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.

![Fig. 1. Violating realizability, message-determinism, clear termination [9]](image-url)

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
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 Trans.).

- 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 \( I = (\mathcal{R}, \mathcal{D}, C, \dom_C, s_C, r_C, \epsilon) \), 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,
- \( \dom_C : C \to \mathcal{D} \) is a function assigning to each message class a data domain,
- \( s_C : C \to \mathcal{R} \) assigns the sender to each message class,
- \( r_C : C \to 2^\mathcal{R} \) 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.
Based on Def. 1, Def. 2 introduces the notion of message. A message constitutes an instance of a message class. Furthermore, we introduce several sets of messages.

**Definition 2 (Messages).**

Let $\mathcal{I} = (\mathcal{R}, \mathcal{D}, C, \text{dom}_C, s_C, r_C, \epsilon)$ 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
- $x \in \text{dom}_C(c)$ is the message content transferred.

Furthermore, we define:

- $\Sigma_C := \{(c', x) \in C \times \Omega_\mathcal{I} \mid c' = c \land x \in \text{dom}_C(c')\}$ 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 $\mathcal{I}$,
- $\Sigma_{\rightarrow R} := \{(c, v) \in \Sigma_\mathcal{I} \mid s_C(c) = R\}$ as set of all messages sent by role $R \in \mathcal{R}$,
- $\Sigma_{\leftarrow R} := \{(c, v) \in \Sigma_\mathcal{I} \mid R \in r_C(c)\}$ as set of all messages received by role $R$,
- $\Sigma_R := \Sigma_{\rightarrow R} \cup \Sigma_{\leftarrow R}$ 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 $\mathcal{I} = (\mathcal{R}, \mathcal{D}, C, \text{dom}_C, s_C, r_C, \epsilon)$ with:

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

<table>
<thead>
<tr>
<th>$s_C(\text{Request PET})$</th>
<th>$s_C(\text{Confirmation})$</th>
<th>$s_C(\text{Request Trans.})$</th>
<th>$s_C(\text{Request Exam.})$</th>
<th>$s_C(\text{Result})$</th>
<th>$s_C(\text{Retransport})$</th>
<th>$s_C(\text{Return})$</th>
<th>$s_C(\text{Confirmation+})$</th>
<th>$r_C(\text{Request PET})$</th>
<th>$r_C(\text{Confirmation})$</th>
<th>$r_C(\text{Request Trans.})$</th>
<th>$r_C(\text{Request Exam.})$</th>
<th>$r_C(\text{Result})$</th>
<th>$r_C(\text{Retransport})$</th>
<th>$r_C(\text{Return})$</th>
<th>$r_C(\text{Confirmation+})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hospital</td>
<td>PET provider</td>
<td>PET provider</td>
<td>Hospital</td>
<td>PET provider</td>
<td>Hospital</td>
<td>Hospital</td>
<td>Hospital</td>
<td>PET provider</td>
<td>Hospital</td>
<td>PET provider</td>
<td>Hospital</td>
<td>Hospital</td>
<td>Hospital</td>
<td>Hospital</td>
<td>Hospital</td>
</tr>
</tbody>
</table>

$\Sigma_\mathcal{I} = \{(\text{Request PET, uncritical}), (\text{Request PET, critical}), (\text{Result, abort}), (\text{Result, continue}), (\text{Request Exam., } \epsilon), (\text{Arrival, } \epsilon), (\text{Confirmation, 1.1.1900}), \ldots, (\text{Confirmation, 31.12.2099}), (\text{Confirmation+, 1.1.1900}), \ldots, (\text{Confirmation+, 31.12.2099}), (\text{Retransport, } \epsilon), (\text{Return, } \epsilon), (\text{Request Trans., 1.1.1900}), \ldots, (\text{Request Trans., 31.12.2099}), (\text{Cancel PET, } \epsilon)\}$
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 \( I = (R, D, C, \text{dom}_C, s_C, r_C, \epsilon) \) be an interaction domain. Then: A Data-Aware Choreography (DAChor) over \( I \) is a tuple \( \text{DAC} = (N, I, G, e_s, E_e, G^s_+, G^m_+, G^s_{dx}, G^s_{ex}, G^m_{dx}, V, \text{class}, \rightarrow, \cdot, \cdot, \text{dom}_V, \text{grd}) \), 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 \( G^s_+ \), the set of AND-mergers \( G^m_+ \), the set of data-based XOR-splits \( G^s_{dx} \), the set of event-based XOR-splits \( G^s_{ex} \), and the set of XOR-mergers \( G^m_{dx} \),
- \( V \) is the set of virtual data objects,
- \( \text{class} : I \rightarrow C \) assigns a message class to each interaction,
- \( \rightarrow \in (N - E_e) \times (N - \{e_s\}) \) is the interaction flow relation,
- \( \cdot \in I \times V \) is the data assignment relation,
- \( \text{dom}_V : V \rightarrow D \) is a function assigning a domain to each virtual data object,
- \( \text{grd} : (\rightarrow) \rightarrow G_V \) is a function assigning a guard to each interaction flow.

<table>
<thead>
<tr>
<th>A DAChor is well-formed, iff the following properties hold:</th>
</tr>
</thead>
<tbody>
<tr>
<td>- the start event, each interaction, and each merge node have exactly one successor, i.e.,</td>
</tr>
<tr>
<td>( \forall n \in {e_s} \cup I \cup G^m_+ \cup G^s_+ : {n' \in N</td>
</tr>
<tr>
<td>- each split node has at least one successor, i.e.,</td>
</tr>
<tr>
<td>( \forall g^s \in G^s_+ \cup G^s_{dx} \cup G^s_{ex} : {n \in N</td>
</tr>
<tr>
<td>- each end event, each interaction, and each split node have exactly one predecessor, i.e.,</td>
</tr>
<tr>
<td>( \forall n \in E_e \cup I \cup G^s_+ \cup G^s_{dx} \cup G^s_{ex} : {n' \in N</td>
</tr>
<tr>
<td>- each merge node has at least one predecessor, i.e.,</td>
</tr>
<tr>
<td>( \forall g^m \in G^m_+ \cup G^m_{dx} : {n \in N</td>
</tr>
<tr>
<td>- each event-based XOR-split is solely followed by interactions, i.e.,</td>
</tr>
<tr>
<td>( \forall g^s_{ex} \in G^s_{ex} : {n \in N</td>
</tr>
<tr>
<td>- guards of interaction flows are constantly true unless the source of an interaction flow is a data-based XOR-split, i.e.,</td>
</tr>
<tr>
<td>( \text{grd}((n_1, n_2)) = \text{true} \iff n_1 \in G^s_{dx} )</td>
</tr>
<tr>
<td>- the data of an interaction is solely assigned to variables of the same data domain, i.e.,</td>
</tr>
<tr>
<td>( \forall i \in I, \forall v \in V : i \rightarrow v \Rightarrow \text{dom}_C(\text{class}(i)) = \text{dom}_V(v) )</td>
</tr>
<tr>
<td>- there is no cycle solely consisting of gateways, i.e.,</td>
</tr>
<tr>
<td>( \exists g_0, g_1, \ldots, g_n \in G : g_0 \rightarrow g_1 \rightarrow \cdots \rightarrow g_n \rightarrow g_0 ).</td>
</tr>
</tbody>
</table>
Example 2 (DAChor).
Consider the scenario from Fig. 2. Basing its interaction domain \( \mathcal{I} \) (cf. Example 1) we can describe the given scenario as DAChor DACH = \((N, I, G, e_8, E_c, G^s, G^m, G^s_d, G^m_e, G^s_d, G^m_e, V, \text{class}, \to, \to, \text{dom}_V, \text{grd})\):

\[
I = \{i_1, \ldots, i_8\}, E_c = \{e_1^c, e_2^c\}, V = \{\text{Status, Order}\}
\]

\[
G^s_d = \{g^s_d_1, g^s_d_2\}, G^m_e = \{g^m_e_1, g^m_e_2\}, G^s_d = G^m_e = G^s = G^m = \varnothing,
\]

\[
\Rightarrow \{(i_1, \text{Status}, (i_2, \text{Order})\}
\]

\[
\Rightarrow \{(e_1, i_1), (i_1, i_2), (i_2, i_3), (i_3, g^s_d_1), (g^s_d_1, i_4), (g^s_d_1, g^m_2), (i_4, i_5), (i_5, g^s_d_2), (g^s_d_2, g^m_2), (g^m_2, i_6)\}
\]

\[
\text{class}(i_1) = \text{Request PET} \quad \text{class}(i_2) = \text{Confirmation} \quad \text{class}(i_3) = \text{Request Trans.}
\]

\[
\text{class}(i_4) = \text{Request Exam.} \quad \text{class}(i_5) = \text{Result} \quad \text{class}(i_6) = \text{Arrival}
\]

\[
\text{class}(i_7) = \text{Retransport} \quad \text{class}(i_8) = \text{Return}
\]

\[
\text{dom}_V(\text{Status}) = D_{\text{Status}} \quad \text{dom}_V(\text{Order}) = D_{\text{Order}}
\]

\[
\text{grd}((g^s_d_1, i_4)) = \text{Status} = \text{critical} \quad \text{grd}((g^s_d_1, g^m_2)) = \text{Status} = \text{uncritical}
\]

\[
\text{grd}((g^s_d_2, e_1^c)) = \text{Order} = \text{abort} \quad \text{grd}((g^s_d_2, g^m_2)) = \text{Order} = \text{continue}
\]

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 \( \mathcal{I} = (\mathcal{R}, \mathcal{D}, C, \text{dom}_C, s_C, r_C, \epsilon) \) be an interaction domain. Then, a Data-Aware Interaction Net (DAI Net) over \( \mathcal{I} \) is a tuple \( \# = (P, p_{in}, P_o, P_f, T, T_S, T_I, V, \text{class}, \to, \to, \text{dom}_V, \text{grd}) \), 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,
- \( \text{dom}_V: V \to \mathcal{D} \) is a function assigning a data domain to each variable,
- \( \text{grd}: T \to \mathcal{G}_V \) is a function assigning a guard to each interaction flow relation.

Further, we define

- \( \Sigma_{\#} := \cup_{t \in \mathcal{T}_I} \Sigma_{\text{class}(i)} \) as the set of all messages corresponding to \( \# \)
- \( P^{\to} := \{p \in P | p \to t\} \) as the set of all places preceding \( t \)
- \( P^{\to} := \{p \in P | t \to p\} \) as the set of all places succeeding \( t \)
- \( P^{\text{far}} := \{p \in P | p \to t \land t \to p\} \) as the set of the faraway places of \( t \)
As below Def. 3, the set of guards $G_V$ 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 $s_1, s_2, \ldots, s_n \in dom_V(v)$ are values of the related data domain. If a guard $g \in G_V$ uses a variable $v \in V$, we denote this as $v \vdash g$. 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.,
  $\forall t_i \in T_I, \forall v \in V : t_i \rightarrow v \Rightarrow dom_C(class(t_i)) = dom_V(v)$.
- 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 $DAC = (N, I, G, e_s, E_c, G_s^{+}, G_m^{+}, G_d^{+}, G_{ex}^{+}, G_{x}^{+}, V, class, \rightarrow, \star, dom_V, grd)$ be a DACHor (cf. Def. 3). Then, DAC can be mapped to a DAI Net defined as $\#: (P, p_{in}, P_o, T, T_I, T_S, V, class', \rightarrow', \star', dom_V, grd')$, with

- $P := \{ p(n_1, n_2) | (n_1, n_2) \in \rightarrow \land n_1 \not\in G_{ex}^{+} \}$ interaction flow
- $p_{in} := p(e_s, n) \in P$, whereby $e_s \rightarrow n \in N$ start event
- $P_o := \{ p(n, e_e) | p(n, e_e) \in P \land e_e \in E_e(i) \in P \}$ end events
- $T := T_I \cup T_S \cup T_{m'}$ interactions
- $T_I := \{ t_i | i \in I \}$ AND gateways
- $T_S := T_{x'} \cup T_{s'}$ XOR-merge gateways
- $T_{m'} := \{ t_{m'}(n, g_s^{+}) | g_s^{+} \in G_{x}^{+} \land n \not\in N \land n \rightarrow g_s^{+} \}$ XOR-split gateways
- $class'(t_i) := class(i)$ message class assignment
- $\rightarrow' := \{ (p(n_1, n_2), t_{n_2}) | n_1 \rightarrow n_2 \land n_1 \not\in G_{ex}^{+} \land n_2 \in I \cup G_s^{+} \land G_{m'}^{+} \}$ interactions/AND-gateways in
- $\cup \{ (t_{n_1}, p(n_1, n_2)) | n_1 \rightarrow n_2 \land n_1 \in I \cup G_s^{+} \land G_{m'}^{+} \}$ interactions/AND-gateways out
- $\cup \{ (t_{m'}(n_1, n_2), p(n_1, n_2)) | n_0 \rightarrow n_1 \rightarrow n_2 \land n_1 \in G_{m'}^{+} \}$ XOR-merge in
- $\cup \{ t_{m'}(n_1, n_2), P(o(n_1, n_2)) | n_1 \rightarrow n_2 \land n_2 \in G_{m'}^{+} \}$ XOR-merge out
- $\cup \{ (p(n_1, n_2), t_{n_1}) | n_1 \rightarrow n_2 \land n_1 \in G_{ex}^{+} \}$ data-based XOR-split in
- $\cup \{ t_{m'}(n_1, n_2), P(o(n_1, n_2)) | n_1 \rightarrow n_2 \land n_1 \in G_{ex}^{+} \}$ data-based XOR-split out
- $\cup \{ (p(n_1, n_2), t_{n_1}) | n_1 \rightarrow n_2 \land n_1 \rightarrow G_{ex}^{+} \}$ event-based XOR-split
- $\cup \{ (p(n_1, n_2), t_{n_2}) | n_1 \rightarrow n_2 \land n_1 \rightarrow G_{ex}^{+} \}$ data assignment relation
- $\cup \{ (t_{m'}(n_1, n_2), t_{n_2}) | n_1 \rightarrow n_2 \land n_1 \rightarrow G_{ex}^{+} \}$ 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 DAChor 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 (cf. Proof 1):

Proof 1 (Theorem 1: Preservation of Well-Formedness (Property 1)).
Let DAC = (N, I, G, ε, E, G^s_e, G^s_m, G^{dx}_e, G^{dx}_m, V, class, →, →', domV, grd) be a well-formed DAChor. 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', →', →', domV, grd').

Case 1: t = t_{g_p} ∈ T^t := \{t_{g_p}\mid g_p ∈ G^s_p \cup G^m_p\}
⇒ \exists n_1, n_2 \in N : n_1 \rightarrow g_{p} \rightarrow n_2
⇒ \exists p(n_1, g_p), p(n_2, g_p) ∈ P : (p(n_1, g_p), t_{g_p}), (t_{g_p}, p(n_2, g_p)) \in →'
i.e., p(n_1, g_p) \rightarrow' t \rightarrow' p(n_2, g_p).

Case 2: t = t_{g_x} ∈ T^t := \{t_{g_x}\mid g_x ∈ G^s_x \land n_1 \in N \land g_x \rightarrow n_1\}
⇒ \exists n_1, n_2 \in N : n_1 \rightarrow g_x \rightarrow n_2
⇒ \exists p(n_1, g_x), p(n_2, g_x) ∈ P : (p(n_1, g_x), t_{g_x}), (t_{g_x}, p(n_2, g_x)) \in →'
i.e., p(n_1, g_x) \rightarrow' t \rightarrow' p(n_2, g_x).

Case 3: t = t_{g_x} ∈ T^t := \{t_{g_x}\mid g_x ∈ G^s_x \land n_1 \in N \land n_1 \rightarrow g_x \rightarrow n_2\}
⇒ \exists n_1, n_2 \in N : n_2 \rightarrow g_x \rightarrow n_1
⇒ \exists p(n_1, g_x), p(n_2, g_x) ∈ P : (p(n_1, g_x), t_{g_x}), (t_{g_x}, p(n_2, g_x)) \in →'
i.e., p(n_1, g_x) \rightarrow' t \rightarrow' p(n_2, g_x).

Case 4: t = t_{v} ∈ T_I := \{t_i\mid i ∈ I\}
⇒ \exists n_1, n_2 \in N : n_1 \rightarrow i \rightarrow n_2

Subcase 4.1: n_1 \in G^s_{ex}
⇒ \exists n_0 \in N \rightarrow G^s_{ex} : n_0 \rightarrow n_1
⇒ \exists p(n_0, n_1), (i, n_2) : (p(n_0, n_1), t_i), (t_i, p(i, n_2)) \in →'
i.e., p(n_0, n_1) \rightarrow' t \rightarrow' p(i, n_2).

Subcase 4.2: n_1 \notin G^s_{ex}
⇒ \exists p(n_0, i), (i, n_2) : (p(n_0, i), i), (i, p(i, n_2)) \in →'
i.e., p(n_0, i) \rightarrow' t \rightarrow' p(i, n_2).

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 (cf. Proof 2):

Proof 2 (Theorem 1: Preservation of Well-Formedness (Property 2)).
Let DAC = (N, I, G, ε, E, G^s_e, G^{dx}_e, G^s_m, G^{dx}_m, V, class, →, →', domV, grd) be a well-formed DAChor. 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', →', →', domV, grd').

Let t_i \in T_I be an interaction transition and v ∈ V a virtual data object of #. Then:
\( (t_i, v) \in →' \Rightarrow (i, v) \in → \)
⇒ domC(class(t_i)) = domC(class(i)) = domV(v)

Thus, data assignments are correct, consequently the second property holds.
Finally, Proof. 4 proves the third property, i.e., there exists no cycle solely consisting of places and silent transitions. For this purpose, we define a function gate, which assigns to each silent transition in $\#$ 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 (cf. Lemma 1 and Proof 3).

**Definition (Gate of a Silent Transition).**

Let $DAC = (N, I, G, e_s, E_e, E_v, G_s^m, G_v^m, G_d^x, G_v^x, G_s^x, G_v^x, V, class, \to, \to, dom_v, grd)$ be a well-formed DAC. DAC is mapped to a DAI Net $\# = (P, p_{in}, P_o, P_{fi}, T, T_S, T_I, V, class', \to', \to', dom_v, grd')$. Then:

The function $\text{gate} : T_S \to (G - G_s^x) : t_s \mapsto \text{gate}(t_s)$ assigns to each silent transition its gate, with

$$\text{gate}(t_s) := \begin{cases} g_s^+ & \text{iff } t = t_{g_s^+} \in T_s = \{t_{g_s^+}, g_s^+ \in G_s^x \cup G_v^m \} \\ g_s^x & \text{iff } t \in T_{s} := \{ t_{g_s^x, n_1} \mid g_s^x \in G_s^x \land n_1 \in N \land g_s^x \to n_1 \} \\ g_v^m & \text{iff } t \in T_{m} := \{ t_{m, g_v^m, n_1} \mid g_v^m \in G_v^m \land n_1 \in N \land n_1 \to g_v^m \} \end{cases}$$

**Lemma 1 (Connected Silent Transitions imply Connected Gateways).**

Let $DAC = (N, I, G, e_s, E_e, E_v, G_s^m, G_v^m, G_d^x, G_v^x, G_s^x, G_v^x, V, class, \to, \to, dom_v, grd)$ be a well-formed DAC. DAC is mapped to a DAI Net $\# = (P, p_{in}, P_o, P_{fi}, T, T_S, T_I, V, class', \to', \to', dom_v, grd')$. If there are two silent transitions $t_s^1, t_s^2 \in T_s$ that are connected by a place $p(n_1, n_2) \in P$ (cf. Def.7), i.e., $t_s^1 \to p \to t_s^2$. Then holds $\text{gate}(t_s^1) \rightarrow \text{gate}(t_s^2)$ holds.

**Proof 3 (Lemma 1): Connected Silent Transitions imply Connected Gateways.**

Let $DAC = (N, I, G, e_s, E_e, G_s^m, G_v^m, G_d^x, G_v^x, G_s^x, G_v^x, V, class, \to, \to, dom_v, grd)$ be a (well-formed) DAC that is mapped to a DAI Net $\#$. The latter is defined as $\# = (P, p_{in}, P_o, P_{fi}, T, T_S, T_I, V, class', \to', \to', dom_v, grd')$:

**Case 1:** $t_s^1 = t_{g_s^+} \in T_s$ and $g_s := \text{gate}(t_s^1) = G_s^x \cup G_v^m$

Subcase 1.1: $t_s^2 = t_{g_s'} \in T_s$ and $g_s' := \text{gate}(t_s^2) \in G_s^x \cup G_v^m$

$\Rightarrow (t_{g_s^+}, p(a, b)), (p(a, b), t_{g_s'}) \in (G_v^m) \Rightarrow a = g_s + b = g_s' \Rightarrow g_s \rightarrow g_s'$

Subcase 1.2: $t_s^2 = t_{m, g_v^m, n_1} \in T_s$ and $g_v^m := \text{gate}(t_s^2) \in G_v^m$

$\Rightarrow (t_{g_s^+}, p(a, b)), (p(a, b), t_{m, g_v^m, n_1}) \in (G_v^m) \Rightarrow a = g_s + b = g_v^m \Rightarrow g_s \rightarrow g_v^m$

**Case 2:** $t_s^1 = t_{g_s^x, n_1} \in T_s$ and $g_s^x := \text{gate}(t_s^1) \in G_s^x$

Subcase 2.1: $t_s^2 = t_{g_s'} \in T_s$ and $g_s' := \text{gate}(t_s^2) \in G_s^x$

$\Rightarrow (t_{g_s^x, n_1}, p(a, b)), (p(a, b), t_{g_s'}) \in (G_s^x) \Rightarrow a = g_s^x + b = n_1 = g_s' \Rightarrow g_s^x \rightarrow g_s'$

Subcase 2.2: $t_s^2 = t_{m, g_v^m, n_1} \in T_s$ and $g_v^m := \text{gate}(t_s^2) \in G_v^m$

$\Rightarrow (t_{g_s^x, n_1}, p(a, b)), (p(a, b), t_{m, g_v^m, n_1}) \in (G_v^m) \Rightarrow a = g_s^x + b = n_1 = g_v^m \Rightarrow g_s^x \rightarrow g_v^m$

Subcase 2.3: $t_s^2 = t_{m, g_v^m, n_1} \in T_s$ and $g_v^m := \text{gate}(t_s^2) \in G_v^m$

$\Rightarrow (t_{g_s^x, n_1}, p(a, b)), (p(a, b), t_{m, g_v^m, n_1}) \in (G_v^m) \Rightarrow a = g_s^x + b = n_1 = g_v^m \Rightarrow g_s^x \rightarrow g_v^m$

**Case 3:** $t_s^1 = t_{m, g_v^m, n_1} \in T_s$ and $g_v^m := \text{gate}(t_s^1) \in G_v^m$

Subcase 3.1: $t_s^2 = t_{g_s'} \in T_s$ and $g_s' := \text{gate}(t_s^2) \in G_s^x$

$\Rightarrow (t_{m, g_v^m, n_1}, p(a, b)), (p(a, b), t_{g_s'}) \in (G_s^x) \Rightarrow a = g_v^m + b = g_s' \Rightarrow g_v^m \rightarrow g_s'$
Subcase 3.2: $t^2_s = t^2_{g'_n,n_2}' \in T^s_x$ and $g^s' := \text{gate}(t^2_s) \in G^n_x \times$
\implies (t^m_{(n_1,g^n_2)},p_{(a,b)})$ or $(p_{(a,b)},t^m_{(n_2,g^n_2)})$ \implies $a = g^m_{n_1} \land b = g^m_{n_2} \implies g^m_{n_1} \rightarrow g^m_{n_2}$

Subcase 3.3: $t^2_s = t^m_{(n_2,g^n_2)}' \in T^m_x$ and $g^m' := \text{gate}(t^2_s) \in G^m_x$
\implies (t^m_{(n_1,g^n_2)},p_{(a,b)})$ or $(p_{(a,b)},t^m_{(n_2,g^n_2)})$ \implies $a = g^m_{n_1} \land b = g^m_{n_2} \implies g^m_{n_1} \rightarrow g^m_{n_2}$

Thus, for all cases $\text{gate}(t^1_s) \rightarrow \text{gate}(t^2_s)$ holds and Lemma 1 is proven. □

We use function gate and Lemma 1 to prove the third property of well-formedness of DAI Nets by contradiction:

**Proof 4 (Theorem 1: Preservation of Well-Formedness (Property 3)).**
Assume property 3 of Theorem 1 is violated. Then: There exists a well-formed DAC = $(N, I, G, e_s, E_e, G^s_e, G^m_e, G^s_{ex}, G^m_{ex}, V, \text{class}, \rightarrow, \rightarrow, \text{dom}_V, \text{ grd})$ that is mapped to a non-well-formed DAI Net #, with # = $(P,p_{in},P_o,P_{fi},T,T_S,T_I,V,\text{class}', \rightarrow', \rightarrow', \text{ dom}_V, \text{ grd}')$:
\implies \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 t_0 \rightarrow p_0
\implies \text{gate}(t_0) \rightarrow \text{gate}(t_1) \rightarrow \cdots \rightarrow \text{gate}(t_n) \rightarrow \text{gate}(t_0)
\implies \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. □

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

**Example 3 (Transformation).**
The DAChor DAC = $(N, I, G, e_s, E_e, G^s_e, G^m_e, G^s_{ex}, G^m_{ex}, V, \text{class}, \rightarrow, \rightarrow, \text{ dom}_V, \text{grd})$ from Example 2 is mapped to the DAI Net # = $(P,p_{in},P_o,P_{fi},T,T_S,T_I,V,\text{class}', \rightarrow', \rightarrow', \text{ dom}_V, \text{grd}')$ as follows (cf. Fig. 4):

$P = \{p_{in} = p_{(e_s,i_1)}, p_{(i_1,i_2)}, p_{(i_2,i_3)}, p_{(i_3,i_4)}, p_{(g^s_{i_2},i_4)}, p_{(g^m_{i_2},i_4)}, p_{(i_4,i_5)}, p_{(i_5,i_6)}, p_{(g^s_{i_5},i_6)}, p_{(g^m_{i_5},i_6)}\}$
$P_{fi} = \{p_{(g^s_{i_2},i_5)}, p_{(i_5,i_6)}\}$
$T^s_x = \{t^s_{(g^s_{i_2},i_4)}, t^s_{(g^s_{i_2},i_4)}, t^s_{(g^s_{i_2},i_4)}\}$
$T^m_x = \{t^m_{(g^m_{i_2},i_4)}, t^m_{(g^m_{i_2},i_4)}\}$
$T_s = T_+ \cup T^s_x \cup T^m_x$
$T_I = \{t_{i_1}, t_{i_2}, \ldots, t_{i_6}\}$
$V = \{\text{Status, Order}\}$
\implies \text{class}'(t_{i_1}) = \text{Request\ PET}$
\implies \text{class}'(t_{i_2}) = \text{Confirmation}$
\implies \text{class}'(t_{i_3}) = \text{Result}$
\implies \text{class}'(t_{i_4}) = \text{Arrival}$
\implies \text{dom}_V(\text{Status}) = \text{D}_{\text{Status}}$
\implies \text{dom}_V(\text{Order}) = \text{D}_{\text{Order}}$
\implies \text{grd}(t^s_{(g^s_{i_2},i_4)}) = \text{Status = critical}$
\implies \text{grd}(t^s_{(g^m_{i_2},i_4)}) = \text{Status = uncritical}$

**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 \( P_{f_i} \). 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 \( \# = (P, p_{in}, P_o, P_{f_i}, T, T_S, T_I, V, \text{class}, \text{→}, \text{→}, \text{dom}_V, \text{grd}) \) be a DAI Net. Then: A marking of \( \# \) is a tuple \( m = (\varnothing, \text{val}) \) with

- \( \varnothing : P \rightarrow \mathbb{N}_0 \) assigns to each place the number of corresponding tokens,
- \( \text{val} : V \rightarrow \Omega_T \) 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 \( M_\# \), whereby
  \( M_\# := \{m = (\varnothing, \text{val}) \mid m \text{ is a marking of } \# \} \)
- initial marking \( m_{in}^\# := (\varnothing_{in}, \text{val}_{in}) \in M_\#, \) whereby
  \( \varnothing_{in}(p) := \begin{cases} 1, & \text{if } p = p_{in} \\ 0, & \text{else} \end{cases} \quad \forall v \in V : \text{val}_{in}(v) := \epsilon \)

- set of all final markings \( F_\# \), whereby
  \( F_\# := \{(\varnothing, \text{val}) \in M_\# \mid \forall p \in P : \varnothing(p) \neq 0 \iff p \in P_{f_i}\} \)
- transition activation relation \( \trianglerighteq \in M_\# \times T \). \( m \trianglerighteq t \) denotes that marking \( m \in M_\# \) activates transition \( t \in T \), iff the following conditions hold:
  1) \( \forall p \in P^{-t} : \varnothing(p) \geq 1 \),
  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 \( \# = (P,p_{in},P_o,P_i,T,T_S,T_I,V,\text{class},\to,\to^*,\text{dom}_V,\text{grd}) \) be a DAI Net, an \( m = (\emptyset,\text{val}) \), \( m' = (\emptyset',\text{val}') \in \mathcal{M}_\# \) be two related markings. Then:

- \( \mathcal{O}_\# := T_S \cup \Sigma_\# \) is the set of all options on \( \# \).
- \( \text{opt}_\# : \mathcal{M}_\# \rightarrow 2^{\mathcal{O}_\#} : m \mapsto \{ o \in \mathcal{O}_\# | \exists m' \land m \stackrel{o}{\Rightarrow} m' \} \) 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:** \( o = t_\mu \in T_S \) is a silent transition. Then: \( m \stackrel{t_\mu}{\rightarrow} m' \) holds, iff each of the following conditions is met:

  1. \( m \sim t_\mu \)
  2. \( \forall p \in P^{\rightarrow t_\mu} : \circ'(p) = \circ(p) - 1 \),
  3. \( \forall p \in P^{\rightarrow t_\mu} : \circ'(p) = \circ(p) + 1 \),
  4. \( \forall p \in P^{^{\rightarrow t_\mu}} : \circ'(p) = \circ(p) \),
  5. \( \forall v \in V : \text{val}'(v) = \text{val}(v) \).

  **Case 2:** \( o = \mu = (c,x) \in \Sigma_\# \) is a message. Then: \( m \stackrel{\mu}{\rightarrow} m' \) holds, iff each of the following conditions are met:

  1. \( \forall t_\mu \in T_I : m \Rightarrow t_\mu \)
  2. \( \exists t_i \in T_I : m \sim t_i \land \mu \in \Sigma_{\text{class}(t_i)} \),
  3. \( \forall p \in P^{\rightarrow t_i} : \circ'(p) = \circ(p) - 1 \),
  4. \( \forall p \in P^{^{\rightarrow t_i}} : \circ'(p) = \circ(p) + 1 \),
  5. \( \forall p \in P^{^{\rightarrow t_i}} : \circ'(p) = \circ(p) \),
  6. \( \forall v \in V \) with \( t_i \Rightarrow v : \text{val}'(v) = x \),
  7. \( \forall v \in V \) with \( t_i \Rightarrow v : \text{val}'(v) = \text{val}(v) \).

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.,

\[
\forall m \in \mathcal{M}_\# : \text{opt}_\#(m) \neq \emptyset \quad \Rightarrow \quad \text{opt}_\#(m) \subseteq T_S \oplus \text{opt}_\#(m) \subseteq \Sigma_\#
\]

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

\[
\forall m \in \mathcal{F}_\# : \text{opt}_\#(m) = \emptyset
\]

We prove Theorem 2 and Theorem 3 by contradiction:

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

\[ \exists m \in \mathcal{M}_\# : \text{opt}_\#(m) \neq \emptyset \quad \text{and} \quad \exists t_0 \in T_S, \mu \in \Sigma_\# : t_0, \mu \in \text{opt}_\#(m) \]

\[ \exists m' \in \mathcal{M}_\# : m \rightarrow m' \quad \Rightarrow \quad \forall t_\mu \in T_S : m \Rightarrow t_\mu \Rightarrow m \Rightarrow t_0 \]

\[ t_0 \notin \text{opt}_\#(m) \]

This contradicts our assumption. Thus, Theorem 2 is proven. \( \square \)
Proof 6 (Theorem 3: Termination of final markings).
Let \( \# = (P, p_{in}, P_0, P_{fi}, T, T_S, T_I, V, \text{class}, \rightarrow, \rightarrow, \text{dom}_V, \text{grd}) \) be a DAI Net. Then: Assume, Theorem 3 is violated for \( \# \):

\[
\Rightarrow \exists m = (\odot, \text{val}) \in F_\#: \text{opt}_\#(m) \neq \emptyset
\]

\[
\Rightarrow \exists t \in T : m \xrightarrow{t} \Rightarrow \exists p \in P : \odot(p) \geq 1 \land p \xrightarrow{t}
\]

\[
p \notin P_{fi} \Rightarrow m \notin F_\#
\]

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 \( \# = (P, p_{in}, P_0, P_{fi}, T, T_S, T_I, V, \text{class}, \rightarrow, \rightarrow, \text{dom}_V, \text{grd}) \) be a DAI Net and \( \tau = (\tau_k)_{k \in [1..n]} \in \mathcal{O}_\#^* \) be a finite sequence of options (i.e. silent transitions and messages) with length \( |\tau| =: n \in \mathbb{N} \). Let further \( m = (m_k)_{k \in [1..n+1]} \in \mathcal{M}_\# \) be a finite sequence of markings with length \( n + 1 \). Then:

- \( \tau \sim m \) denotes that \( \tau \) and \( m \) are related sequences, iff \( \forall l \in [1..n] : m_l \xrightarrow{\tau_l} m_{l+1} \) and \( m_1 = m_{in} \).
- \( \text{last} : \mathcal{M}_\#^* \rightarrow \mathcal{M}_\# \) with \( (m_k)_{k \in [1..n]} \xrightarrow{m_n} \) is a function mapping a sequence of markings to its last marking.
- \( \tau \in \mathcal{O}_\#^* \) is a trace, iff \( \exists m \in \mathcal{M}_\#^* \) and \( \tau \sim m \). If last\((m)\) \( \in \mathcal{F}_\# \), we denote \( \tau \) as completed trace.
- \( \mathcal{T}_\# \) denotes the set of all traces on \( \# \).
- \( \mathcal{T}_\#^c \) denotes the set of all completed traces on \( \# \).

(B) Let \( L \subseteq \mathcal{M}_\#^* \) be a set of finite sequences over a set \( M \) and let \( a = (a_k)_{k \in [1..n]}, b = (b_k)_{k \in [1..l]}, c = (c_m)_{m \in [1..m]} \in L \) 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 \( n \leq l \) (\( n \leq l \)) 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 \( \forall i \in [1..m] : c_i = b_{n+i} \).
- \( L^{gb} := \{ a \in L | a \sqsubseteq b \} \) (\( L^{gb} := \{ a \in L | a \sqsubseteq b \} \)) denotes the subset of \( L \) that contains all prefixes (real prefixes) of \( b \in L \), and
- \( L^{gb} := \{ a \in L | b \sqsubseteq a \} \) (\( L^{gb} := \{ a \in L | b \sqsubseteq a \} \)) 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., \( \forall \tau \in \mathcal{T}_\#^c : |\{ m \in \mathcal{M}_\#^* | m \sim \tau \}| = 1. \)

Let \( \# \) be a deterministic DAI Net. Then:

- \( \text{mark}_\# \) maps each trace on \( \# \) to its current marking, i.e. the last marking of the related sequence of markings: \( \text{mark}_\#: \mathcal{T}_\# \rightarrow \mathcal{M}_\#: \tau \mapsto \text{mark}_\#(\tau) := \text{last}(m) \), whereby \( m \) is defined by \( \tau \sim : m \in \mathcal{M}_\#^* \).

Since \( \# \) is deterministic, the definition of \( m \) is unique. Thus, \( \text{mark}_\# \) is well defined.
We call a deterministic DAI Net sound, iff the following conditions hold:

- There exist completed traces on #, i.e., \( T_{\#}^c \neq \emptyset \).
- Each trace on # is a prefix of a completed trace, i.e., \( \forall \mathbf{v} \in T_{\#} \exists \tau \in T_{\#}^c : \mathbf{v} \subseteq \tau \).
- The set of reachable markings is finite, i.e., \(|\{m \in M_{\#} \mid \exists \tau \in T_{\#} : \text{last}(\tau) = m\}| \in \mathbb{N} \).

Note that the observable behavior of any DAI Net is solely explained through the messages exchanged. Hence, we must abstract from the silent elements of traces (i.e. silent transitions) and define the observable behavior as a conversation being the projection of a trace to its messages (i.e., the part of the trace defining its semantic). In the following, we first introduce projections of sequences.

**Definition 12** (Projections and Conversations).
Let \( A, B \) be two sets with \( B \subseteq A \), and \( \# = (P, p_{in}, P_o, P_{fi}, T, T_S, T_I, V, \text{class}, \rightarrow, \rightarrow, \text{dom}_V, \text{grd}) \) be a DAI Net and \( \tau \in T_{\#} \) be a trace on \#. Then:

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

\[
\begin{align*}
\tau_1 &= \langle \text{Request PET,uncritical},(\text{Confirmation,}1),(\text{Request Trans.,}1), t^s_{(g_{d1},g_{w})}, t^m_{(g_{d2},g_{w})}, \\
&\quad \text{(Arrival,}e), \text{(Retransport,}e), \text{(Return,}e) \rangle \\
\tau_2 &= \langle \text{Request PET,}critical,(\text{Confirmation,}1),(\text{Request Trans.,}1), t^s_{(g_{d1},i4)},(\text{Request Exam.,}e), \\
&\quad \text{(Result,}abort), t^s_{(g_{d2},e^2)} \rangle \\
\tau_3 &= \langle \text{Request PET,}critical,(\text{Confirmation,}1),(\text{Request Trans.,}1), t^s_{(g_{d1},i4)},(\text{Request Exam.,}e), \\
&\quad \text{(Result,}continue), t^s_{(g_{d2},g_{w})}, t^m_{(g_{d3},g_{w})}, (\text{Arrival,}e), (\text{Retransport,}e), (\text{Return,}e) \rangle \\
\eta_1 &= \text{con}_{\#}(\tau_1) := \Pi_{\#}(\tau_1) = \langle \text{Request PET,uncritical},(\text{Confirmation,}1),(\text{Request Trans.,}1), \\
&\quad \text{(Arrival,}e), (\text{Retransport,}e), (\text{Return,}e) \rangle \\
\eta_2 &= \text{con}_{\#}(\tau_2) := \Pi_{\#}(\tau_2) = \langle \text{Request PET,}critical,(\text{Confirmation,}1),(\text{Request Trans.,}1), \\
&\quad \text{(Result,}e), \text{(Result,}abort) \rangle \\
\eta_3 &= \text{con}_{\#}(\tau_3) := \Pi_{\#}(\tau_3) = \langle \text{Request PET,}critical,(\text{Confirmation,}1),(\text{Request Trans.,}1), \\
&\quad \text{(Result,}continue), \text{(Arrival,}e), (\text{Retransport,}e), (\text{Return,}e) \rangle 
\end{align*}
\]

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.

\[
\forall \tau \in \mathcal{T}_\#: \exists N \in \mathbb{N} \text{ such that } \forall v \in \mathcal{T}_\#^{\leq \tau} \text{ with } |\tau| + N < |v| \Rightarrow \text{con}_\#(\tau) \neq \text{con}_\#(v).
\]

To prove 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_0, P_f, T, T_S, T_I, V, class, →, →, dom_V, 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 \(p_0 \rightarrow t_0 \rightarrow p_1 \rightarrow t_1 \rightarrow \cdots \rightarrow p_n \rightarrow t_n\),
- \(\mathcal{W}_\#\) as the set of all silent ways on \#, and
- \(\mathcal{W}_\#^{p \rightarrow t} := \{\langle p_0, \ldots, t_n \rangle \in \mathcal{W}_\# | p_0 = p \land t_n = t\}\) 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** (Silent Ways Inequality).

Let \# = (P, p_{in}, P_0, P_f, T, T_S, T_I, V, class, →, →, dom_V, grd) be a deterministic and sound DAI Net and \(t, t' \in T_S\) be silent transitions. Then:

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

**Proof 7** (Lemma 2: Silent Ways Inequality).

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

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

\[
\Rightarrow \exists! \omega' = \langle p, t, q, t_0, p_1, t_1, \ldots, p_n, t_n \rangle \in \mathcal{W}_\#^{p \rightarrow t'}
\]

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

\[
p \in P^{\rightarrow t} \Rightarrow \bigcup_{q \in P^{\rightarrow t}} \mathcal{W}_\#^{q \rightarrow t'} \supseteq \mathcal{W}_\#^{p \rightarrow t'}
\]

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

Thus, Lemma 2 is proven. □

\(^1\)For reasons of simplification, we abstract from irrelevant message contents 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 $\# = (P, p_{in}, P_o, P_{f_1}, T, T_S, T_I, V, \text{class}, \rightarrow, \rightarrow, \text{dom}_V, \text{grd})$ be a deterministic and sound DAI Net. Then, the transition ranking function $\zeta$ and the net ranking function $\xi$ are defined as below:

- $\zeta : M_\# \times T_S \rightarrow \mathbb{N} : ((\emptyset, \text{val}), t) \mapsto \zeta(((\emptyset, \text{val}), t)$ is the transition ranking function that maps a marking and a silent transition to a natural number, with
  \[
  \zeta((\emptyset, \text{val}), t) := \sum_{p \in P} \text{oc}(p) \times |W_{\#}^{p \rightarrow t}|
  \]

- $\xi : M_\# \rightarrow \mathbb{N} : m \mapsto \xi(m)$ 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 $\# = (P, p_{in}, P_o, P_{f_1}, T, T_S, T_I, V, \text{class}, \rightarrow, \rightarrow, \text{dom}_V, \text{grd})$ be a deterministic and sound DAI Net. Further, $m, m' \in M_\#$ be two markings of $\#$, with $m = (\emptyset, \text{val}), m' = (\emptyset', \text{val}')$. 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:

\[
\xi(m) = \sum_{t \in T_S} \zeta(m, t) = \underbrace{\zeta(m, t_0)}_{\text{rank of fired silent transition}} + \sum_{t \in T_S \setminus \{t_0\}} \zeta(m, t) \underbrace{\zeta(m, t)}_{\text{sum of ranks of unfired silent transitions}}
\]

We consider both parts (i.e. $\zeta(m, t_0)$ and $\sum_{t \in T_S \setminus \{t_0\}} \zeta(m, t)$) on its own to show that $\xi$ decreases, whenever a silent transition $t_0$ is fired. First, Lemma 4 shows that $\zeta(m, t_0)$ decreases when firing $t_0$. Second, Lemma 5 shows that $\sum_{t \in T_S \setminus \{t_0\}} \zeta(m, t)$ does not increase, when $t_0$ is fired.

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

Let $\# = (P, p_{in}, P_o, P_{f_1}, T, T_S, T_I, V, \text{class}, \rightarrow, \rightarrow, \text{dom}_V, \text{grd})$ be a deterministic and sound DAI Net. Further, $m, m' \in M_\#$ be two markings of $\#$, with $m = (\emptyset, \text{val}), m' = (\emptyset', \text{val}')$. 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 (Lemma 4: The Rank of a Fired Silent Transition Decreases).

\[ \zeta(m, t_0) = \sum_{p \in P} \left( \delta(p) \cdot |W^m_{#p}| \right) = \sum_{p \in P^{t-}} \left( \delta(p) \cdot |W^m_{#p}^{p-t}| \right) + \sum_{p \in P^{t-}} \left( \delta(p) \cdot |W^m_{#p}^{p-}\right) + \sum_{p \in P^{t-}} \left( \delta(p) \cdot |W^m_{#p}^{p-}\right) \]

Thus, \( \zeta(m, t_0) > \zeta(m', t_0) \) holds and Lemma 4 is proven. □

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

Let \( # = (P, p_{int}, P_o, P_f, T, T_S, T_I, V, class, \rightarrow, \rightarrow, dom_V, grd) \) be a deterministic and sound DAI Net. Further, \( m, m' \in M_# \) be two markings of \( # \), with \( m = (\delta, val) \), \( m' = (\delta', val') \). Finally, \( t_0 \in T_S \) be a silent transition of \( # \) with \( m \xrightarrow{t_0} m' \). Then:

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

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

\[ \sum_{t \in T_S - \{t_0\}} \zeta(m, t) = \sum_{t \in T_S - \{t_0\}} \left( \sum_{p \in P} \left( \delta(p) \cdot |W^m_{#p}^{p-t}| \right) \right) \]

\[ = \sum_{t \in T_S - \{t_0\}} \left( \sum_{p \in P^{t-}} \left( \delta(p) \cdot |W^m_{#p}^{p-t}| \right) + \sum_{p \in P^{t-}} \left( \delta(p) \cdot |W^m_{#p}^{p-t}| \right) + \sum_{p \in P^{t-}} \left( \delta(p) \cdot |W^m_{#p}^{p-t}| \right) \right) \]

\[ = \sum_{t \in T_S - \{t_0\}} \left( \sum_{p \in P^{t-}} \left( \delta(p) \cdot |W^m_{#p}^{p-t}| \right) + \sum_{p \in P^{t-}} \left( \delta(p) \cdot |W^m_{#p}^{p-t}| \right) + \sum_{p \in P^{t-}} \left( \delta(p) \cdot |W^m_{#p}^{p-t}| \right) \right) \]

\[ \geq \sum_{t \in T_S - \{t_0\}} \left( \sum_{p \in P^{t-}} \left( \delta(p) \cdot |W^m_{#p}^{p-t}| \right) + \sum_{p \in P^{t-}} \left( \delta(p) \cdot |W^m_{#p}^{p-t}| \right) + \sum_{p \in P^{t-}} \left( \delta(p) \cdot |W^m_{#p}^{p-t}| \right) \right) \]

Thus, \( \sum_{t \in T_S - \{t_0\}} \zeta(m, t) \geq \sum_{t \in T_S - \{t_0\}} \zeta(m', t) \) holds and Lemma 5 is proven. □
Having proven that Lemma 4 and Lemma 5 hold, we now can prove Lemma 3:

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

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

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** (Theorem 4: Termination of Silent Subtraces).

From Lemma 3 results:

\[
\forall \tau \in T_\#: \exists N := \xi(mark_\#(\tau)) \in \mathbb{N} : \forall v \in T_\#^\tau \text{ with } |\tau| + N < |v| \Rightarrow con_\#(\tau) \neq con_\#(v).
\]

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.,

\[
\forall \tau, v \in T_\#: \left( opt_\#(mark_\#(\tau)), opt_\#(mark_\#(v)) \subseteq \Sigma_\# \land \Pi_{\Sigma_\#}(\tau) = \Pi_{\Sigma_\#}(v) \right) \Rightarrow opt_\#(mark_\#(\tau)) = opt_\#(mark_\#(v))
\]

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

\[mo_\# : C_\# \rightarrow 2^{\Sigma_\#} : \eta \mapsto mo_\#(\eta) \text{ maps each conversation to the set of messages it activates, with } mo_\#(\eta) := opt_\#(mark_\#(\tau)), \tau \in O_\# \text{ is defined by } \eta = con_\#(\tau) \text{ and } opt_\#(mark_\#(\tau)) \subseteq \Sigma_\#.
\]

Since \# is message-deterministic, the definition is unique. Thus, \(mo_\#\) 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 \( I = (R, D, C, dom_C, s_C, r_C, e) \) be an interaction domain and let the tuple \# = \((P, p_m, P, P_f, T, T_S, T_I, V, class, \rightarrow, \rightarrow, dom_V, grd)\) be a sound, deterministic, and message-deterministic DAI Net. Let further \( R \in R \) be a role. Then we can define the following views

- \( v_C^R : C_\# \rightarrow \Sigma_R^\#: (\eta_k)_{k \in [1..n]} \mapsto v_C^R(\eta) := \Pi_{\Sigma_R}(\eta) \text{ maps each conversation on } \# \text{ to the view of } R \text{ on it, whereby the view is the projection of the conversation to the messages sent or received by Role } R,\)

- \( \nu_C^R : C_\# \rightarrow \Sigma_R^{\#, \rightarrow} : (\eta_k)_{k \in [1..n]} \mapsto \nu_C^R(\eta) := \Pi_{\Sigma_{R \rightarrow}}(\eta) \text{ maps each conversation on } \# \text{ to the projection of the conversation to the messages sent by Role } R,\)
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 \( R \in R \):

- \( # \) is realizable, iff the messages role \( R \) may send solely depend on the messages \( R \) has sent and received before, i.e.,

\[
\forall R \in R : \forall \eta, \kappa \in C_{#} : vc_{#}(\eta) = vc_{#}(\kappa) \Rightarrow vo_{#}(mo_{#}(\eta)) = vo_{#}(mo_{#}(\kappa))
\]

- \( # \) clearly terminates, iff it solely depends on the messages \( R \) has sent and received before whether further interaction with \( R \) will occur, i.e.,

\[
\forall R \in R : \forall \eta \in C_{#} : \exists \kappa \in C_{#} : vc_{#}(\eta) \not< vc_{#}(\kappa)
\]

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]. A data-driven approach for coordinating the behavior of business objects as well as their interactions is presented in [25], [26]. In particular, micro processes represent object behavior, while macro processes coordinated the execution and interactions of inter-dependent micro processes. Overall, this approach focuses on realizing object-aware processes within enterprise information systems, but does not deal with inter-organizational processes and their interactions. 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 not be 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

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 Simultaneous 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
<table>
<thead>
<tr>
<th>Paper Number</th>
<th>Authors</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>92-08*</td>
<td>Uwe Schöning</td>
<td>On Random Reductions from Sparse Sets to Tally Sets</td>
</tr>
<tr>
<td>92-09*</td>
<td>Hermann von Hasseln, Laura Martignon</td>
<td>Consistency in Stochastic Network</td>
</tr>
<tr>
<td>92-10</td>
<td>Michael Schmitt</td>
<td>A Slightly Improved Upper Bound on the Size of Weights Sufficient to Represent Any Linearly Separable Boolean Function</td>
</tr>
<tr>
<td>92-11</td>
<td>Johannes Köbler, Seinosuke Toda</td>
<td>On the Power of Generalized MOD-Classes</td>
</tr>
<tr>
<td>92-12</td>
<td>V. Arvind, J. Köbler, M. Mundhenk</td>
<td>Reliable Reductions, High Sets and Low Sets</td>
</tr>
<tr>
<td>92-13</td>
<td>Alfons Geser</td>
<td>On a monotonic semantic path ordering</td>
</tr>
<tr>
<td>92-14*</td>
<td>Joost Engelfriet, Heiko Vogler</td>
<td>The Translation Power of Top-Down Tree-To-Graph Transducers</td>
</tr>
<tr>
<td>93-01</td>
<td>Alfred Lupper, Konrad Froitzheim</td>
<td>AppleTalk Link Access Protocol basierend auf dem Abstract Personal Communications Manager</td>
</tr>
<tr>
<td>93-03</td>
<td>Thomas Thierauf, Seinosuke Toda, Osamu Watanabe</td>
<td>On Sets Bounded Truth-Table Reducible to P-selective Sets</td>
</tr>
<tr>
<td>93-04</td>
<td>Jin-Yi Cai, Frederic Green, Thomas Thierauf</td>
<td>On the Correlation of Symmetric Functions</td>
</tr>
<tr>
<td>93-05</td>
<td>K.Kuhn, M.Reichert, M. Nathe, T. Beuter, C. Heinlein, P. Dadam</td>
<td>A Conceptual Approach to an Open Hospital Information System</td>
</tr>
<tr>
<td>93-06</td>
<td>Klaus Gaßner</td>
<td>Rechnerunterstützung für die konzeptuelle Modellierung</td>
</tr>
<tr>
<td>93-07</td>
<td>Ullrich Keßler, Peter Dadam</td>
<td>Towards Customizable, Flexible Storage Structures for Complex Objects</td>
</tr>
<tr>
<td>94-01</td>
<td>Michael Schmitt</td>
<td>On the Complexity of Consistency Problems for Neurons with Binary Weights</td>
</tr>
<tr>
<td>94-02</td>
<td>Armin Kühnemann, Heiko Vogler</td>
<td>A Pumping Lemma for Output Languages of Attributed Tree Transducers</td>
</tr>
<tr>
<td>94-03</td>
<td>Harry Buhrman, Jim Kadin, Thomas Thierauf</td>
<td>On Functions Computable with Nonadaptive Queries to NP</td>
</tr>
<tr>
<td>94-04</td>
<td>Heinz Faßbender, Heiko Vogler, Andrea Wedel</td>
<td>Implementation of a Deterministic Partial E-Unification Algorithm for Macro Tree Transducers</td>
</tr>
<tr>
<td>Paper Number</td>
<td>Authors</td>
<td>Title</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>94-05</td>
<td>V. Arvind, J. Köbler, R. Schuler</td>
<td>On Helping and Interactive Proof Systems</td>
</tr>
<tr>
<td>94-06</td>
<td>Christian Kalus, Peter Dadam</td>
<td>Incorporating record subtyping into a relational data model</td>
</tr>
<tr>
<td>94-07</td>
<td>Markus Tresch, Marc H. Scholl</td>
<td>A Classification of Multi-Database Languages</td>
</tr>
<tr>
<td>94-08</td>
<td>Friedrich von Henke, Harald Rueß</td>
<td>Arbeitstreffen Typtheorie: Zusammenfassung der Beiträge</td>
</tr>
<tr>
<td>94-10</td>
<td>Axel Dold</td>
<td>Formalisierung schematischer Algorithmen</td>
</tr>
<tr>
<td>94-11</td>
<td>Johannes Köbler, Osamu Watanabe</td>
<td>New Collapse Consequences of NP Having Small Circuits</td>
</tr>
<tr>
<td>94-12</td>
<td>Rainer Schuler</td>
<td>On Average Polynomial Time</td>
</tr>
<tr>
<td>94-13</td>
<td>Rainer Schuler, Osamu Watanabe</td>
<td>Towards Average-Case Complexity Analysis of NP Optimization Problems</td>
</tr>
<tr>
<td>94-14</td>
<td>Wolfram Schulte, Ton Vullinghs</td>
<td>Linking Reactive Software to the X-Window System</td>
</tr>
<tr>
<td>94-15</td>
<td>Alfred Lupper</td>
<td>Namensverwaltung und Adressierung in Distributed Shared Memory-Systemen</td>
</tr>
<tr>
<td>94-16</td>
<td>Robert Regn</td>
<td>Verteilte Unix-Betriebssysteme</td>
</tr>
<tr>
<td>94-17</td>
<td>Helmuth Partsch</td>
<td>Again on Recognition and Parsing of Context-Free Grammars: Two Exercises in Transformational Programming</td>
</tr>
<tr>
<td>94-18</td>
<td>Helmuth Partsch</td>
<td>Transformational Development of Data-Parallel Algorithms: an Example</td>
</tr>
<tr>
<td>95-01</td>
<td>Oleg Verbitsky</td>
<td>On the Largest Common Subgraph Problem</td>
</tr>
<tr>
<td>95-02</td>
<td>Uwe Schöning</td>
<td>Complexity of Presburger Arithmetic with Fixed Quantifier Dimension</td>
</tr>
<tr>
<td>95-03</td>
<td>Harry Buhrman, Thomas Thierauf</td>
<td>The Complexity of Generating and Checking Proofs of Membership</td>
</tr>
<tr>
<td>95-04</td>
<td>Rainer Schuler, Tomoyuki Yamakami</td>
<td>Structural Average Case Complexity</td>
</tr>
<tr>
<td>95-05</td>
<td>Klaus Achatz, Wolfram Schulte</td>
<td>Architecture Independent Massive Parallelization of Divide-And-Conquer Algorithms</td>
</tr>
<tr>
<td>95-06</td>
<td>Christoph Karg, Rainer Schuler</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-------------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Structure in Average Case Complexity</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>95-07</th>
<th>P. Dadam, K. Kuhn, M. Reichert, T. Beuter, M. Nathe</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ADEPT: Ein integrierender Ansatz zur Entwicklung flexibler, zuverlässiger kooperierender Assistenzsysteme in klinischen Anwendungsumgebungen</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>95-08</th>
<th>Jürgen Kehrer, Peter Schulthess</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Aufbereitung von gescannten Röntgenbildern zur filmlosen Diagnostik</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>95-09</th>
<th>Hans-Jörg Burtschick, Wolfgang Lindner</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>On Sets Turing Reducible to P-Selective Sets</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>95-10</th>
<th>Boris Hartmann</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Berücksichtigung lokaler Randbedingung bei globaler Zieloptimierung mit neuronalen Netzen im Beispiel Truck Backer-Upper</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>95-11</th>
<th>Thomas Beuter, Peter Dadam</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Prinzipien der Replikationskontrolle in verteilten Systemen</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>95-12</th>
<th>Klaus Achatz, Wolfram Schulte</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Massive Parallelization of Divide-and-Conquer Algorithms over Powerlists</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>95-13</th>
<th>Andrea Mößle, Heiko Vogler</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Efficient Call-by-value Evaluation Strategy of Primitive Recursive Program Schemes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>95-14</th>
<th>Axel Dold, Friedrich W. von Henke, Holger Pfeifer, Harald Rueß</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A Generic Specification for Verifying Peephole Optimizations</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>96-01</th>
<th>Ercüment Canver, Jan-Tecker Gayen, Adam Moik</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Formale Entwicklung der Steuerungssoftware für eine elektrisch ortsbediente Weiche mit VSE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>96-02</th>
<th>Bernhard Nebel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Solving Hard Qualitative Temporal Reasoning Problems: Evaluating the Efficiency of Using the ORD-Horn Class</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>96-03</th>
<th>Ton Vullinghs, Wolfram Schulte, Thilo Schwinn</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>An Introduction to TkGofer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>96-04</th>
<th>Thomas Beuter, Peter Dadam</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Anwendungsspezifische Anforderungen an Workflow-Mangement-Systeme am Beispiel der Domäne Concurrent-Engineering</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>96-05</th>
<th>Gerhard Schellhorn, Wolfgang Ahrendt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Verification of a Prolog Compiler - First Steps with KIV</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>96-06</th>
<th>Manindra Agrawal, Thomas Thierauf</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Satisfiability Problems</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>96-07</th>
<th>Vikraman Arvind, Jacobo Torán</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A nonadaptive NC Checker for Permutation Group Intersection</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>96-08</th>
<th>David Cyrluk, Oliver Möller, Harald Rueß</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>An Efficient Decision Procedure for a Theory of Fix-Sized Bitvectors with Composition and Extraction</td>
</tr>
</tbody>
</table>
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
ADEPT flex - 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 Wennemers
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 Wennemers
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

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
<table>
<thead>
<tr>
<th>Year</th>
<th>Author(s)</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000-04</td>
<td>Wolfgang Gehring</td>
<td>Ein Rahmenwerk zur Einführung von Leistungspunktsystemen</td>
</tr>
<tr>
<td>2000-05</td>
<td>Susanne Boll, Christian Heinlein, Wolfgang Klas, Jochen Wandel</td>
<td>Intelligent Prefetching and Buffering for Interactive Streaming of MPEG Videos</td>
</tr>
<tr>
<td>2000-06</td>
<td>Wolfgang Reif, Gerhard Schellhorn, Andreas Thums</td>
<td>Fehlersuche in Formalen Spezifikationen</td>
</tr>
<tr>
<td>2000-08</td>
<td>Thomas Bauer, Manfred Reichert, Peter Dadam</td>
<td>Effiziente Durchführung von Prozessmigrationen in verteilten Workflow-Management-Systemen</td>
</tr>
<tr>
<td>2000-09</td>
<td>Thomas Bauer, Peter Dadam</td>
<td>Vermeidung von Überlastsituationen durch Replikation von Workflow-Servern in ADEPT</td>
</tr>
<tr>
<td>2000-10</td>
<td>Thomas Bauer, Manfred Reichert, Peter Dadam</td>
<td>Adaptives und verteiltes Workflow-Management</td>
</tr>
<tr>
<td>2000-11</td>
<td>Christian Heinlein</td>
<td>Workflow and Process Synchronization with Interaction Expressions and Graphs</td>
</tr>
<tr>
<td>2001-01</td>
<td>Hubert Hug, Rainer Schuler</td>
<td>DNA-based parallel computation of simple arithmetic</td>
</tr>
<tr>
<td>2001-02</td>
<td>Friedhelm Schwenker, Hans A. Kestler, Günther Palm</td>
<td>3-D Visual Object Classification with Hierarchical Radial Basis Function Networks</td>
</tr>
<tr>
<td>2001-03</td>
<td>Hans A. Kestler, Friedhelm Schwenker, Günther Palm</td>
<td>RBF network classification of ECGs as a potential marker for sudden cardiac death</td>
</tr>
<tr>
<td>2001-04</td>
<td>Christian Dietrich, Friedhelm Schwenker, Klaus Riede, Günther Palm</td>
<td>Classification of Bioacoustic Time Series Utilizing Pulse Detection, Time and Frequency Features and Data Fusion</td>
</tr>
<tr>
<td>2002-01</td>
<td>Stefanie Rinderle, Manfred Reichert, Peter Dadam</td>
<td>Effiziente Verträglichkeitsprüfung und automatische Migration von Workflow-Instanzen bei der Evolution von Workflow-Schemata</td>
</tr>
<tr>
<td>2002-02</td>
<td>Walter Guttmann</td>
<td>Deriving an Applicative Heapsort Algorithm</td>
</tr>
<tr>
<td>2003-01</td>
<td>Manfred Reichert, Stefanie Rinderle, Peter Dadam</td>
<td>A Formal Framework for Workflow Type and Instance Changes Under Correctness Checks</td>
</tr>
<tr>
<td>2003-02</td>
<td>Stefanie Rinderle, Manfred Reichert, Peter Dadam</td>
<td>Supporting Workflow Schema Evolution By Efficient Compliance Checks</td>
</tr>
<tr>
<td>Year</td>
<td>Authors/Title</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
</tbody>
</table>
| 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
Entscheidungs/documentation 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^c 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 instansspezifischer 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