friedrich W. von Henke, Alexander Seitz, Adelinde Uhrmacher, Steffen Wolf
Ein fallbasiertes System für die Interpretation von Literatur zur Knochenheilung

98-02 Thomas Bauer, Peter Dadam
Architekturen für skalierbare Workflow-Management-Systeme - Klassifikation und Analyse

98-03 Marko Luther, Martin Strecker
A guided tour through Typelab

98-04 Heiko Neumann, Luiz Pessoa
Visual Filling-in and Surface Property Reconstruction

98-05 Ercüment Canver
Formal Verification of a Coordinated Atomic Action Based Design

98-06 Andreas Küchler
On the Correspondence between Neural Folding Architectures and Tree Automata

98-07 Heiko Neumann, Thorsten Hansen, Luiz Pessoa
Interaction of ON and OFF Pathways for Visual Contrast Measurement

98-08 Thomas Wennekers
Synfire Graphs: From Spike Patterns to Automata of Spiking Neurons

98-09 Thomas Bauer, Peter Dadam
Variable Migration von Workflows in ADEPT

98-10 Heiko Neumann, Wolfgang Sepp
Recurrent V1 – V2 Interaction in Early Visual Boundary Processing
98-11  Frank Houdek, Dietmar Ernst, Thilo Schwinn
Prüfen von C–Code und Statmate/Matlab–Spezifikationen: Ein Experiment

98-12  Gerhard Schellhorn
Proving Properties of Directed Graphs: A Problem Set for Automated Theorem
Provers

98-13  Gerhard Schellhorn, Wolfgang Reif
Theorems from Compiler Verification: A Problem Set for Automated Theorem
Provers

98-14  Mohammad Ali Livani
SHARE: A Transparent Mechanism for Reliable Broadcast Delivery in CAN

98-15  Mohammad Ali Livani, Jörg Kaiser
Predictable Atomic Multicast in the Controller Area Network (CAN)

99-01  Susanne Boll, Wolfgang Klas, Utz Westermann
A Comparison of Multimedia Document Models Concerning Advanced Requirements

99-02  Thomas Bauer, Peter Dadam
Verteilungsmodelle für Workflow-Management-Systeme - Klassifikation und
Simulation

99-03  Uwe Schöning
On the Complexity of Constraint Satisfaction

99-04  Ercument Canver
Model-Checking zur Analyse von Message Sequence Charts über Statecharts

99-05  Johannes Köbler, Wolfgang Lindner, Rainer Schuler
Derandomizing RP if Boolean Circuits are not Learnable

99-06  Utz Westermann, Wolfgang Klas
Architecture of a DataBlade Module for the Integrated Management of Multimedia
Assets

99-07  Peter Dadam, Manfred Reichert
Enterprise-wide and Cross-enterprise Workflow Management: Concepts, Systems,
Applications. Paderborn, Germany, October 6, 1999, GI–Workshop Proceedings,
Informatik '99

99-08  Vikraman Arvind, Johannes Köbler
Graph Isomorphism is Low for ZPPNP and other Lowness results

99-09  Thomas Bauer, Peter Dadam
Efficient Distributed Workflow Management Based on Variable Server Assignments

2000-02  Thomas Bauer, Peter Dadam
Variable Serverzuordnungen und komplexe Bearbeiterzuordnungen im Workflow-
Management-System ADEPT

2000-03  Gregory Baratoff, Christian Toepfer, Heiko Neumann
Combined space-variant maps for optical flow based navigation
2000-04  Wolfgang Gehring
Ein Rahmenwerk zur Einführung von Leistungspunktsystemen

2000-05  Susanne Boll, Christian Heinlein, Wolfgang Klas, Jochen Wandel
Intelligent Prefetching and Buffering for Interactive Streaming of MPEG Videos

2000-06  Wolfgang Reif, Gerhard Schellhorn, Andreas Thums
Fehlersuche in Formalen Spezifikationen

2000-07  Gerhard Schellhorn, Wolfgang Reif (eds.)

2000-08  Thomas Bauer, Manfred Reichert, Peter Dadam
Effiziente Durchführung von Prozessmigrationen in verteilten Workflow-Management-Systemen

2000-09  Thomas Bauer, Peter Dadam
Vermeidung von Überlastsituationen durch Replikation von Workflow-Servern in ADEPT

2000-10  Thomas Bauer, Manfred Reichert, Peter Dadam
Adaptives und verteiltes Workflow-Management

2000-11  Christian Heinlein
Workflow and Process Synchronization with Interaction Expressions and Graphs

2001-01  Hubert Hug, Rainer Schuler
DNA-based parallel computation of simple arithmetic

2001-02  Friedhelm Schwenker, Hans A. Kestler, Günther Palm
3-D Visual Object Classification with Hierarchical Radial Basis Function Networks

2001-03  Hans A. Kestler, Friedhelm Schwenker, Günther Palm
RBF network classification of ECGs as a potential marker for sudden cardiac death

2001-04  Christian Dietrich, Friedhelm Schwenker, Klaus Riede, Günther Palm
Classification of Bioacoustic Time Series Utilizing Pulse Detection, Time and Frequency Features and Data Fusion

2002-01  Stefanie Rinderle, Manfred Reichert, Peter Dadam
Effiziente Verträglichkeitsprüfung und automatische Migration von Workflow-Instanzen bei der Evolution von Workflow-Schemata

2002-02  Walter Guttmann
Deriving an Applicative Heapsort Algorithm

2002-03  Axel Dold, Friedrich W. von Henke, Vincent Vialard, Wolfgang Goerigk
A Mechanically Verified Compiling Specification for a Realistic Compiler

2003-01  Manfred Reichert, Stefanie Rinderle, Peter Dadam
A Formal Framework for Workflow Type and Instance Changes Under Correctness Checks

2003-02  Stefanie Rinderle, Manfred Reichert, Peter Dadam
Supporting Workflow Schema Evolution By Efficient Compliance Checks
2003-03  Christian Heinlein
Safely Extending Procedure Types to Allow Nested Procedures as Values

2003-04  Stefanie Rinderle, Manfred Reichert, Peter Dadam
On Dealing With Semantically Conflicting Business Process Changes.

2003-05  Christian Heinlein
Dynamic Class Methods in Java

2003-06  Christian Heinlein
Vertical, Horizontal, and Behavioural Extensibility of Software Systems

2003-07  Christian Heinlein
Safely Extending Procedure Types to Allow Nested Procedures as Values
(Corrected Version)

2003-08  Changling Liu, Jörg Kaiser
Survey of Mobile Ad Hoc Network Routing Protocols

2004-01  Thom Frühwirth, Marc Meister (eds.)
First Workshop on Constraint Handling Rules

2004-02  Christian Heinlein
Concept and Implementation of C+++, an Extension of C++ to Support User-Defined
Operator Symbols and Control Structures

2004-03  Susanne Biundo, Thom Frühwirth, Günther Palm (eds.)
Poster Proceedings of the 27th Annual German Conference on Artificial Intelligence

2005-01  Armin Wolf, Thom Frühwirth, Marc Meister (eds.)
19th Workshop on (Constraint) Logic Programming

2005-02  Wolfgang Lindner (Hg.), Universität Ulm, Christopher Wolf (Hg.) KU Leuven
2. Krypto-Tag – Workshop über Kryptographie, Universität Ulm

2005-03  Walter Guttmann, Markus Maucher
Constrained Ordering

2006-01  Stefan Sarstedt
Model-Driven Development with ACTIVECHARTS, Tutorial

2006-02  Alexander Raschke, Ramin Tavakoli Kolagari
Ein experimenteller Vergleich zwischen einer plan-getriebenen und einer
leichtgewichtigen Entwicklungsmethode zur Spezifikation von eingebetteten
Systemen

2006-03  Jens Kohlmeyer, Alexander Raschke, Ramin Tavakoli Kolagari
Eine qualitative Untersuchung zur Produktlinien-Integration über
Organisationsgrenzen hinweg

2006-04  Thorsten Liebig
Reasoning with OWL - System Support and Insights –

2008-01  H.A. Kestler, J. Messner, A. Müller, R. Schuler
On the complexity of intersecting multiple circles for graphical display
2008-02  Manfred Reichert, Peter Dadam, Martin Jurisch, Ulrich Kreher, Kevin Göser, Markus Lauer
Architectural Design of Flexible Process Management Technology

2008-03  Frank Raiser
Semi-Automatic Generation of CHR Solvers from Global Constraint Automata

2008-04  Ramin Tavakoli Kolagari, Alexander Raschke, Matthias Schneiderhan, Ian Alexander
Entscheidungsdokumentation bei der Entwicklung innovativer Systeme für produktlinien-basierte Entwicklungsprozesse

2008-05  Markus Kalb, Claudia Dittrich, Peter Dadam
Support of Relationships Among Moving Objects on Networks

2008-06  Matthias Frank, Frank Kargl, Burkhard Stiller (Hg.)
WMAN 2008 – KuVS Fachgespräch über Mobile Ad-hoc Netzwerke

2008-07  M. Maucher, U. Schöning, H.A. Kestler
An empirical assessment of local and population based search methods with different degrees of pseudorandomness

2008-08  Henning Wunderlich
Covers have structure

2008-09  Karl-Heinz Niggl, Henning Wunderlich
Implicit characterization of FPTIME and NC revisited

2008-10  Henning Wunderlich
On span-P^{cc} and related classes in structural communication complexity

2008-11  M. Maucher, U. Schöning, H.A. Kestler
On the different notions of pseudorandomness

2008-12  Henning Wunderlich
On Toda’s Theorem in structural communication complexity

2008-13  Manfred Reichert, Peter Dadam
Realizing Adaptive Process-aware Information Systems with ADEPT2

2009-01  Peter Dadam, Manfred Reichert
The ADEPT Project: A Decade of Research and Development for Robust and Flexible Process Support
Challenges and Achievements

2009-02  Peter Dadam, Manfred Reichert, Stefanie Rinderle-Ma, Kevin Göser, Ulrich Kreher, Martin Jurisch
2009-03  Alena Hallerbach, Thomas Bauer, Manfred Reichert  
Correct Configuration of Process Variants in Provop  

2009-04  Martin Bader  
On Reversal and Transposition Medians  

2009-05  Barbara Weber, Andreas Lanz, Manfred Reichert  
Time Patterns for Process-aware Information Systems: A Pattern-based Analysis  

2009-06  Stefanie Rinderle-Ma, Manfred Reichert  
Adjustment Strategies for Non-Compliant Process Instances  

Statistical Computing 2009 – Abstracts der 41. Arbeitstagung  

2009-08  Ulrich Kreher, Manfred Reichert, Stefanie Rinderle-Ma, Peter Dadam  
Effiziente Repräsentation von Vorlagen- und Instanzdaten in Prozess-Management-Systemen  

2009-09  Dammertz, Holger, Alexander Keller, Hendrik P.A. Lensch  
Progressive Point-Light-Based Global Illumination  

2009-10  Dao Zhou, Christoph Müssel, Ludwig Lausser, Martin Hopfensitz, Michael Kühl, Hans A. Kestler  
Boolean networks for modeling and analysis of gene regulation  

2009-11  J. Hanika, H.P.A. Lensch, A. Keller  
Two-Level Ray Tracing with Recordering for Highly Complex Scenes  

2009-12  Stephan Buchwald, Thomas Bauer, Manfred Reichert  
Durchgängige Modellierung von Geschäftsprozessen durch Einführung eines Abbildungsmodells: Ansätze, Konzepte, Notationen  

2010-01  Hariolf Betz, Frank Raiser, Thom Frühwirth  
A Complete and Terminating Execution Model for Constraint Handling Rules  

2010-02  Ulrich Kreher, Manfred Reichert  
Speichereffiziente Repräsentation instanzspezifischer Änderungen in Prozess-Management-Systemen  

2010-03  Patrick Frey  
Case Study: Engine Control Application  

2010-04  Matthias Lohrmann und Manfred Reichert  
Basic Considerations on Business Process Quality  

2010-05  HA Kestler, H Binder, B Lausen, H-P Klenk, M Schmid, F Leisch (eds):  
Statistical Computing 2010 - Abstracts der 42. Arbeitstagung  

2010-06  Vera Künzle, Barbara Weber, Manfred Reichert  
Object-aware Business Processes: Properties, Requirements, Existing Approaches
2011-01  Stephan Buchwald, Thomas Bauer, Manfred Reichert
Flexibilisierung Service-orientierter Architekturen

2011-02  Johannes Hanika, Holger Dammertz, Hendrik Lensch
Edge-Optimized À-Trous Wavelets for Local Contrast Enhancement with Robust Denoising

2011-03  Stefanie Kaiser, Manfred Reichert
Datenflussvarianten in Prozessmodellen: Szenarien, Herausforderungen, Ansätze

2011-04  Hans A. Kestler, Harald Binder, Matthias Schmid, Friedrich Leisch, Johann M. Kraus (eds):

2011-05  Vera Künzle, Manfred Reichert
PHILharmonicFlows: Research and Design Methodology

2011-06  David Knuplesch, Manfred Reichert
Ensuring Business Process Compliance Along the Process Life Cycle

2011-07  Marcel Dausend
Towards a UML Profile on Formal Semantics for Modeling Multimodal Interactive Systems

2011-08  Dominik Gessenharter
Model-Driven Software Development with ACTIVECHARTS - A Case Study

2012-01  Andreas Steigmiller, Thorsten Liebig, Birte Glimm
Extended Caching, Backjumping and Merging for Expressive Description Logics

2012-02  Hans A. Kestler, Harald Binder, Matthias Schmid, Johann M. Kraus (eds):
Statistical Computing 2012 - Abstracts der 44. Arbeitstagung

2012-03  Felix Schüssel, Frank Honold, Michael Weber
Influencing Factors on Multimodal Interaction at Selection Tasks

2012-04  Jens Kolb, Paul Hübner, Manfred Reichert
Model-Driven User Interface Generation and Adaption in Process-Aware Information Systems

2012-05  Matthias Lohrmann, Manfred Reichert
Formalizing Concepts for Efficacy-aware Business Process Modeling*

2012-06  David Knuplesch, Rüdiger Pryss, Manfred Reichert
A Formal Framework for Data-Aware Process Interaction Models