Theoretical and Applied Informatics
ISSN 1896–5334
Vol.24 (2012), no. 1
pp. 33–66
DOI: 10.2478/v10179-012-0003-6

Foundation for Equivalences of BPMN Models

VITUS S.W. LAM

Computer Centre, The University of Hong Kong,
Pokfulam Road, Hong Kong.
vitus.lam@ieee.org

Received 28 January 2012, Revised 15 February 2012, Accepted 24 February 2012.

Abstract: Business Process Modelling Notation (BPMN) is a visual specification language without well-defined concepts for equivalences. This necessitates the establishment of fundamental notions that underpin the equivalences of BPMN processes. The main body of the paper is centered around the principle of substitutibility in which different types of equivalences of BPMN processes are formally described. Additionally, these results provide a basis for defining the behavioural equivalence of BPMN models. Our research investigation contributes to the field of business process management by developing a tight connection between BPMN and its associated equivalence notions.

Keywords: BPMN, process equivalence, model equivalence, equivalence classification

1. Introduction

Business Process Modelling Notation (BPMN) [1], Unified Modelling Language (UML) activity diagrams [2], Business Process Executable Language (BPEL) and Web Services Choreography Description Language (WS-CDL) are notable emerging standards in the domain of business process management. Both BPMN and UML activity diagrams, maintained by the Object Management Group, are visual modelling languages for documenting, specifying and designing business processes. BPEL is an XML-based orchestration language, whereas WS-CDL is an XML-based choreography language. They are managed, respectively, by Organization for the Advancement of Structured Information Standards (OASIS) and World Wide Web Consortium (W3C). Unlike BPMN and UML activity diagrams that are graphical modelling languages, BPEL and WS-CDL are text-based modelling languages. In addition, BPEL is executable, whereas BPMN and UML activity diagrams are non-executable.

During the construction of business process models, a key aspect is to determine whether a business process model is a substitute for another business process model regardless of their physical representations. An important motivation for the replacement of the original business process model is to reduce design complexity through the use of an equivalent model with a more compact representation. As BPEL and WS-CDL are intended to be used by programmers rather than business analysts for capturing the design of business processes, this leads to theoretical and practical challenges for developing sound theories about the equivalence checking of models targeted at BPMN and UML activity diagrams in lieu of the two text-based modelling languages. In our prior works [3] and [4], a variety of equivalences for BPMN processes and UML activity diagrams are examined. This work is considered as part of a series of research studies with regard to the behavioural equivalences of workflow models. In particular, additional equivalence notions of BPMN processes other than those covered in [3] as well as behavioural equivalence of BPMN models are explored.

The rest of this paper proceeds as follows. Section 2 describes related studies in the area. A review of the notational elements of BPMN is presented in Section 3. Section 4 introduces a foundation for BPMN that serves as an underlying model for defining when BPMN models are equivalent. Section 5 deals with the equivalences of BPMN processes as well as the behavioural equivalence of BPMN models. A practical application of the various types of equivalences is illustrated by means of a concrete example. Section 6 concludes the paper and points to promising areas for future work.

2. Related Work

A critical review of the literature is given in this section. We begin by considering studies related to the simulation, analysis and verification of business processes modelled as BPMN 1.0 [5]. The techniques adopted by these attempts fall into two main categories: process-algebraic techniques and graphical-based techniques. The π-calculus [6, 7, 8] and Communicating Sequential Processes (CSP) [9] are utilized in the process-algebraic approaches, whereas Petri-nets and Colored Petri-nets are employed in the graphical-based methods.

Bog et al. [10, 11, 12, 13] propose an approach to encode BPMN models in the π-calculus. An automated tool PiVizTool [10, 11, 12, 13] is then used for simulating and analyzing the associated π-calculus specifications. Dijkman et al. [14] advocate the analysis of BPMN models with ProM framework by transforming them into Petri nets. Through the definition of a semantic mapping between BPMN and CSP, Wong and Gibbons [15, 16] analyze the compatibility between BPMN processes by means of the Failure-divergence Refinement (FDR) model checker [17]. They provide an extension of previous endeavour in [18, 19] by developing a relative-time semantic model on the basis of CSP.

Puhlmann [20] converts BPMN models into the π-calculus in order to check the validity of the models by using the Advanced Bisimulation Checker (ABC) [21]. Ou-Yang and Lin [22] present a two-step transformation in which BPMN models are translated into BPEL4WS and BPEL4WS into Colored Petri-net XML (CPNXML). A verification of various properties is then carried out with CPN tools [23]. To verify the correctness of BPMN models, Raedts et al. [24] make use of Petri nets as an intermediate representation when formalizing BPMN models as mCRL2. As opposed to these research studies that emphasize the simulation, analysis and verification of BPMN, the primary focus of our work is on developing a theoretical framework for the equivalences of BPMN processes and diagrams. Besides, our formalization is concerned with BPMN 1.2 in lieu of BPMN 1.0 and covers all graphical constructs of BPMN 1.2.

In what follows, we offer a review of previous contributions that are related to the study of equivalence checking in the context of business modelling. Equivalences of UML statechart diagrams comprising isomorphism, strong behavioural equivalence and weak behavioural equivalence are formally specified in terms of structural congruence and open bisimulations in [25]. Gruber and Eder et al. [26, 27] systematize different types of semantics-preserving transformations of workflows. Our work is distinguished from these studies [26, 27] in two respects. Firstly, we examine the equivalences of BPMN processes and diagrams in lieu of structured workflow graphs. Secondly, we propose several BPMN specific equivalences that allow us to substitute one BPMN process for another BPMN process repeatedly. In [4], we discuss a methodological framework for categorizing various kinds of equivalences for UML activity diagrams. A foundational theory for the equivalences of BPMN 1.1 processes is delineated in our prior work [3]. This paper goes one step further by establishing a formal basis for the equivalences of BPMN 1.2 models.

3. Graphical Syntax and Execution Semantics of BPMN

This section, which the diagrams are adapted from [3], intends to give a brief introduction to BPMN 1.2. We refer the reader to [1] and [28] for further reading on this subject.

The four kinds of graphical elements in BPMN are flow objects, connecting objects, artifacts and swimlanes. The flow objects as depicted in Figure 1 are divided into three types: events, activities and gateways. The start of a process is represented by a start event in which a token is created. A none start event signifies either the event type is undefined or the commencement of a subprocess. The receipt of a message, the occurrence of a particular date and time, the holding of a condition and the receipt of a signal are denoted by a message start event, a timer start event, a conditional start event and a signal start event, respectively. A multiple start event can be triggered by more than one start events.

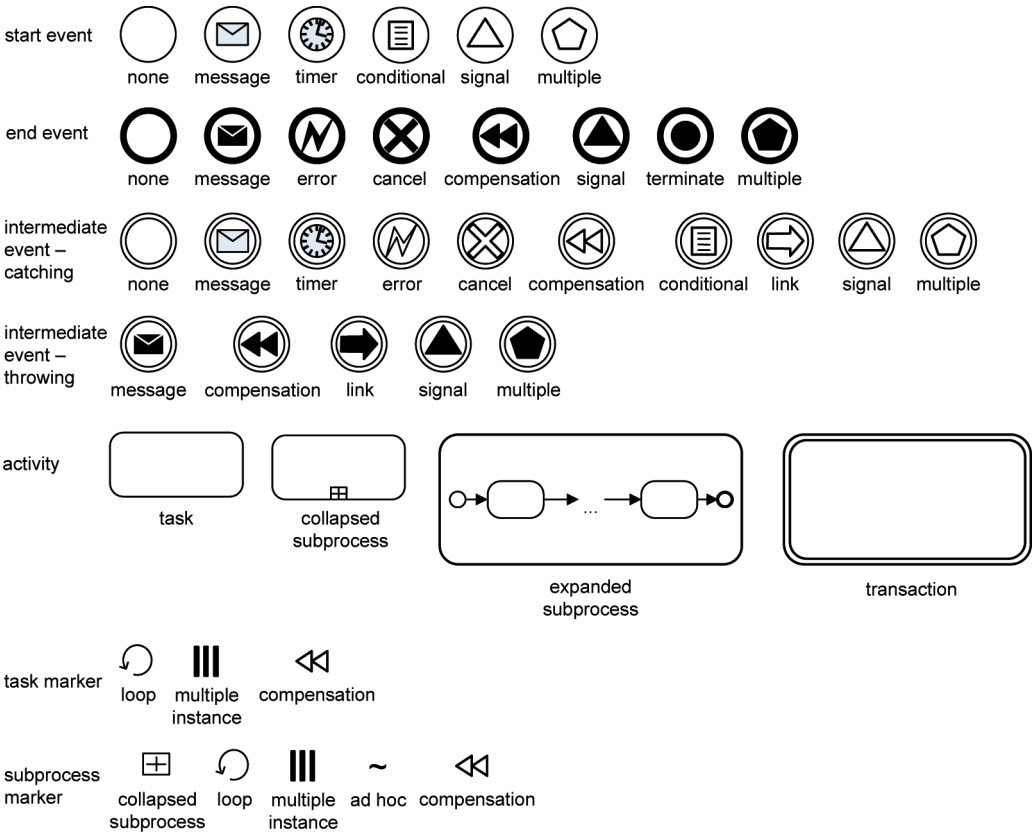

Fig. 1. BPMN notational elements

The end of a process is notated as an end event in which a token is consumed. A none end event symbolizes either the event type is undefined or the end of a subprocess. The sending of a message to another participant, the generation of an error, the cancellation of a transaction subprocess, the triggering of a compensation, the broadcasting of a signal and the immediate termination of a process as well as its subprocesses are rendered by a message end event, an error end event, a cancel end event, a compensation end event, a signal end event and a terminate end event. A multiple end event can throw more than one end events.

The occurrence of an event during the execution of a process is represented as either a catch intermediate event or a throw intermediate event. A catching none intermediate event, a catching message intermediate event, a catching timer intermediate event, a catching conditional intermediate event, a catching signal intermediate event and a catching multiple intermediate event are defined in a similar way as a none start event, a message start event, a timer start event, a conditional start event, a signal start

event and a multiple start event. In the same spirit, a throwing message intermediate event, a throwing compensation intermediate event, a throwing signal intermediate event and a throwing multiple intermediate event are similar to a message end event, a compensation end event, a signal end event and a multiple end event. A catching error intermediate event, a catching cancel intermediate event, a catching compensation intermediate event, a catching link intermediate event and a throwing link intermediate event indicate the catching of an error, the catching of a transaction cancellation, the catching of a compensation event, the receipt of a link event and the sending of a link event.

An activity is either a task or a subprocess. There are three kinds of task markers: loop, multiple instance and compensation. Similarly, the subprocess markers are divided into five categories: collapsed subprocess, loop, multiple instance, ad hoc and compensation. A task is atomic, whereas a subprocess is decomposable. Depending on whether the details of a subprocess are hidden or not, a subprocess is classified as a collapsed subprocess or a expanded subprocess. A transaction is a subprocess whose enclosed activities are either completed or reverted.

A data-based exclusive decision gateway sends a token along one of the mutually exclusive outgoing sequence flows according to which conditional expression holds (Figure 2). A data-based exclusive merge gateway emits a token on the outgoing sequence flow whenever a token is received from one of the incoming sequence flows. An event-based exclusive decision gateway offers a token to each of the mutually exclusive outgoing sequence flows based on the receipt of an event. An event-based exclusive merge gateway passes any received token to the outgoing sequence flow.

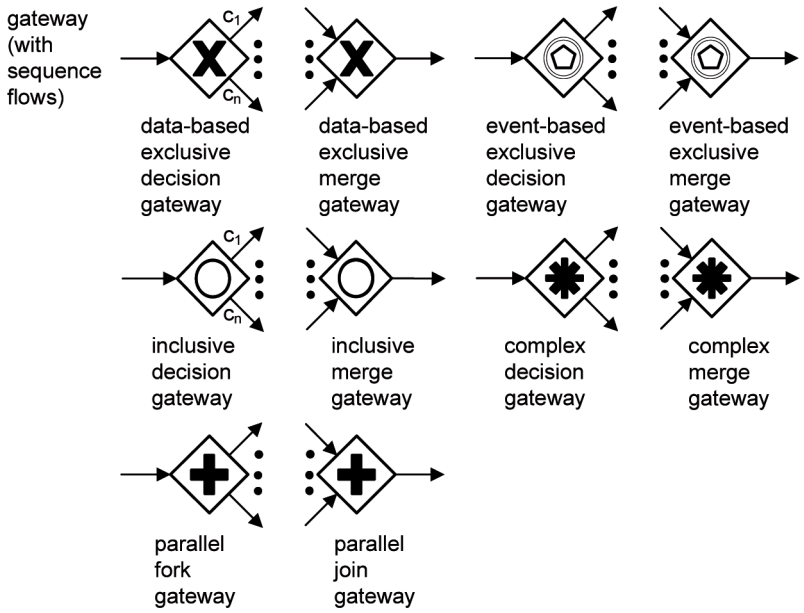

Fig. 2. BPMN notational elements (continued)

An inclusive decision gateway activates multiple outgoing sequence flows by generating a token on all outgoing sequence flows in which the associated conditional expressions hold. An inclusive merge gateway blocks until all the expected tokens are arrived before a token is sent on the outgoing sequence flow. A complex decision gateway and a complex merge gateway decide, respectively, the collection of outgoing sequence flows that are activated and the set of incoming sequence flows that tokens are expected depending on an expression. A parallel fork gateway splits a process flow by sending tokens on all outgoing sequence flows. A parallel join gateway merge process flows by waiting until a token is arrived from each incoming sequence flow.

As shown in Figure 3, there are three types of connecting objects: sequence flows, message flows and associations. A sequence flow connects two flow objects in a process. A normal flow refers to a flow that passes over a set of gateways, whereas an uncontrolled flow does not include any gateways in between the start event and the end event. A conditional flow is a sequence flow that contains a conditional expression. A default flow is selected only if the conditional expressions of all the other sequence flows do not hold. A message flow specifies the interaction between two participants. An association links up a flow object with an artifact. A directed association is used for defining a data object is an input or output of an activity. An (non-directional) association connects a text annotation with a flow object.

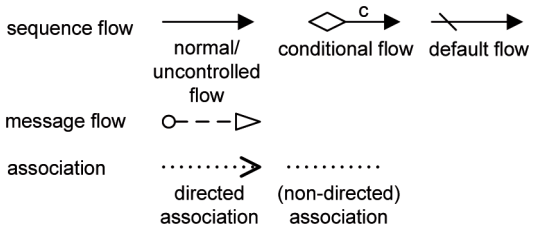

Fig. 3. BPMN notational elements (continued)

The three standard artifacts in BPMN as delineated in Figure 4 are data objects, groups and text annotations. A data object, which does not affect the flow of a process, stands for data or document. It is the input and output of an activity. A group is a graphical element for highlighting a group of notational elements. A text annotation furnishes further description on a process or notational element.

A pool, which symbolizes a participant, is a container of a process. Each pool consists of one or more lanes in which there is a unique name for each lane. A lane provides a mechanism for grouping the notational elements of a process.

4. Formal Definitions for BPMN

BPMN is described in narrative form using informal English in [1]. This section takes on the challenge of providing a mathematical model for BPMN 1.2. It builds upon

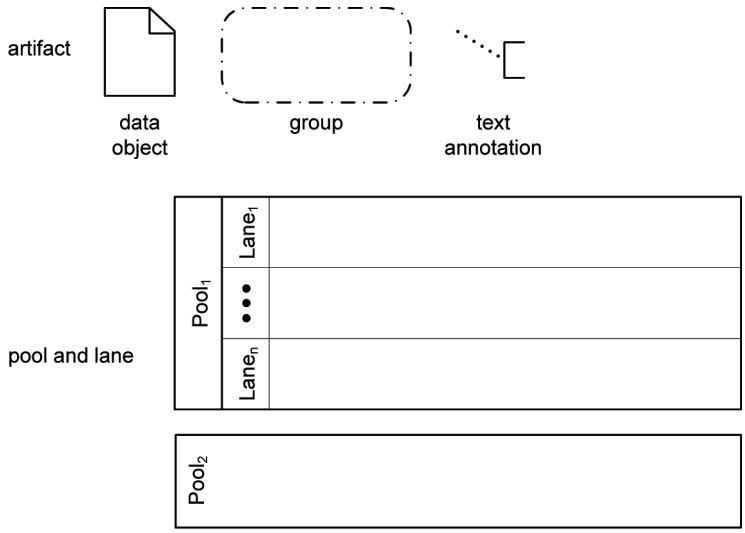

Fig. 4. BPMN notational elements (continued)

the mathematical definitions advocated in [3]. Definitions 1–16 are an adaptation of the ones presented in [3]. Definitions 17–36 attempt to extend our previous work by incorporating lanes, pools and business process diagrams into the model. The results of this section provide the formal rigour necessary for studying the equivalences of BPMN models.

We start by giving three definitions that capture the concepts of start events, intermediate events and end events from a formal perspective. With the presence of these definitions, a formalization of the notion of events is then introduced.

Definition 1 (Start-event Tuple) *A start-event tuple is a 6-tuple* $\Omega_{\mathrm{SE}} = (F_{\mathrm{SE}}^{\mathrm{None}}, F_{\mathrm{SE}}^{\mathrm{Msg}}, F_{\mathrm{SE}}^{\mathrm{Timer}}, F_{\mathrm{SE}}^{\mathrm{Cond}}, F_{\mathrm{SE}}^{\mathrm{Sign}}, F_{\mathrm{SE}}^{\mathrm{Multi}})$ *where* $F_{\mathrm{SE}}^{\mathrm{None}}$, $F_{\mathrm{SE}}^{\mathrm{Msg}}$, $F_{\mathrm{SE}}^{\mathrm{Timer}}$, $F_{\mathrm{SE}}^{\mathrm{Cond}}$, $F_{\mathrm{SE}}^{\mathrm{Sign}}$ *and* $F_{\mathrm{SE}}^{\mathrm{Multi}}$ *are sets of none start events, message start events, timer start events, conditional start events, signal start events and multiple start events for catching the event triggers.*

A start-event tuple contains six kinds of start events comprising none, message, timer, conditional, signal and multiple.

Definition 2 (Intermediate-event Tuple) *An intermediate-event tuple is a 15-tuple* $\Omega_{\mathrm{IE}} = (F_{\mathrm{IE}}^{\mathrm{None}}, F_{\mathrm{IE}}^{\mathrm{Msg}}, F_{\mathrm{IE}}^{\overline{\mathrm{Msg}}}, F_{\mathrm{IE}}^{\mathrm{Timer}}, F_{\mathrm{IE}}^{\mathrm{Err}}, F_{\mathrm{IE}}^{\mathrm{Cncl}}, F_{\mathrm{IE}}^{\mathrm{Cmpen}}, F_{\mathrm{IE}}^{\overline{\mathrm{Cmpen}}}, F_{\mathrm{IE}}^{\mathrm{Cond}}, F_{\mathrm{IE}}^{\mathrm{Link}}, F_{\mathrm{IE}}^{\overline{\mathrm{Link}}}, F_{\mathrm{IE}}^{\mathrm{Sign}}, F_{\mathrm{IE}}^{\overline{\mathrm{Sign}}}, F_{\mathrm{IE}}^{\mathrm{Multi}}, F_{\mathrm{IE}}^{\overline{\mathrm{Multi}}})$ *where*

— $F_{\mathrm{IE}}^{\mathrm{None}}$, $F_{\mathrm{IE}}^{\mathrm{Msg}}$, $F_{\mathrm{IE}}^{\mathrm{Timer}}$, $F_{\mathrm{IE}}^{\mathrm{Err}}$, $F_{\mathrm{IE}}^{\mathrm{Cncl}}$, $F_{\mathrm{IE}}^{\mathrm{Cmpen}}$, $F_{\mathrm{IE}}^{\mathrm{Cond}}$, $F_{\mathrm{IE}}^{\mathrm{Link}}$, $F_{\mathrm{IE}}^{\mathrm{Sign}}$ *and* $F_{\mathrm{IE}}^{\mathrm{Multi}}$ *are sets of none intermediate events, message intermediate events, timer intermediate*

events, error intermediate events, cancel intermediate events, compensation intermediate events, conditional intermediate events, link intermediate events, signal intermediate events and multiple intermediate events for catching the event triggers; and

— $F_{\mathrm{IE}}^{\overline{\mathrm{Msg}}}$, $F_{\mathrm{IE}}^{\overline{\mathrm{Cmpen}}}$, $F_{\mathrm{IE}}^{\overline{\mathrm{Link}}}$, $F_{\mathrm{IE}}^{\overline{\mathrm{Sign}}}$ *and* $F_{\mathrm{IE}}^{\overline{\mathrm{Multi}}}$ *are sets of message intermediate events, compensation intermediate events, link intermediate events, signal intermediate events and multiple intermediate events for throwing the event triggers.*

In BPMN, there are ten kinds of intermediate events for catching the event triggers and five types of intermediate events for throwing the event triggers. The notion is captured formally in form of an intermediate-event tuple.

Definition 3 (End-event Tuple) *A end-event tuple is a 8-tuple* $\Omega_{\mathrm{EE}} = (F_{\mathrm{EE}}^{\overline{\mathrm{None}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Msg}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Err}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Cncl}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Cmpen}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Sign}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Term}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Multi}}})$ *where* $F_{\mathrm{EE}}^{\overline{\mathrm{None}}}$, $F_{\mathrm{EE}}^{\overline{\mathrm{Msg}}}$, $F_{\mathrm{EE}}^{\overline{\mathrm{Err}}}$, $F_{\mathrm{EE}}^{\overline{\mathrm{Cncl}}}$, $F_{\mathrm{EE}}^{\overline{\mathrm{Cmpen}}}$, $F_{\mathrm{EE}}^{\overline{\mathrm{Sign}}}$, $F_{\mathrm{EE}}^{\overline{\mathrm{Term}}}$ *and* $F_{\mathrm{EE}}^{\overline{\mathrm{Multi}}}$ *are sets of none end events, message end events, error end events, cancel end events, compensation end events, signal end events, terminate end events and multiple end events for throwing the event triggers.*

An end-event tuple divides end events into the following categories: none, message, error, cancel, compensation, signal, terminate and multiple.

Definition 4 (Event Tuple) *Suppose* $\Gamma_{\mathrm{SE}} = \{\mathrm{None}, \mathrm{Msg}, \mathrm{Timer}, \mathrm{Cond}, \mathrm{Sign}, \mathrm{Multi}\}$, $\Gamma_{\mathrm{EE}} = \{\mathrm{None}, \mathrm{Msg}, \mathrm{Err}, \mathrm{Cncl}, \mathrm{Cmpen}, \mathrm{Sign}, \mathrm{Term}, \mathrm{Multi}\}$, $\Gamma_{\mathrm{IE}} = \{\mathrm{None}, \mathrm{Msg}, \mathrm{Timer}, \mathrm{Err}, \mathrm{Cncl}, \mathrm{Cmpen}, \mathrm{Cond}, \mathrm{Link}, \mathrm{Sign}, \mathrm{Multi}\}$, $\Gamma_{\overline{\mathrm{IE}}} = \{\overline{\mathrm{Msg}}, \overline{\mathrm{Cmpen}}, \overline{\mathrm{Link}}, \overline{\mathrm{Sign}}, \overline{\mathrm{Multi}}\}$, $F_{\mathrm{SE}} = \bigcup_{i \in \Gamma_{\mathrm{SE}}} F_{\mathrm{SE}}^{i}$, $F_{\mathrm{EE}} = \bigcup_{i \in \Gamma_{\mathrm{EE}}} F_{\mathrm{EE}}^{i}$, $F_{\mathrm{IE}} = \bigcup_{i \in (\Gamma_{\mathrm{IE}} \cup \Gamma_{\overline{\mathrm{IE}}})} F_{\mathrm{IE}}^{i}$, $F_{\mathrm{E}} = \bigcup_{i \in \{\mathrm{SE},\mathrm{EE},\mathrm{IE}\}} F_i$, $S_{\mathrm{E}}^{\mathrm{Att}}$ *is a set of event attributes and* $S_{\mathrm{E}}^{\mathrm{AttV}}$ *is a set of event attribute values. An event tuple is a 4-tuple* $\Omega_{\mathrm{E}} = (\Omega_{\mathrm{SE}}, \Omega_{\mathrm{IE}}, \Omega_{\mathrm{EE}}, \Phi_{\mathrm{E}}^{\mathrm{Att}})$ *where*

— Ω_{SE} *is a start-event tuple;*

— Ω_{IE} *is an intermediate-event tuple;*

— Ω_{EE} *is an end-event tuple; and*

— $\Phi_{\mathrm{E}}^{\mathrm{Att}} : F_{\mathrm{E}} \times S_{\mathrm{E}}^{\mathrm{Att}} \rightarrow S_{\mathrm{E}}^{\mathrm{AttV}}$ *relates an event and an event attribute to an event attribute value.*

F_{SE}, F_{EE} and F_{IE} are sets of start events, end events and intermediate events. We define a function $\Phi_{\mathrm{E}}^{\mathrm{Att}}$ which returns the event attribute value for a particular event attribute of an event. An event tuple is specified in terms of a start-event tuple, an intermediate-event tuple, an end-event tuple and a function $\Phi_{\mathrm{E}}^{\mathrm{Att}}$. Next, we concentrate on the formal description of tasks and subprocesses on which the notion of activities is built.

Definition 5 (Task Tuple) *Suppose M_{L} represents the loop marker, M_{MI} represents the multiple instance marker, M_{C} represents the compensation marker, the valid combination of markers for tasks $S_{\mathrm{T}}^{\mathrm{M}} = \{\{M_{\mathrm{L}}\}, \{M_{\mathrm{MI}}\}, \{M_{\mathrm{C}}\}, \{M_{\mathrm{L}}, M_{\mathrm{C}}\}, \{M_{\mathrm{MI}}, M_{\mathrm{C}}\}\}$, the types of BPMN tasks $\Gamma_{\mathrm{T}} = \{$Service, Receive, Send, User, Script, Manual, Reference, None$\}$ and S_{TNames} is a set of task names. A task tuple is a 4-tuple $\Omega_{\mathrm{T}} = (F_{\mathrm{T}}, \Phi_{\mathrm{TM}}, \Phi_{\mathrm{Ttype}}, \Phi_{\mathrm{TName}})$ where*

- *F_{T} is a set of tasks;*
- *$\Phi_{\mathrm{TM}} : F_{\mathrm{T}} \rightarrow S_{\mathrm{T}}^{\mathrm{M}}$ defines for a task its set of markers;*
- *$\Phi_{\mathrm{Ttype}} : F_{\mathrm{T}} \rightarrow \Gamma_{\mathrm{T}}$ returns the type of a task; and*
- *$\Phi_{\mathrm{TName}} : F_{\mathrm{T}} \rightarrow S_{\mathrm{TNames}}$ maps a task to its name.*

Definition 6 (Subprocess Tuple) *Suppose M_{CSP} represents the collapsed subprocess marker, M_{L} represents the loop marker, M_{MI} represents the multiple instance marker, M_{AD} represents the ad hoc marker, M_{C} represents the compensation marker, the valid combination of markers for collapsed subprocesses $S_{\mathrm{CSP}}^{\mathrm{M}} = \{\{M_{\mathrm{CSP}}\}, \{M_{\mathrm{CSP}}, M_{\mathrm{L}}\}, \{M_{\mathrm{CSP}}, M_{\mathrm{MI}}\}, \{M_{\mathrm{CSP}}, M_{\mathrm{AD}}\}, \{M_{\mathrm{CSP}}, M_{\mathrm{C}}\}, \{M_{\mathrm{CSP}}, M_{\mathrm{L}}, M_{\mathrm{AD}}\}, \{M_{\mathrm{CSP}}, M_{\mathrm{L}}, M_{\mathrm{C}}\}, \{M_{\mathrm{CSP}}, M_{\mathrm{MI}}, M_{\mathrm{AD}}\}, \{M_{\mathrm{CSP}}, M_{\mathrm{MI}}, M_{\mathrm{C}}\}, \{M_{\mathrm{CSP}}, M_{\mathrm{C}}, M_{\mathrm{AD}}\}, \{M_{\mathrm{CSP}}, M_{\mathrm{L}}, M_{\mathrm{AD}}, M_{\mathrm{C}}\}, \{M_{\mathrm{CSP}}, M_{\mathrm{MI}}, M_{\mathrm{AD}}, M_{\mathrm{C}}\}\}$, the valid combination of markers for expanded subprocesses $S_{\mathrm{ESP}}^{\mathrm{M}} = \{\{\}, \{M_{\mathrm{L}}\}, \{M_{\mathrm{MI}}\}, \{M_{\mathrm{AD}}\}, \{M_{\mathrm{C}}\}, \{M_{\mathrm{L}}, M_{\mathrm{AD}}\}, \{M_{\mathrm{L}}, M_{\mathrm{C}}\}, \{M_{\mathrm{MI}}, M_{\mathrm{AD}}\}, \{M_{\mathrm{MI}}, M_{\mathrm{C}}\}, \{M_{\mathrm{C}}, M_{\mathrm{AD}}\}, \{M_{\mathrm{L}}, M_{\mathrm{AD}}, M_{\mathrm{C}}\}, \{M_{\mathrm{MI}}, M_{\mathrm{AD}}, M_{\mathrm{C}}\}\}$, S_{NP} is a set of none-start-events processes, S_{P} is a set of processes and $\mathbb{B}$ is the set of Boolean values. A subprocess tuple is a 10-tuple $\Omega_{\mathrm{SP}} = (F_{\mathrm{SP}}^{\mathrm{Embed}}, F_{\mathrm{SP}}^{\mathrm{Reuse}}, F_{\mathrm{SP}}^{\mathrm{Ref}}, \Phi_{\mathrm{IsTX}}, \Phi_{\mathrm{SPM}}, \Phi_{\mathrm{SE}}^{\mathrm{Bdy}}, \Phi_{\mathrm{EE}}^{\mathrm{Bdy}}, \Phi_{\mathrm{NP}}, \Phi_{\mathrm{P}}, \Phi_{\mathrm{RP}})$ where*

- *$F_{\mathrm{SP}}^{\mathrm{Embed}}$ is a set of embedded subprocesses;*
- *$F_{\mathrm{SP}}^{\mathrm{Reuse}}$ is a set of reusable subprocesses;*
- *$F_{\mathrm{SP}}^{\mathrm{Ref}}$ is a set of reference subprocesses;*
- *$\Phi_{\mathrm{IsTX}} : F_{\mathrm{SP}}^{\mathrm{Embed}} \cup F_{\mathrm{SP}}^{\mathrm{Reuse}} \cup F_{\mathrm{SP}}^{\mathrm{Ref}} \rightarrow \mathbb{B}$ returns whether a subprocess is a transaction or not;*
- *$\Phi_{\mathrm{SPM}} : F_{\mathrm{SP}}^{\mathrm{Embed}} \cup F_{\mathrm{SP}}^{\mathrm{Reuse}} \cup F_{\mathrm{SP}}^{\mathrm{Ref}} \rightarrow S_{\mathrm{ESP}}^{\mathrm{M}} \cup S_{\mathrm{CSP}}^{\mathrm{M}}$ specifies for a subprocess its set of markers;*
- *$\Phi_{\mathrm{SE}}^{\mathrm{Bdy}} : \{x | x \in (F_{\mathrm{SP}}^{\mathrm{Embed}} \cup F_{\mathrm{SP}}^{\mathrm{Reuse}}) \wedge \Phi_{\mathrm{SPM}}(x) \in S_{\mathrm{ESP}}^{\mathrm{M}}\} \rightarrow 2^{F_{\mathrm{SE}}}$ returns the set of start events attached to the boundary of an expanded subprocess;*
- *$\Phi_{\mathrm{EE}}^{\mathrm{Bdy}} : \{x | x \in (F_{\mathrm{SP}}^{\mathrm{Embed}} \cup F_{\mathrm{SP}}^{\mathrm{Reuse}}) \wedge \Phi_{\mathrm{SPM}}(x) \in S_{\mathrm{ESP}}^{\mathrm{M}}\} \rightarrow 2^{F_{\mathrm{EE}}}$ returns the set of end events attached to the boundary of an expanded subprocess;*
- *$\Phi_{\mathrm{NP}} : F_{\mathrm{SP}}^{\mathrm{Embed}} \rightarrow S_{\mathrm{NP}}$ returns the associated none-start-events process;*
- *$\Phi_{\mathrm{P}} : F_{\mathrm{SP}}^{\mathrm{Reuse}} \rightarrow S_{\mathrm{P}}$ returns the called process; and*

— $\Phi_{\mathrm{RP}} : F_{\mathrm{SP}}^{\mathrm{Ref}} \rightarrow \bigcup_{i \in \{\mathrm{Embed,Reuse,Ref}\}} F_{\mathrm{SP}}^{i}$ *returns the subprocess being referenced.*

A task tuple and a subprocess tuple comprise a collection of functions as well as, respectively, a set of tasks and sets of embedded subprocesses, reusable subprocesses and reference subprocesses. There are three kinds of task markers: loop markers, multiple instance markers and compensation markers. The valid combination of markers is based on Section 9.4.3 in [1]. Likewise, four subprocess markers are allowed to use in both collapsed subprocesses and expanded subprocesses. These encompass loop markers, multiple instance markers, ad hoc markers and compensation markers. The sets $\{M_{\mathrm{CSP}}$, M_{L}, $M_{\mathrm{AD}}\}$ and $\{M_{\mathrm{CSP}}$, M_{L}, $M_{\mathrm{C}}\}$ specify that the placement of both a loop marker and a multiple instance marker in a collapsed subprocess is invalid as defined in Section 9.4.2 of the BPMN specification [1].

Definition 7 (Activity Tuple) *Suppose* $\Gamma_{\mathrm{SP}} = \{$Embed, Reuse, Ref$\}$, $\Gamma_{\mathrm{IE}} = \{$None, Msg, Timer, Err, Cncl, Cmpen, Cond, Link, Sign, Multi$\}$, $\Gamma_{\mathrm{NLC}} = \{$None, Link, Cncl$\}$, $\Gamma_{\mathrm{NL}} = \{$None, Link$\}$, $F_{\mathrm{A}} = F_{\mathrm{T}} \cup \bigcup_{i \in \Gamma_{\mathrm{SP}}} F_{\mathrm{SP}}^{i}$, $S_{\mathrm{TX}} = \{x | x \in \bigcup_{i \in \Gamma_{\mathrm{SP}}} F_{\mathrm{SP}}^{i} \wedge \Phi_{\mathrm{IsTX}}(x) = true\}$, $S_{\mathrm{A}}^{\mathrm{Att}}$ *is a set of activity attributes and* $S_{\mathrm{A}}^{\mathrm{AttV}}$ *is a set of activity attribute values. An activity tuple is a 5-tuple* $\Omega_{\mathrm{A}} = (\Omega_{\mathrm{T}}, \Omega_{\mathrm{SP}}, \Phi_{\mathrm{IE}}^{\mathrm{Bdy[-TX]}}, \Phi_{\mathrm{IE}}^{\mathrm{Bdy[TX]}}, \Phi_{\mathrm{A}}^{\mathrm{Att}})$ *where*

— Ω_{T} *is a task tuple;*

— Ω_{SP} *is a subprocess tuple;*

— $\Phi_{\mathrm{IE}}^{\mathrm{Bdy[-TX]}} : F_{\mathrm{A}} \setminus S_{\mathrm{TX}} \rightarrow 2^{\bigcup_{i \in \Gamma_{\mathrm{IE}} \setminus \Gamma_{\mathrm{NLC}}} F_{\mathrm{IE}}^{i}}$ *returns the set of intermediate events attached to the boundary of an activity that is not a transaction;*

— $\Phi_{\mathrm{IE}}^{\mathrm{Bdy[TX]}} : S_{\mathrm{TX}} \rightarrow 2^{\bigcup_{i \in \Gamma_{\mathrm{IE}} \setminus \Gamma_{\mathrm{NL}}} F_{\mathrm{IE}}^{i}}$ *returns the set of intermediate events attached to the boundary of a transaction; and*

— $\Phi_{\mathrm{A}}^{\mathrm{Att}} : F_{\mathrm{A}} \times S_{\mathrm{A}}^{\mathrm{Att}} \rightarrow S_{\mathrm{A}}^{\mathrm{AttV}}$ *returns the activity attribute value of an activity and an activity attribute.*

A task tuple, a subprocess tuple and a number of functions constitute an activity tuple. None intermediate events and link intermediate events cannot be attached to the boundary of an activity or a transaction. Additionally, cancel intermediate events are restricted to be placed on the boundary of a transaction.

We now present four definitions that specify formally the concepts of exclusive gateways, inclusive gateways, complex gateways and parallel gateways. The notion of gateways is then defined by means of these definitions.

Definition 8 (Exclusive-gateway Tuple) *An exclusive-gateway tuple is a 4-tuple* $\Omega_{\mathrm{XG}} = (F_{\mathrm{XDG}}^{\mathrm{D}}, F_{\mathrm{XMG}}^{\mathrm{D}}, F_{\mathrm{XDG}}^{\mathrm{E}}, F_{\mathrm{XMG}}^{\mathrm{E}})$ *where*

— $F_{\mathrm{XDG}}^{\mathrm{D}}$ *is a set of data-based exclusive decision gateways (DXDGs);*

— $F^{\mathrm{D}}_{\mathrm{XMG}}$ *is a set of data-based exclusive merge gateways (DXMGs);*

— $F^{\mathrm{E}}_{\mathrm{XDG}}$ *is a set of event-based exclusive decision gateways (EXDGs); and*

— $F^{\mathrm{E}}_{\mathrm{XMG}}$ *is a set of event-based exclusive merge gateways (EXMGs).*

Sets of data-based exclusive decision gateways, data-based exclusive merge gateways, event-based exclusive decision gateways and event-based exclusive merge gateways form an exclusive-gateway tuple.

Definition 9 (Inclusive-gateway Tuple) *An inclusive-gateway tuple is a 2-tuple* $\Omega_{\mathrm{IG}} = (F_{\mathrm{IDG}}, F_{\mathrm{IMG}})$ *where*

— F_{IDG} *is a set of inclusive decision gateways (IDGs); and*

— F_{IMG} *is a set of inclusive merge gateways (IMGs).*

Definition 10 (Complex-gateway Tuple) *A complex-gateway tuple is a 2-tuple* $\Omega_{\mathrm{CG}} = (F_{\mathrm{CDG}}, F_{\mathrm{CMG}})$ *where*

— F_{CDG} *is a set of complex decision gateways (CDGs); and*

— F_{CMG} *is a set of complex merge gateways (CMGs).*

Definition 11 (Parallel-gateway Tuple) *A parallel-gateway tuple is a 2-tuple* $\Omega_{\mathrm{PG}} = (F_{\mathrm{PFG}}, F_{\mathrm{PJG}})$ *where*

— F_{PFG} *is a set of parallel fork gateways (PFGs); and*

— F_{PJG} *is a set of parallel join gateways (PJGs).*

Definition 12 (Gateway Tuple) *Suppose* $\Gamma_{\mathrm{XG}} = \{\mathrm{XDG}, \mathrm{XMG}\}$, $\Gamma_{\mathrm{IG}} = \{\mathrm{IDG}, \mathrm{IMG}\}$, $\Gamma_{\mathrm{CG}} = \{\mathrm{CDG}, \mathrm{CMG}\}$, $\Gamma_{\mathrm{PG}} = \{\mathrm{PFG}, \mathrm{PJG}\}$, $F_{\mathrm{XG}} = \bigcup_{i\in\{\mathrm{D,E}\}} \bigcup_{j\in\Gamma_{\mathrm{XG}}} F^i_j$, $F_{\mathrm{IG}} = \bigcup_{i\in\Gamma_{\mathrm{IG}}} F_i$, $F_{\mathrm{CG}} = \bigcup_{i\in\Gamma_{\mathrm{CG}}} F_i$, $F_{\mathrm{PG}} = \bigcup_{i\in\Gamma_{\mathrm{PG}}} F_i$, $F_{\mathrm{G}} = \bigcup_{i\in\{\mathrm{XG,IG,CG,PG}\}} F_i$, $S^{\mathrm{Att}}_{\mathrm{G}}$ *is a set of gateway attributes and* $S^{\mathrm{AttV}}_{\mathrm{G}}$ *is a set of gateway attribute values. A gateway tuple is a 5-tuple* $\Omega_{\mathrm{G}} = (\Omega_{\mathrm{XG}}, \Omega_{\mathrm{IG}}, \Omega_{\mathrm{CG}}, \Omega_{\mathrm{PG}}, \Phi^{\mathrm{Att}}_{\mathrm{G}})$ *where*

— Ω_{XG} *is an exclusive-gateway tuple;*

— Ω_{IG} *is an inclusive-gateway tuple;*

— Ω_{CG} *is a complex-gateway tuple;*

— Ω_{PG} *is a parallel-gateway tuple; and*

— $\Phi^{\mathrm{Att}}_{\mathrm{G}}$*:* $F_{\mathrm{G}} \times S^{\mathrm{Att}}_{\mathrm{G}} \rightarrow S^{\mathrm{AttV}}_{\mathrm{G}}$ *defines for a gateway and a gateway attribute the corresponding gateway attribute value.*

The two types of inclusive gateways are: inclusive decision gateways and inclusive merge gateways. There are two sorts of complex gateways: complex decision gateways and complex merge gateways. A parallel-gateway tuple is composed of sets of parallel fork gateways and parallel join gateways. By combining an exclusive-gateway tuple, an inclusive-gateway tuple, a complex-gateway tuple, a parallel-gateway tuple and a function $\Phi_{\mathrm{G}}^{\mathrm{Att}}$, a gateway tuple is obtained.

In what follows, a definition for connecting-object tuple is offered. A process is then defined in terms of events, activities, gateways and connecting objects.

Definition 13 (Connecting-object Tuple) *Suppose* $\Gamma_{\mathrm{IE}}^{\mathrm{src}} = \{\mathrm{Msg}, \overline{\mathrm{Msg}}, \mathrm{Timer}, \mathrm{Cond}, \mathrm{Link}, \overline{\mathrm{Link}}, \mathrm{Sign}, \overline{\mathrm{Sign}}\}$, $\Gamma_{\mathrm{IE}}^{\mathrm{trg}} = \{\mathrm{None}, \mathrm{Msg}, \overline{\mathrm{Msg}}, \mathrm{Timer}, \mathrm{Cmpen}, \overline{\mathrm{Cmpen}}, \mathrm{Cond}, \mathrm{Link}, \overline{\mathrm{Link}}, \mathrm{Sign}, \overline{\mathrm{Sign}}\}$, $S_{\mathrm{F}} = \bigcup_{i\in\{\mathrm{E,A,G}\}} F_i$, $S_{\mathrm{IE}}^{\mathrm{Bdy[-Cmpen]}} = \bigcup_{A\in(F_{\mathrm{T}}\cup\bigcup_{i\in\Gamma_{\mathrm{SP}}} F_{\mathrm{SP}}^i\setminus S_{\mathrm{TX}})} \Phi_{\mathrm{IE}}^{\mathrm{Bdy[-TX]}}(A) \cup \bigcup_{Tx\in S_{\mathrm{TX}}} \Phi_{\mathrm{IE}}^{\mathrm{Bdy[TX]}}(Tx) \setminus F_{\mathrm{IE}}^{\mathrm{Cmpen}}$ *is the set of non-compensation intermediate events attached to activities and transactions,* $S_{\mathrm{A}}^{\mathrm{Cmpen}} = \{x | (x \in F_{\mathrm{T}} \wedge M_{\mathrm{C}} \in \Phi_{\mathrm{TM}}(x)) \vee (x \in \bigcup_{i\in\Gamma_{\mathrm{SP}}} F_{\mathrm{SP}}^i \wedge M_{\mathrm{C}} \in \Phi_{\mathrm{SPM}}(x))\}$ *is the set of activities with the compensation marker,* $S_{\mathrm{IE}}^{\mathrm{NF[src]}} = \bigcup_{i\in\Gamma_{\mathrm{IE}}^{\mathrm{src}}} F_{\mathrm{IE}}^i \setminus (\bigcup_{A\in(F_{\mathrm{T}}\cup\bigcup_{i\in\Gamma_{\mathrm{SP}}} F_{\mathrm{SP}}^i\setminus S_{\mathrm{TX}})} \Phi_{\mathrm{IE}}^{\mathrm{Bdy[-TX]}}(A) \cup \bigcup_{Tx\in S_{\mathrm{TX}}} \Phi_{\mathrm{IE}}^{\mathrm{Bdy[TX]}}(Tx))$ *is the set of intermediate events that are sources of normal or uncontrolled flows,* $S_{\mathrm{IE}}^{\mathrm{NF[trg]}} = \bigcup_{i\in\Gamma_{\mathrm{IE}}^{\mathrm{trg}}} F_{\mathrm{IE}}^i \setminus (\bigcup_{A\in(F_{\mathrm{T}}\cup\bigcup_{i\in\Gamma_{\mathrm{SP}}} F_{\mathrm{SP}}^i\setminus S_{\mathrm{TX}})} \Phi_{\mathrm{IE}}^{\mathrm{Bdy[-TX]}}(A) \cup \bigcup_{Tx\in S_{\mathrm{TX}}} \Phi_{\mathrm{IE}}^{\mathrm{Bdy[TX]}}(Tx))$ *is the set of intermediate events that are targets of normal or uncontrolled flows,* $S_{\mathrm{C}}^{\mathrm{Att}}$ *is a set of connecting object attributes and* $S_{\mathrm{C}}^{\mathrm{AttV}}$ *is a set of connecting object attribute values. A connecting-object tuple is a 7-tuple* $\Omega_{\mathrm{C}} = (A_{\mathrm{DO}}, C_{\mathrm{SF}}, C_{\mathrm{DA}}, S_{\mathrm{Cond}}, \Phi_{\mathrm{Cond}}, \Phi_{\mathrm{IsDf}}, \Phi_{\mathrm{C}}^{\mathrm{Att}})$ *where*

- A_{DO} *is a set of data objects;*
- $C_{\mathrm{SF}} \subseteq (S_{\mathrm{F}} \setminus (F_{\mathrm{EE}} \cup S_{\mathrm{A}}^{\mathrm{Cmpen}} \cup F_{\mathrm{IE}}) \cup (\bigcup_{P\in F_{\mathrm{SP}}^{\mathrm{Embed}}} \Phi_{\mathrm{EE}}^{\mathrm{Bdy}}(P) \cup S_{\mathrm{IE}}^{\mathrm{Bdy[-Cmpen]}} \cup S_{\mathrm{IE}}^{\mathrm{NF[src]}})) \times (S_{\mathrm{F}} \setminus (F_{\mathrm{SE}} \cup F_{\mathrm{IE}}) \cup (\bigcup_{P\in F_{\mathrm{SP}}^{\mathrm{Embed}}} \Phi_{\mathrm{SE}}^{\mathrm{Bdy}}(P) \cup S_{\mathrm{IE}}^{\mathrm{NF[trg]}}))$ *is a set of sequence flows (SFs);*
- $C_{\mathrm{DA}} \subseteq (F_{\mathrm{A}} \times A_{\mathrm{DO}}) \cup (A_{\mathrm{DO}} \times F_{\mathrm{A}}) \cup (F_{\mathrm{IE}}^{\mathrm{Cmpen}} \times F_{\mathrm{A}})$ *is a set of directed associations;*
- S_{Cond} *is a set of conditions;*
- $\Phi_{\mathrm{Cond}} : C_{\mathrm{SF}} \to S_{\mathrm{Cond}}$ *returns the condition of a sequence flow;*
- $\Phi_{\mathrm{IsDf}} : C_{\mathrm{SF}} \to \mathbb{B}$ *returns whether a sequence flow is a default sequence flow; and*
- $\Phi_{\mathrm{C}}^{\mathrm{Att}} : \bigcup_{i\in\{\mathrm{SF,DA}\}} C_i \times S_{\mathrm{C}}^{\mathrm{Att}} \to S_{\mathrm{C}}^{\mathrm{AttV}}$ *relates a connecting object and a connecting object attribute to a connecting object attribute value.*

In Definition 13, the expression $(S_{\mathrm{F}} \setminus (F_{\mathrm{EE}} \cup S_{\mathrm{A}}^{\mathrm{Cmpen}} \cup F_{\mathrm{IE}}) \cup (\bigcup_{P \in F_{\mathrm{SP}}^{\mathrm{Embed}}} \Phi_{\mathrm{EE}}^{\mathrm{Bdy}}(P) \cup S_{\mathrm{IE}}^{\mathrm{Bdy}[-\mathrm{Cmpen}]} \cup S_{\mathrm{IE}}^{\mathrm{NF}[\mathrm{src}]}))$ states that

(i) an end event cannot be a source flow object with the exception that it is attached to the boundary of an expanded subprocess; and

(ii) a compensation activity does not have any outgoing sequence flows.

In a similar way, the expression $(S_{\mathrm{F}} \setminus (F_{\mathrm{SE}} \cup F_{\mathrm{IE}}) \cup (\bigcup_{P \in F_{\mathrm{SP}}^{\mathrm{Embed}}} \Phi_{\mathrm{SE}}^{\mathrm{Bdy}}(P) \cup S_{\mathrm{IE}}^{\mathrm{NF}[\mathrm{trg}]}))$ says that a start event cannot be a target flow object except it is attached to the boundary of an expanded subprocess. A sequence flow is a subset of the cross product of these two expressions. The expression $(F_{\mathrm{A}} \times A_{\mathrm{DO}}) \cup (A_{\mathrm{DO}} \times F_{\mathrm{A}}) \cup (S_{\mathrm{IE}}^{\mathrm{Cmpen}} \times F_{\mathrm{A}})$ stipulates that a directed association connects

(i) a data object with an activity; or

(ii) a compensation intermediate event for catching the event trigger with an activity.

Definition 14 (Process) *A process is a 4-tuple* $P = (\Omega_{\mathrm{E}}, \Omega_{\mathrm{A}}, \Omega_{\mathrm{G}}, \Omega_{\mathrm{C}})$ *where*

— Ω_{E} *is an event tuple;*

— Ω_{A} *is an activity tuple;*

— Ω_{G} *is a gateway tuple; and*

— Ω_{C} *is a connecting-object tuple.*

A process consists of four components: an event tuple, an activity tuple, a gateway tuple and a connecting-object tuple.

Definition 15 (None-start-events Process) *Given a process* P *with a start-event tuple* $\Omega_{\mathrm{SE}} = (F_{\mathrm{SE}}^{\mathrm{None}}, F_{\mathrm{SE}}^{\mathrm{Msg}}, F_{\mathrm{SE}}^{\mathrm{Timer}}, F_{\mathrm{SE}}^{\mathrm{Cond}}, F_{\mathrm{SE}}^{\mathrm{Sign}}, F_{\mathrm{SE}}^{\mathrm{Multi}})$. *The process* P *is a none-start-events process if and only if* $\bigwedge_{i \in \Gamma_{\mathrm{SE}} \setminus \{\mathrm{None}\}} (F_{\mathrm{SE}}^{i} = \emptyset)$.

A none-start-events process is a process that contains solely none start events.

Definition 16 *The function* Φ_{TP}*, defined below, returns the task name, none-start-events process, called process or referenced subprocess depending on whether the parameter is a task, an embedded subprocess, a reusable subprocess or a reference subprocess.*

$$\Phi_{\mathrm{TP}}(x) = \begin{cases} \Phi_{\mathrm{TName}}(x) & \text{if } x \in F_{\mathrm{T}} \\ \Phi_{\mathrm{NP}}(x) & \text{if } x \in F_{\mathrm{SP}}^{\mathrm{Embed}} \\ \Phi_{\mathrm{P}}(x) & \text{if } x \in F_{\mathrm{SP}}^{\mathrm{Reuse}} \\ \Phi_{\mathrm{RP}}(x) & \text{if } x \in F_{\mathrm{SP}}^{\mathrm{Ref}}. \end{cases}$$

Definition 17 (Equivalence Class 29) *Suppose R is an equivalence relation on a set S and $s \in S$. An equivalence class of s, written $[s]_R$, is defined by*

$$[s]_R = \{x \in S | (x, s) \in R\}.$$

An equivalence relation, which is a binary relation, divides a set into equivalence classes which are pairwise disjoint. Each element of the set is a member of solely one equivalence class. All elements of an equivalence class are regarded as equivalent.

Definition 18 *Suppose a process $P = (\Omega_{\mathrm{E}}, \Omega_{\mathrm{A}}, \Omega_{\mathrm{G}}, \Omega_{\mathrm{C}})$, $\Omega_{\mathrm{E}} = (\Omega_{\mathrm{SE}}, \Omega_{\mathrm{IE}}, \Omega_{\mathrm{EE}}, \Phi_{\mathrm{E}}^{\mathrm{Att}})$, $\Omega_{\mathrm{A}} = (\Omega_{\mathrm{T}}, \Omega_{\mathrm{SP}}, \Phi_{\mathrm{IE}}^{\mathrm{Bdy[-TX]}}, \Phi_{\mathrm{IE}}^{\mathrm{Bdy[TX]}}, \Phi_{\mathrm{A}}^{\mathrm{Att}})$, $\Omega_{\mathrm{G}} = (\Omega_{\mathrm{XG}}, \Omega_{\mathrm{IG}}, \Omega_{\mathrm{CG}}, \Omega_{\mathrm{PG}}, \Phi_{\mathrm{G}}^{\mathrm{Att}})$, $\Omega_{\mathrm{C}} = (A_{\mathrm{DO}}, C_{\mathrm{SF}}, C_{\mathrm{DA}}, S_{\mathrm{Cond}}, \Phi_{\mathrm{Cond}}, \Phi_{\mathrm{IsDf}}, \Phi_{\mathrm{C}}^{\mathrm{Att}})$, $\Omega_{\mathrm{SE}} = (F_{\mathrm{SE}}^{\mathrm{None}}, F_{\mathrm{SE}}^{\mathrm{Msg}}, F_{\mathrm{SE}}^{\mathrm{Timer}}, F_{\mathrm{SE}}^{\mathrm{Cond}}, F_{\mathrm{SE}}^{\mathrm{Sign}}, F_{\mathrm{SE}}^{\mathrm{Multi}})$, $\Omega_{\mathrm{IE}} = (F_{\mathrm{IE}}^{\mathrm{None}}, F_{\mathrm{IE}}^{\mathrm{Msg}}, F_{\mathrm{IE}}^{\overline{\mathrm{Msg}}}, F_{\mathrm{IE}}^{\mathrm{Timer}}, F_{\mathrm{IE}}^{\mathrm{Err}}, F_{\mathrm{IE}}^{\mathrm{Cncl}}, F_{\mathrm{IE}}^{\mathrm{Cmpen}}, F_{\mathrm{IE}}^{\overline{\mathrm{Cmpen}}}, F_{\mathrm{IE}}^{\mathrm{Cond}}, F_{\mathrm{IE}}^{\mathrm{Link}}, F_{\mathrm{IE}}^{\overline{\mathrm{Link}}}, F_{\mathrm{IE}}^{\mathrm{Sign}}, F_{\mathrm{IE}}^{\overline{\mathrm{Sign}}}, F_{\mathrm{IE}}^{\mathrm{Multi}}, F_{\mathrm{IE}}^{\overline{\mathrm{Multi}}})$, $\Omega_{\mathrm{EE}} = (F_{\mathrm{EE}}^{\overline{\mathrm{None}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Msg}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Err}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Cncl}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Cmpen}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Sign}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Term}}}, F_{\mathrm{EE}}^{\overline{\mathrm{Multi}}})$, $\Omega_{\mathrm{T}} = (F_{\mathrm{T}}, \Phi_{\mathrm{TM}}, \Phi_{\mathrm{Ttype}}, \Phi_{\mathrm{TName}})$, $\Omega_{\mathrm{SP}} = (F_{\mathrm{SP}}^{\mathrm{Embed}}, F_{\mathrm{SP}}^{\mathrm{Reuse}}, F_{\mathrm{SP}}^{\mathrm{Ref}}, \Phi_{\mathrm{IsTX}}, \Phi_{\mathrm{SPM}}, \Phi_{\mathrm{SE}}^{\mathrm{Bdy}}, \Phi_{\mathrm{EE}}^{\mathrm{Bdy}}, \Phi_{\mathrm{NP}}, \Phi_{\mathrm{P}}, \Phi_{\mathrm{RP}})$, $\Omega_{\mathrm{XG}} = (F_{\mathrm{XDG}}^{\mathrm{D}}, F_{\mathrm{XMG}}^{\mathrm{D}}, F_{\mathrm{XDG}}^{\mathrm{E}}, F_{\mathrm{XMG}}^{\mathrm{E}})$, $\Omega_{\mathrm{IG}} = (F_{\mathrm{IDG}}, F_{\mathrm{IMG}})$, $\Omega_{\mathrm{CG}} = (F_{\mathrm{CDG}}, F_{\mathrm{CMG}})$, $\Omega_{\mathrm{PG}} = (F_{\mathrm{PFG}}, F_{\mathrm{PJG}})$, $domain_{\mathrm{SF}}(C_{\mathrm{SF}}) = S_{\mathrm{F}} \setminus (F_{\mathrm{EE}} \cup S_{\mathrm{A}}^{\mathrm{Cmpen}} \cup F_{\mathrm{IE}}) \cup (\bigcup_{P \in F_{\mathrm{SP}}^{\mathrm{Embed}}} \Phi_{\mathrm{EE}}^{\mathrm{Bdy}}(P) \cup S_{\mathrm{IE}}^{\mathrm{Bdy[-Cmpen]}} \cup S_{\mathrm{IE}}^{\mathrm{NF[src]}})$, $range_{\mathrm{SF}}(C_{\mathrm{SF}}) = S_{\mathrm{F}} \setminus (F_{\mathrm{SE}} \cup F_{\mathrm{IE}}) \cup (\bigcup_{P \in F_{\mathrm{SP}}^{\mathrm{Embed}}} \Phi_{\mathrm{SE}}^{\mathrm{Bdy}}(P) \cup S_{\mathrm{IE}}^{\mathrm{NF[trg]}})$, $domain_{\mathrm{DA}}(C_{\mathrm{DA}}) = F_{\mathrm{A}} \cup A_{\mathrm{DO}} \cup F_{\mathrm{IE}}^{\mathrm{Cmpen}}$ and $range_{\mathrm{DA}}(C_{\mathrm{DA}}) = A_{\mathrm{DO}} \cup F_{\mathrm{A}}$. The functions $source_{\mathrm{SF}} : C_{\mathrm{SF}} \to domain_{\mathrm{SF}}(C_{\mathrm{SF}})$ and $target_{\mathrm{SF}} : C_{\mathrm{SF}} \to range_{\mathrm{SF}}(C_{\mathrm{SF}})$ relate a sequence flow $(o_1, o_2) \in C_{\mathrm{SF}}$ to o_1 and o_2, respectively. The functions $source_{\mathrm{DA}} : C_{\mathrm{DA}} \to domain_{\mathrm{DA}}(C_{\mathrm{DA}})$ and $target_{\mathrm{DA}} : C_{\mathrm{DA}} \to range_{\mathrm{DA}}(C_{\mathrm{DA}})$ map a directed association $(a_1, a_2) \in C_{\mathrm{DA}}$, respectively, to a_1 and a_2.*

The functions $source_{\mathrm{SF}}$ and $target_{\mathrm{SF}}$ return, respectively, the source and target flow objects of a sequence flow. Likewise, the functions $source_{\mathrm{DA}}$ and $target_{\mathrm{DA}}$ determine the source and target objects of a directed association.

The formal definition that follows specifies how a process is partitioned. With the concept of partitioned process in place, the notions lane and pool are then defined subsequently.

Definition 19 (Partitioned Process) *Suppose a process $P = (\Omega_{\mathrm{E}}, \Omega_{\mathrm{A}}, \Omega_{\mathrm{G}}, \Omega_{\mathrm{C}})$. Assume that Ω_{E}, Ω_{A}, Ω_{G} and Ω_{C} are defined in Definition 18 and equivalence relations R_{DO}, R_{SF}, R_{DA}, R_{Cond}, $R_{\mathrm{SE}}^{\mathrm{None}}$, $R_{\mathrm{SE}}^{\mathrm{Msg}}$, $R_{\mathrm{SE}}^{\mathrm{Timer}}$, $R_{\mathrm{SE}}^{\mathrm{Cond}}$, $R_{\mathrm{SE}}^{\mathrm{Sign}}$, $R_{\mathrm{SE}}^{\mathrm{Multi}}$, $R_{\mathrm{IE}}^{\mathrm{None}}$, $R_{\mathrm{IE}}^{\mathrm{Msg}}$, $R_{\mathrm{IE}}^{\overline{\mathrm{Msg}}}$, $R_{\mathrm{IE}}^{\mathrm{Timer}}$, $R_{\mathrm{IE}}^{\mathrm{Err}}$, $R_{\mathrm{IE}}^{\mathrm{Cncl}}$, $R_{\mathrm{IE}}^{\mathrm{Cmpen}}$, $R_{\mathrm{IE}}^{\overline{\mathrm{Cmpen}}}$, $R_{\mathrm{IE}}^{\mathrm{Cond}}$, $R_{\mathrm{IE}}^{\mathrm{Link}}$, $R_{\mathrm{IE}}^{\overline{\mathrm{Link}}}$, $R_{\mathrm{IE}}^{\mathrm{Sign}}$, $R_{\mathrm{IE}}^{\overline{\mathrm{Sign}}}$, $R_{\mathrm{IE}}^{\mathrm{Multi}}$, $R_{\mathrm{IE}}^{\overline{\mathrm{Multi}}}$, $R_{\mathrm{EE}}^{\overline{\mathrm{None}}}$, $R_{\mathrm{EE}}^{\overline{\mathrm{Msg}}}$, $R_{\mathrm{EE}}^{\overline{\mathrm{Err}}}$, $R_{\mathrm{EE}}^{\overline{\mathrm{Cncl}}}$, $R_{\mathrm{EE}}^{\overline{\mathrm{Cmpen}}}$, $R_{\mathrm{EE}}^{\overline{\mathrm{Sign}}}$, $R_{\mathrm{EE}}^{\overline{\mathrm{Term}}}$, $R_{\mathrm{EE}}^{\overline{\mathrm{Multi}}}$, R_{T}, $R_{\mathrm{SP}}^{\mathrm{Embed}}$, $R_{\mathrm{SP}}^{\mathrm{Reuse}}$, $R_{\mathrm{SP}}^{\mathrm{Ref}}$, $R_{\mathrm{XDG}}^{\mathrm{D}}$, $R_{\mathrm{XMG}}^{\mathrm{D}}$, $R_{\mathrm{XDG}}^{\mathrm{E}}$, $R_{\mathrm{XMG}}^{\mathrm{E}}$, R_{IDG}, R_{IMG}, R_{CDG}, R_{CMG},*

R_{PFG}, R_{PJG} *on* A_{DO}, C_{SF}, C_{DA}, S_{Cond}, $F^{\mathrm{None}}_{\mathrm{SE}}$, $F^{\mathrm{Msg}}_{\mathrm{SE}}$, $F^{\mathrm{Timer}}_{\mathrm{SE}}$, $F^{\mathrm{Cond}}_{\mathrm{SE}}$, $F^{\mathrm{Sign}}_{\mathrm{SE}}$, $F^{\mathrm{Multi}}_{\mathrm{SE}}$, $F^{\mathrm{None}}_{\mathrm{IE}}$, $F^{\mathrm{Msg}}_{\mathrm{IE}}$, $F^{\overline{\mathrm{Msg}}}_{\mathrm{IE}}$, $F^{\mathrm{Timer}}_{\mathrm{IE}}$, $F^{\mathrm{Err}}_{\mathrm{IE}}$, $F^{\mathrm{Cncl}}_{\mathrm{IE}}$, $F^{\mathrm{Cmpen}}_{\mathrm{IE}}$, $F^{\overline{\mathrm{Cmpen}}}_{\mathrm{IE}}$, $F^{\mathrm{Cond}}_{\mathrm{IE}}$, $F^{\mathrm{Link}}_{\mathrm{IE}}$, $F^{\overline{\mathrm{Link}}}_{\mathrm{IE}}$, $F^{\mathrm{Sign}}_{\mathrm{IE}}$, $F^{\overline{\mathrm{Sign}}}_{\mathrm{IE}}$, $F^{\mathrm{Multi}}_{\mathrm{IE}}$, $F^{\overline{\mathrm{Multi}}}_{\mathrm{IE}}$, $F^{\overline{\mathrm{None}}}_{\mathrm{EE}}$, $F^{\overline{\mathrm{Msg}}}_{\mathrm{EE}}$, $F^{\overline{\mathrm{Err}}}_{\mathrm{EE}}$, $F^{\overline{\mathrm{Cncl}}}_{\mathrm{EE}}$, $F^{\overline{\mathrm{Cmpen}}}_{\mathrm{EE}}$, $F^{\overline{\mathrm{Sign}}}_{\mathrm{EE}}$, $F^{\overline{\mathrm{Term}}}_{\mathrm{EE}}$, $F^{\overline{\mathrm{Multi}}}_{\mathrm{EE}}$, F_{T}, $F^{\mathrm{Embed}}_{\mathrm{SP}}$, $F^{\mathrm{Reuse}}_{\mathrm{SP}}$, $F^{\mathrm{Ref}}_{\mathrm{SP}}$, $F^{\mathrm{D}}_{\mathrm{XDG}}$, $F^{\mathrm{D}}_{\mathrm{XMG}}$, $F^{\mathrm{E}}_{\mathrm{XDG}}$, $F^{\mathrm{E}}_{\mathrm{XMG}}$, F_{IDG}, F_{IMG}, F_{CDG}, F_{CMG}, F_{PFG}, F_{PJG}. *The process* P *is a partitioned process.*

Definition 20 (Lane-start-event Tuple) *Suppose a process* P *is a partitioned process. Assume that* Γ_{SE} *is defined in Definition 4. A lane-start-event tuple is a 6-tuple* $\omega_{\mathrm{SE}} = (FF^{\mathrm{None}}_{\mathrm{SE}}, FF^{\mathrm{Msg}}_{\mathrm{SE}}, FF^{\mathrm{Timer}}_{\mathrm{SE}}, FF^{\mathrm{Cond}}_{\mathrm{SE}}, FF^{\mathrm{Sign}}_{\mathrm{SE}}, FF^{\mathrm{Multi}}_{\mathrm{SE}})$ *where* $\bigwedge_{i\in\Gamma_{\mathrm{SE}}}(FF^{i}_{\mathrm{SE}} \subseteq F^{i}_{\mathrm{SE}} \wedge \forall\sigma_1,\sigma_2 \in FF^{i}_{\mathrm{SE}}.[\sigma_1]_{R^{i}_{\mathrm{SE}}} = [\sigma_2]_{R^{i}_{\mathrm{SE}}})$.

Definition 21 (Lane-intermediate-event Tuple) *Suppose a process* P *is a partitioned process. Assume that* Γ_{IE} *and* $\Gamma_{\overline{\mathrm{IE}}}$ *are defined in Definition 4. A lane-intermediate-event tuple is a 15-tuple* $\omega_{\mathrm{IE}} = (FF^{\mathrm{None}}_{\mathrm{IE}}, FF^{\mathrm{Msg}}_{\mathrm{IE}}, FF^{\overline{\mathrm{Msg}}}_{\mathrm{IE}}, FF^{\mathrm{Timer}}_{\mathrm{IE}}, FF^{\mathrm{Err}}_{\mathrm{IE}}, FF^{\mathrm{Cncl}}_{\mathrm{IE}}, FF^{\mathrm{Cmpen}}_{\mathrm{IE}}, FF^{\overline{\mathrm{Cmpen}}}_{\mathrm{IE}}, FF^{\mathrm{Cond}}_{\mathrm{IE}}, FF^{\mathrm{Link}}_{\mathrm{IE}}, FF^{\overline{\mathrm{Link}}}_{\mathrm{IE}}, FF^{\mathrm{Sign}}_{\mathrm{IE}}, FF^{\overline{\mathrm{Sign}}}_{\mathrm{IE}}, FF^{\mathrm{Multi}}_{\mathrm{IE}}, FF^{\overline{\mathrm{Multi}}}_{\mathrm{IE}})$ *where* $\bigwedge_{i\in\Gamma_{\mathrm{IE}}\cup\Gamma_{\overline{\mathrm{IE}}}}(FF^{i}_{\mathrm{IE}} \subseteq F^{i}_{\mathrm{IE}} \wedge \forall\sigma_1, \sigma_2 \in FF^{i}_{\mathrm{IE}}.\ [\sigma_1]_{R^{i}_{\mathrm{IE}}} = [\sigma_2]_{R^{i}_{\mathrm{IE}}})$.

Definition 22 (Lane-end-event Tuple) *Suppose a process* P *is a partitioned process. Assume that* Γ_{EE} *is defined in Definition 4. A lane-end-event tuple is a 8-tuple* $\omega_{\mathrm{EE}} = (FF^{\overline{\mathrm{None}}}_{\mathrm{EE}}, FF^{\overline{\mathrm{Msg}}}_{\mathrm{EE}}, FF^{\overline{\mathrm{Err}}}_{\mathrm{EE}}, FF^{\overline{\mathrm{Cncl}}}_{\mathrm{EE}}, FF^{\overline{\mathrm{Cmpen}}}_{\mathrm{EE}}, FF^{\overline{\mathrm{Sign}}}_{\mathrm{EE}}, FF^{\overline{\mathrm{Term}}}_{\mathrm{EE}}, FF^{\overline{\mathrm{Multi}}}_{\mathrm{EE}})$ *where* $\bigwedge_{i\in\Gamma_{\mathrm{EE}}}(FF^{\bar{i}}_{\mathrm{EE}} \subseteq F^{\bar{i}}_{\mathrm{EE}} \wedge \forall\sigma_1,\sigma_2 \in FF^{i}_{\mathrm{EE}}.[\sigma_1]_{R^{i}_{\mathrm{EE}}} = [\sigma_2]_{R^{i}_{\mathrm{EE}}})$.

Definition 23 (Lane-event Tuple) *Suppose a process* P *is a partitioned process. A lane-event tuple is a 4-tuple* $\omega_{\mathrm{E}} = (\omega_{\mathrm{SE}}, \omega_{\mathrm{IE}}, \omega_{\mathrm{EE}}, \Phi^{\mathrm{Att}}_{\mathrm{E}})$ *where*

— ω_{SE} *is a lane-start-event tuple;*

— ω_{IE} *is a lane-intermediate-event tuple; and*

— ω_{EE} *is a lane-end-event tuple.*

A lane-start-event tuple relies on a start-event tuple in which every element of the lane-start-event tuple is a subset of the respective element of the start-event tuple. Likewise, a lane-intermediate-event tuple and a lane-end-event tuple relate, respectively, to an intermediate-event tuple and an end-event tuple. A lane-start-event tuple, a lane-intermediate-event tuple and a lane-end-event tuple and the function $\Phi^{\mathrm{Att}}_{\mathrm{E}}$ make up a lane-event tuple.

Definition 24 (Lane-task Tuple) *Suppose a process* P *is a partitioned process. A lane-task tuple is a 4-tuple* $\omega_{\mathrm{T}} = (FF_{\mathrm{T}}, \Phi_{\mathrm{TM}}, \Phi_{\mathrm{Ttype}}, \Phi_{\mathrm{TName}})$ *where* $FF_{\mathrm{T}} \subseteq F_{\mathrm{T}} \wedge \forall\sigma_1$, $\sigma_2 \in FF_{\mathrm{T}}.\ [\sigma_1]_{R_{\mathrm{T}}} = [\sigma_2]_{R_{\mathrm{T}}}$.

Definition 25 (Lane-subprocess Tuple) *Suppose a process P is a partitioned process and $\Gamma_{\mathrm{SP}} = \{\mathrm{Embed}, \mathrm{Reuse}, \mathrm{Ref}\}$. A lane-subprocess tuple is a 10-tuple $\omega_{\mathrm{SP}} = (FF_{\mathrm{SP}}^{\mathrm{Embed}}, FF_{\mathrm{SP}}^{\mathrm{Reuse}}, FF_{\mathrm{SP}}^{\mathrm{Ref}}, \Phi_{\mathrm{IsTX}}, \Phi_{\mathrm{SPM}}, \Phi_{\mathrm{SE}}^{\mathrm{Bdy}}, \Phi_{\mathrm{EE}}^{\mathrm{Bdy}}, \Phi_{\mathrm{NP}}, \Phi_{\mathrm{P}}, \Phi_{\mathrm{RP}})$ where $\bigwedge_{i \in \Gamma_{\mathrm{SP}}} (FF_{\mathrm{SP}}^{i} \subseteq F_{\mathrm{SP}}^{i} \wedge \forall \sigma_1, \sigma_2 \in FF_{\mathrm{SP}}^{i}.\ [\sigma_1]_{R_{\mathrm{SP}}^{i}} = [\sigma_2]_{R_{\mathrm{SP}}^{i}})$.*

Definition 26 (Lane-activity Tuple) *Suppose a process P is a partitioned process. A lane-activity tuple is a 5-tuple $\omega_{\mathrm{A}} = (\omega_{\mathrm{T}}, \omega_{\mathrm{SP}}, \Phi_{\mathrm{IE}}^{\mathrm{Bdy}[-\mathrm{TX}]}, \Phi_{\mathrm{IE}}^{\mathrm{Bdy}[\mathrm{TX}]}, \Phi_{\mathrm{A}}^{\mathrm{Att}})$ where*

— *ω_{T} is a lane-task tuple; and*

— *ω_{SP} is a lane-subprocess tuple.*

There are correspondences between lane-task tuple and task tuple as well as lane-subprocess tuple and subprocess tuple. A lane-activity tuple is defined in terms of a lane-task tuple and a lane-subprocess tuple.

Definition 27 (Lane-exclusive-gateway Tuple) *Suppose a process P is a partitioned process, $\Gamma_{\mathrm{XT}} = \{\mathrm{D}, \mathrm{E}\}$ and $\Gamma_{\mathrm{XG}} = \{\mathrm{XDG}, \mathrm{XMG}\}$. A lane-exclusive-gateway tuple is a 4-tuple $\omega_{\mathrm{XG}} = (FF_{\mathrm{XDG}}^{\mathrm{D}}, FF_{\mathrm{XMG}}^{\mathrm{D}}, FF_{\mathrm{XDG}}^{\mathrm{E}}, FF_{\mathrm{XMG}}^{\mathrm{E}})$ where $\bigwedge_{i \in \Gamma_{\mathrm{XT}}} \bigwedge_{j \in \Gamma_{\mathrm{XG}}} (FF_j^i \subseteq F_j^i \wedge \forall \sigma_1, \sigma_2 \in FF_j^i.[\sigma_1]_{R_j^i} = [\sigma_2]_{R_j^i})$.*

Definition 28 (Lane-inclusive-gateway Tuple) *Suppose a process P is a partitioned process and $\Gamma_{\mathrm{IG}} = \{\mathrm{IDG}, \mathrm{IMG}\}$. A lane-inclusive-gateway tuple is a 2-tuple $\omega_{\mathrm{IG}} = (FF_{\mathrm{IDG}}, FF_{\mathrm{IMG}})$ where $\bigwedge_{i \in \Gamma_{\mathrm{IG}}} (FF_i \subseteq F_i \wedge \forall \sigma_1, \sigma_2 \in FF_i.[\sigma_1]_{R_i} = [\sigma_2]_{R_i})$.*

Definition 29 (Lane-complex-gateway Tuple) *Suppose a process P is a partitioned process and $\Gamma_{\mathrm{CG}} = \{\mathrm{CDG}, \mathrm{CMG}\}$. A lane-complex-gateway tuple is a 2-tuple $\omega_{\mathrm{CG}} = (FF_{\mathrm{CDG}}, FF_{\mathrm{CMG}})$ where $\bigwedge_{i \in \Gamma_{\mathrm{CG}}} (FF_i \subseteq F_i \wedge \forall \sigma_1, \sigma_2 \in FF_i.[\sigma_1]_{R_i} = [\sigma_2]_{R_i})$.*

Definition 30 (Lane-parallel-gateway Tuple) *Suppose a process P is a partitioned process and $\Gamma_{\mathrm{PG}} = \{\mathrm{PFG}, \mathrm{PJG}\}$. A lane-parallel-gateway tuple is a 2-tuple $\omega_{\mathrm{PG}} = (FF_{\mathrm{PFG}}, FF_{\mathrm{PJG}})$ where $\bigwedge_{i \in \Gamma_{\mathrm{PG}}} (FF_i \subseteq F_i \wedge \forall \sigma_1, \sigma_2 \in FF_i.[\sigma_1]_{R_i} = [\sigma_2]_{R_i})$.*

Definition 31 (Lane-gateway Tuple) *Suppose a process P is a partitioned process. A lane-gateway tuple is a 5-tuple $\omega_{\mathrm{G}} = (\omega_{\mathrm{XG}}, \omega_{\mathrm{IG}}, \omega_{\mathrm{CG}}, \omega_{\mathrm{PG}}, \Phi_{\mathrm{G}}^{\mathrm{Att}})$ where*

— *ω_{XG} is a lane-exclusive gateway tuple;*

— *ω_{IG} is a lane-inclusive gateway tuple;*

— *ω_{CG} is a lane-complex-gateway tuple; and*

— *ω_{PG} is a lane-parallel-gateway tuple.*

Definition 32 (Lane-connecting-object Tuple) *Suppose a process P is a partitioned process and $\Gamma_{\mathrm{CO}} = \{\mathrm{SF}, \mathrm{DA}\}$. A lane-connecting-object tuple is a 7-tuple $\omega_{\mathrm{C}} = (AA_{\mathrm{DO}},\ CC_{\mathrm{SF}},\ CC_{\mathrm{DA}},\ SS_{\mathrm{Cond}},\ \Phi_{\mathrm{Cond}},\ \Phi_{\mathrm{IsDf}},\ \Phi_{\mathrm{C}}^{\mathrm{Att}})$ where $(AA_{\mathrm{DO}} \subseteq A_{\mathrm{DO}} \wedge \forall \sigma_1, \sigma_2 \in AA_{\mathrm{DO}}.[\sigma_1]_{R_{\mathrm{DO}}} = [\sigma_2]_{R_{\mathrm{DO}}}) \wedge \bigwedge_{i \in \Gamma_{\mathrm{CO}}} (CC_i \subseteq C_i \wedge \forall \sigma_1, \sigma_2 \in CC_i.\ [\sigma_1]_{R_i} = [\sigma_2]_{R_i}) \wedge (SS_{\mathrm{Cond}} \subseteq C_{\mathrm{Cond}} \wedge \forall \sigma_1, \sigma_2 \in SS_{\mathrm{Cond}}.[\sigma_1]_{R_{\mathrm{Cond}}} = [\sigma_2]_{R_{\mathrm{Cond}}})$.*

Analogously, we define a lane-exclusive-gateway tuple, a lane-inclusive-gateway tuple, a lane-complex-gateway tuple, a lane-parallel-gateway tuple, a lane-gateway tuple and a lane-connecting-object tuple based on the same principles.

Definition 33 (Lane Structure) *Suppose a process P is a partitioned process. A lane structure is a 4-tuple $LS = (\omega_{\mathrm{E}}, \omega_{\mathrm{A}}, \omega_{\mathrm{G}}, \omega_{\mathrm{C}})$ where*

- *ω_{E} is a lane-event tuple;*
- *ω_{A} is a lane-activity tuple;*
- *ω_{G} is a lane-gateway tuple; and*
- *ω_{C} is a lane-connecting-object tuple.*

A lane structure describes the way in which events, activities, gateways and connecting objects are connected together in a lane.

Definition 34 (Lane) *A lane is a 2-tuple $LANE = (S_{\mathrm{LNames}}, LS)$ where*

- *S_{LNames} is a set of lane names; and*
- *LS is a lane structure.*

Definition 35 (Pool) *A pool is a 2-tuple $POOL = (PName, S_{\mathrm{LANE}})$ where*

- *$PName$ is a pool name; and*
- *S_{LANE} is a set of lanes.*

As lanes can be nested, a lane is uniquely identified by a set of lane names. Each pool has a pool name and embodies a collection of lanes.

In the following, we end this section by providing a definition of a business process diagram.

Definition 36 (Business Process Diagram) *Let ε represents the empty string, $F^{\mathrm{Msg}}_{\mathrm{SE}^{(\mathrm{P}_i)}}$ be the set of message start events of process P_i for catching the event triggers, $F^{\mathrm{Msg}}_{\mathrm{IE}^{(\mathrm{P}_i)}}$ be the set of message intermediate events of process P_i for catching the event triggers, $F^{\overline{\mathrm{Msg}}}_{\mathrm{IE}^{(\mathrm{P}_i)}}$ be the set of message intermediate events of process P_i for throwing the event triggers, $F^{\overline{\mathrm{Msg}}}_{\mathrm{EE}^{(\mathrm{P}_i)}}$ be the set of message end events of process P_i for throwing the event triggers, $F_{A^{(P_i)}}$ be the set of activities of process P_i and $PName^{(\mathrm{POOL_i})}$ be the pool name of pool $POOL_i$ for $i = 1, 2$. A business process diagram is a 4-tuple $BPD = (S_{\mathrm{POOL}}, S_{\mathrm{P}}, \Phi_{\mathrm{POOL}\to\mathrm{P}}, C_{\mathrm{MF}})$ where*

— S_{POOL} *is a set of pools;*

— S_{P} *is a set of processes;*

— $\Phi_{\mathrm{POOL}\to\mathrm{P}} : S_{\mathrm{POOL}} \to S_{\mathrm{P}} \cup \{\varepsilon\}$ *relates a pool to a process; and*

— $C_{\mathrm{MF}} \subseteq \bigcup_{P_1,P_2\in S_{\mathrm{P}}}((F^{\overline{\mathrm{Msg}}}_{\mathrm{EE}^{(\mathrm{P}_1)}} \cup F^{\overline{\mathrm{Msg}}}_{\mathrm{IE}^{(\mathrm{P}_1)}} \cup F_{\mathrm{A}^{(\mathrm{P}_1)}})$
$\times(F^{\mathrm{Msg}}_{\mathrm{SE}^{(\mathrm{P}_2)}} \cup F^{\mathrm{Msg}}_{\mathrm{IE}^{(\mathrm{P}_2)}} \cup F_{\mathrm{A}^{(\mathrm{P}_2)}}))\cup$
$\bigcup_{\substack{P_1\in S_{\mathrm{P}}\\ POOL_2\in S_{\mathrm{POOL}}}}((F^{\overline{\mathrm{Msg}}}_{\mathrm{EE}^{(\mathrm{P}_1)}} \cup F^{\overline{\mathrm{Msg}}}_{\mathrm{IE}^{(\mathrm{P}_1)}} \cup F_{\mathrm{A}^{(\mathrm{P}_1)}}) \times \{PName^{(POOL_2)}\})\cup$
$\bigcup_{\substack{P_2\in S_{\mathrm{P}}\\ POOL_1\in S_{\mathrm{POOL}}}}(\{PName^{(POOL_1)}\} \times (F^{\mathrm{Msg}}_{\mathrm{SE}^{(\mathrm{P}_2)}} \cup F^{\mathrm{Msg}}_{\mathrm{IE}^{(\mathrm{P}_2)}} \cup F_{\mathrm{A}^{(\mathrm{P}_2)}}))\cup$
$\bigcup_{POOL_1,POOL_2\in S_{\mathrm{POOL}}}(\{PName^{(POOL_1)}\}\times\{PName^{(POOL_2)}\})$ *is a set of message flows such that* $\Phi_{\mathrm{POOL}\to\mathrm{P}}(POOL_i) = P_i$.

The expression $\bigcup_{P_1,P_2\in S_{\mathrm{P}}}((F^{\overline{\mathrm{Msg}}}_{\mathrm{EE}^{(\mathrm{P}_1)}} \cup F^{\overline{\mathrm{Msg}}}_{\mathrm{IE}^{(\mathrm{P}_1)}} \cup F_{\mathrm{A}^{(\mathrm{P}_1)}}) \times (F^{\mathrm{Msg}}_{\mathrm{SE}^{(\mathrm{P}_2)}} \cup F^{\mathrm{Msg}}_{\mathrm{IE}^{(\mathrm{P}_2)}} \cup F_{\mathrm{A}^{(\mathrm{P}_2)}}))$ states that a message flow joins the flow objects of two different pools. The expressions $\bigcup_{\substack{P_1\in S_{\mathrm{P}}\\ POOL_2\in S_{\mathrm{POOL}}}}((F^{\overline{\mathrm{Msg}}}_{\mathrm{EE}^{(\mathrm{P}_1)}} \cup F^{\overline{\mathrm{Msg}}}_{\mathrm{IE}^{(\mathrm{P}_1)}} \cup F_{\mathrm{A}^{(\mathrm{P}_1)}}) \times \{PName^{(POOL_2)}\})$ and $\bigcup_{\substack{P_2\in S_{\mathrm{P}}\\ POOL_1\in S_{\mathrm{POOL}}}}(\{PName^{(POOL_1)}\} \times (F^{\mathrm{Msg}}_{\mathrm{SE}^{(\mathrm{P}_2)}} \cup F^{\mathrm{Msg}}_{\mathrm{IE}^{(\mathrm{P}_2)}} \cup F_{\mathrm{A}^{(\mathrm{P}_2)}}))$ stipulate that a message flow links up a pool and a flow object of another pool. The expression $\bigcup_{POOL_1,POOL_2\in S_{\mathrm{POOL}}}(\{PName^{(POOL_1)}\} \times \{PName^{(POOL_2)}\})$ says that a message flow connects two different pools.

5. Equivalences of BPMN Models

A collection of equivalences for BPMN processes is discussed in considerable detail in our previous work [3]. This section aims to further develop the ideas by (i) introducing another set of equivalences; and (ii) defining precisely when two business process diagrams are behavioural equivalent.

To capture the concept of equivalences of BPMN models, we begin with a definition for inner-outer-DXG form. The inner-outer-DXG form is motivated by the need to restructure a BPMN process. The main idea is that the structure of a BPMN process can be expressed as an alternative representation by swapping (i) an inner data-based exclusive decision gateway with an outer data-based exclusive decision gateway and (ii) an inner data-based exclusive merge gateway with an outer data-based exclusive merge gateway. Based on this definition, we then present the IO-DXG-equivalence and its properties.

Definition 37 (Inner-outer-DXG Form) *Let* P_1 *be a process where* Ω_{C}, C_{SF}, S_{Cond} *are replaced by* $\Omega_{\mathrm{C}^{(\mathrm{P}_1)}}$, $C_{\mathrm{SF}^{(\mathrm{P}_1)}}$, $S_{\mathrm{Cond}^{(\mathrm{P}_1)}}$. *If* FO_1, $FO_2 \in S_{\mathrm{F}}$, A_i, $A_j \in F_{\mathrm{A}}$, G_1, $G_2 \in F^{\mathrm{D}}_{\mathrm{XDG}}$, G_3, $G_4 \in F^{\mathrm{D}}_{\mathrm{XMG}}$, c_1, c_2, …, c_m, c_{m+1}, c_{m+2}, …, c_n, $c_{n+1} \in S_{\mathrm{Cond}^{(\mathrm{P}_1)}}$, (FO_1, G_2), (G_2, A_1), (G_2, A_2), …, (G_2, A_m), (G_2, G_1), (G_1, A_{m+1}), (G_1, A_{m+2}),

$\ldots$, $(G_1,\ A_n)$, $(A_1,\ G_4)$, $(A_2,\ G_4)$, $\ldots$, $(A_m,\ G_4)$, $(A_{m+1},\ G_3)$, $(A_{m+2},\ G_3)$, $\ldots$, (A_n, G_3), (G_3, G_4), $(G_4, FO_2) \in C_{\mathrm{SF}^{(\mathrm{P}_1)}}$, $\Phi_{\mathrm{Cond}}((G_2, A_i)) = c_i$, $\Phi_{\mathrm{Cond}}((G_1, A_j)) = c_j$, $\Phi_{\mathrm{Cond}}((G_2, G_1)) = c_{n+1}$, $CC_{\mathrm{SF}^{(\mathrm{P}_1)}} = \{(FO_1,\ G_2)$, $(G_2,\ A_1)$, $(G_2,\ A_2)$, $\ldots$, $(G_2,\ A_m)$, $(G_2,\ G_1)$, $(G_1,\ A_{m+1})$, $(G_1,\ A_{m+2})$, $\ldots$, $(G_1,\ A_n)$, $(A_1,\ G_4)$, $(A_2,\ G_4)$, $\ldots$, $(A_m,\ G_4)$, $(A_{m+1},\ G_3)$, $(A_{m+2},\ G_3)$, $\ldots$, $(A_n,\ G_3)$, $(G_3,\ G_4)$, $(G_4,\ FO_2)\}$, $SS_{\mathrm{Cond}^{(\mathrm{P}_1)}} = \{c_{m+1},\ c_{m+2},\ \ldots,\ c_n,\ c_{n+1}\}$ *for* $i = 1, \ldots, m$ *and* $j = m+1, \ldots, n$, *then there is a unique process* P_2 *which is in inner-outer-DXG form such that*

- Ω_{C}, C_{SF}, S_{Cond} *are replaced by* $\Omega_{\mathrm{C}^{(\mathrm{P}_2)}}$, $C_{\mathrm{SF}^{(\mathrm{P}_2)}}$, $S_{\mathrm{Cond}^{(\mathrm{P}_2)}}$,
- FO_1, $FO_2 \in S_{\mathrm{F}}$,
- A_i, $A_j \in F_{\mathrm{A}}$,
- G_1, $G_2 \in F^{\mathrm{D}}_{\mathrm{XDG}}$, G_3, $G_4 \in F^{\mathrm{D}}_{\mathrm{XMG}}$,
- c_1, c_2, $\ldots$, c_m, $c_{n+1} \wedge c_{m+1}$, $c_{n+1} \wedge c_{m+2}$, $\ldots$, $c_{n+1} \wedge c_n$, $c_1 \vee c_2 \vee \ldots \vee c_m \in S_{\mathrm{Cond}^{(\mathrm{P}_2)}}$,
- $(FO_1,\ G_1)$, $(G_1,\ G_2)$, $(G_2,\ A_1)$, $(G_2,\ A_2)$, $\ldots$, $(G_2,\ A_m)$, $(G_1,\ A_{m+1})$, $(G_1,\ A_{m+2})$, $\ldots$, $(G_1,\ A_n)$, $(A_1,\ G_4)$, $(A_2,\ G_4)$, $\ldots$, $(A_m,\ G_4)$, $(G_4,\ G_3)$, $(A_{m+1},\ G_3)$, $(A_{m+2},\ G_3)$, $\ldots$, $(A_n,\ G_3)$, $(G_3,\ FO_2) \in C_{\mathrm{SF}^{(\mathrm{P}_2)}}$,
- $\Phi_{\mathrm{Cond}}((G_2, A_i)) = c_i$,
- $\Phi_{\mathrm{Cond}}((G_1, A_j)) = c_{n+1} \wedge c_j$,
- $\Phi_{\mathrm{Cond}}((G_1, G_2)) = c_1 \vee c_2 \vee \ldots \vee c_m$,
- $CC_{\mathrm{SF}^{(\mathrm{P}_2)}} = \{(FO_1,\ G_1)$, $(G_1,\ G_2)$, $(G_2,\ A_1)$, $(G_2,\ A_2)$, $\ldots$, $(G_2,\ A_m)$, $(G_1,\ A_{m+1})$, $(G_1,\ A_{m+2})$, $\ldots$, $(G_1,\ A_n)$, $(A_1,\ G_4)$, $(A_2,\ G_4)$, $\ldots$, $(A_m,\ G_4)$, $(G_4,\ G_3)$, $(A_{m+1},\ G_3)$, $(A_{m+2},\ G_3)$, $\ldots$, $(A_n,\ G_3)$, $(G_3,\ FO_2)\}$,
- $SS_{\mathrm{Cond}^{(\mathrm{P}_2)}} = \{c_{n+1} \wedge c_{m+1},\ c_{n+1} \wedge c_{m+2},\ \ldots,\ c_{n+1} \wedge c_n,\ c_1 \vee c_2 \vee \ldots \vee c_m\}$,
- $C_{\mathrm{SF}^{(\mathrm{P}_2)}} = C_{\mathrm{SF}^{(\mathrm{P}_1)}} \setminus CC_{\mathrm{SF}^{(\mathrm{P}_1)}} \cup CC_{\mathrm{SF}^{(\mathrm{P}_2)}}$ *and*
- $S_{\mathrm{Cond}^{(\mathrm{P}_2)}} = S_{\mathrm{Cond}^{(\mathrm{P}_1)}} \setminus SS_{\mathrm{Cond}^{(\mathrm{P}_1)}} \cup SS_{\mathrm{Cond}^{(\mathrm{P}_2)}}$

for $i = 1, \ldots, m$ *and* $j = m+1, \ldots, n$.

The rationale behind this definition is to capture the concept that an equivalent representation is obtained by (i) swapping a pair of inner data-based exclusive decision gateway and data-based exclusive merge gateway with a pair of outer data-based exclusive decision gateway and data-based exclusive merge gateway; (ii) modifying the associated conditional expressions from c_j to $c_{n+1} \wedge c_j$; (iii) adding a new conditional expression $c_1 \vee c_2 \vee \ldots \vee c_m$; and (iv) deleting the conditional expression c_{n+1}. Swapping the gateways repeatedly results in a unique process P_2 where no further transformation can be applied.

The intuitive meaning of the equivalence of BPMN processes is that two BPMN processes are considered as equivalent if and only if there is a third BPMN process that is the normal form of these two processes. Typically, an inner-outer-DXG form is a normal form. Based on these concepts, the equivalence of BPMN processes is formalized in terms of inner-outer-DXG form as the following definition.

Definition 38 (IO-DXG-equivalence) *For any BPMN processes P_1 and P_2, P_1 is IO-DXG-equivalent to P_2, denoted by $P_1 \approx^{\mathrm{IO}}_{\mathrm{DXG}} P_2$, if and only if there is a BPMN process P_3 such that P_3 is an inner-outer-DXG form of P_1 and P_2.*

Definition 38 states that two BPMN processes are IO-DXG-equivalent if they can be convertible into a BPMN process which is in inner-outer-DXG form.

Proposition 1 *The relation $\approx^{\mathrm{IO}}_{\mathrm{DXG}}$ is transitive.*
Proof. *Suppose $P_1 \approx^{\mathrm{IO}}_{\mathrm{DXG}} P_2$ and $P_2 \approx^{\mathrm{IO}}_{\mathrm{DXG}} P_3$. Since $P_1 \approx^{\mathrm{IO}}_{\mathrm{DXG}} P_2$, $P_2 \approx^{\mathrm{IO}}_{\mathrm{DXG}} P_3$ and the normal form is a unique process, it follows from Definition 38 that P_4 is a inner-outer-DXG form of P_1, P_2 and P_3. Since P_4 is a inner-outer-DXG form of P_1 and P_3, we obtain $P_1 \approx^{\mathrm{IO}}_{\mathrm{DXG}} P_3$. Thus, $\approx^{\mathrm{IO}}_{\mathrm{DXG}}$ is transitive.* □

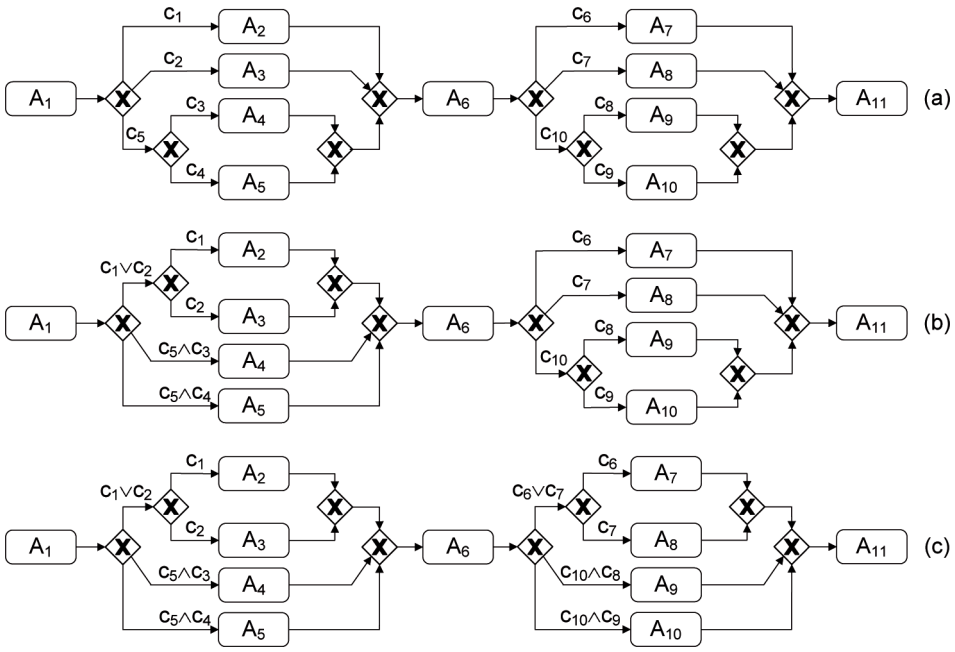

Fig. 5. IO-DXG-equivalent BPMN processes

Proposition 2 *The relation* $\approx^{\mathrm{IO}}_{\mathrm{DXG}}$ *is an equivalence.*
Proof. *For reflexivity and symmetry, these follow immediately from Definition 38. Thus,* $\approx^{\mathrm{IO}}_{\mathrm{DXG}}$ *is an equivalence as* $\approx^{\mathrm{IO}}_{\mathrm{DXG}}$ *is transitive by Proposition 1.* □

Consider Figure 5 (a), a pair of inner data-based exclusive decision gateway and data-based exclusive merge gateway is swapped with a pair of outer data-based exclusive decision gateway and data-based exclusive merge gateway. The conditional expressions c_3 and c_4 are changed to $c_5 \wedge c_3$ and $c_5 \wedge c_4$. A new conditional expression $c_1 \vee c_2$ is added and the conditional expression c_5 is removed as shown in Figure 5 (b). The two BPMN processes (Figures 5 (a) and 5 (b)) are IO-DXG-equivalent since the BPMN process in Figure 5 (c) is a normal form of them.

A BPMN process can be simplified through the elimination of a pair of inner data-based exclusive gateways. A formal definition is given below.

Definition 39 (Eliminated-inner-DXG Form) *Let* P_1 *be a process where* Ω_{G}, Ω_{C}, Ω_{PG}, F_{PFG}, F_{PJG}, C_{SF}, S_{Cond} *are replaced by* $\Omega_{\mathrm{G}^{(\mathrm{P}_1)}}$, $\Omega_{\mathrm{C}^{(\mathrm{P}_1)}}$, $\Omega_{\mathrm{PG}^{(\mathrm{P}_1)}}$, $F_{\mathrm{PFG}^{(\mathrm{P}_1)}}$, $F_{\mathrm{PJG}^{(\mathrm{P}_1)}}$, $C_{\mathrm{SF}^{(\mathrm{P}_1)}}$, $S_{\mathrm{Cond}^{(\mathrm{P}_1)}}$. *If* FO_1, $FO_2 \in S_{\mathrm{F}}$, A_i, $A_j \in \mathrm{F}_{\mathrm{A}}$, G_1, $G_2 \in F^{\mathrm{D}}_{\mathrm{XDG}}$, G_3, $G_4 \in F^{\mathrm{D}}_{\mathrm{XMG}}$, c_1, c_2, …, c_m, c_{m+1}, c_{m+2}, …, c_n, $c_{n+1} \in S_{\mathrm{Cond}^{(\mathrm{P}_1)}}$, (FO_1, G_2), (G_2, A_1), (G_2, A_2), …, (G_2, A_m), (G_2, G_1), (G_1, A_{m+1}), (G_1, A_{m+2}), …, (G_1, A_n), (A_1, G_4), (A_2, G_4), …, (A_m, G_4), (A_{m+1}, G_3), (A_{m+2}, G_3), …, (A_n, G_3), (G_3, G_4), $(G_4, FO_2) \in C_{\mathrm{SF}^{(\mathrm{P}_1)}}$, $\Phi_{\mathrm{Cond}}((G_2, A_i)) = c_i$, $\Phi_{\mathrm{Cond}}((G_1, A_j)) = c_j$, $\Phi_{\mathrm{Cond}}((G_2, G_1)) = c_{n+1}$, $CC_{\mathrm{SF}^{(\mathrm{P}_1)}} = \{(G_2, G_1), (G_1, A_{m+1}), (G_1, A_{m+2}), \ldots, (G_1, A_n), (A_{m+1}, G_3), (A_{m+2}, G_3), \ldots, (A_n, G_3), (G_3, G_4)\}$, $SS_{\mathrm{Cond}^{(\mathrm{P}_1)}} = \{c_{m+1}, c_{m+2}, \ldots, c_n, c_{n+1}\}$ *for* $i = 1, \ldots, m$ *and* $j = m+1, \ldots, n$, *then there is a unique process* P_2 *which is in eliminated-inner-DXG form such that*

- Ω_{G}, Ω_{C}, Ω_{PG}, F_{PFG}, F_{PJG} C_{SF}, S_{Cond} *are replaced by* $\Omega_{\mathrm{G}^{(\mathrm{P}_2)}}$, $\Omega_{\mathrm{C}^{(\mathrm{P}_2)}}$, $\Omega_{\mathrm{PG}^{(\mathrm{P}_2)}}$, $F_{\mathrm{PFG}^{(\mathrm{P}_2)}}$, $F_{\mathrm{PJG}^{(\mathrm{P}_2)}}$, $C_{\mathrm{SF}^{(\mathrm{P}_2)}}$, $S_{\mathrm{Cond}^{(\mathrm{P}_2)}}$,
- FO_1, $FO_2 \in S_{\mathrm{F}}$,
- A_i, $A_j \in F_{\mathrm{A}}$,
- $G_2 \in F^{\mathrm{D}}_{\mathrm{XDG}}$, $G_4 \in F^{\mathrm{D}}_{\mathrm{XMG}}$,
- c_1, c_2, …, c_m, $c_{n+1} \wedge c_{m+1}$, $c_{n+1} \wedge c_{m+2}$, …, $c_{n+1} \wedge c_n \in S_{\mathrm{Cond}^{(\mathrm{P}_2)}}$,
- (FO_1, G_2), (G_2, A_1), (G_2, A_2), …, (G_2, A_m), (G_2, A_{m+1}), (G_2, A_{m+2}), …, (G_2, A_n), (A_1, G_4), (A_2, G_4), …, (A_m, G_4), (A_{m+1}, G_4), (A_{m+2}, G_4), …, (A_n, G_4), $(G_4, FO_2) \in C_{\mathrm{SF}^{(\mathrm{P}_2)}}$,
- $\Phi_{\mathrm{Cond}}((G_2, A_i)) = c_i$,
- $\Phi_{\mathrm{Cond}}((G_2, A_j)) = c_{n+1} \wedge c_j$,
- $CC_{\mathrm{SF}^{(\mathrm{P}_2)}} = \{(G_2, A_{m+1}), (G_2, A_{m+2}), \ldots, (G_2, A_n), (A_{m+1}, G_4), (A_{m+2}, G_4), \ldots, (A_n, G_4)\}$,

— $SS_{\mathrm{Cond}^{(\mathrm{P}_2)}} = \{c_{n+1} \wedge c_{m+1},\, c_{n+1} \wedge c_{m+2},\, \ldots,\, c_{n+1} \wedge c_n\}$,
— $F_{\mathrm{PFG}^{(\mathrm{P}_2)}} = F_{\mathrm{PFG}^{(\mathrm{P}_1)}} \setminus \{G_1\}$,
— $F_{\mathrm{PJG}^{(\mathrm{P}_2)}} = F_{\mathrm{PJG}^{(\mathrm{P}_1)}} \setminus \{G_3\}$,
— $C_{\mathrm{SF}^{(\mathrm{P}_2)}} = C_{\mathrm{SF}^{(\mathrm{P}_1)}} \setminus CC_{\mathrm{SF}^{(\mathrm{P}_1)}} \cup CC_{\mathrm{SF}^{(\mathrm{P}_2)}}$ *and*
— $S_{\mathrm{Cond}^{(\mathrm{P}_2)}} = S_{\mathrm{Cond}^{(\mathrm{P}_1)}} \setminus SS_{\mathrm{Cond}^{(\mathrm{P}_1)}} \cup SS_{\mathrm{Cond}^{(\mathrm{P}_2)}}$

for $i = 1, \ldots, m$ *and* $j = m + 1, \ldots, n$.

Besides the swapping of a pair of inner data-based exclusive gateways with a pair of outer data-based exclusive gateways as specified in Definition 37, an alternative way to get an equivalent representation is to remove the pair of inner data-based exclusive gateways and amend the conditional expressions from c_j to $c_{n+1} \wedge c_j$ as stipulated by Definition 39.

Definition 40 (EI-DXG-equivalence) *For any BPMN processes* P_1 *and* P_2, P_1 *is EI-DXG-equivalent to* P_2, *denoted by* $P_1 \approx^{\mathrm{EI}}_{\mathrm{DXG}} P_2$, *if and only if there is a BPMN process* P_3 *such that* P_3 *is an eliminated-inner-DXG form of* P_1 *and* P_2.

The existence of ways for transforming two BPMN processes into a BPMN process in eliminated-inner-DXG form implies that they are EI-DXG-equivalent (Definition 40).

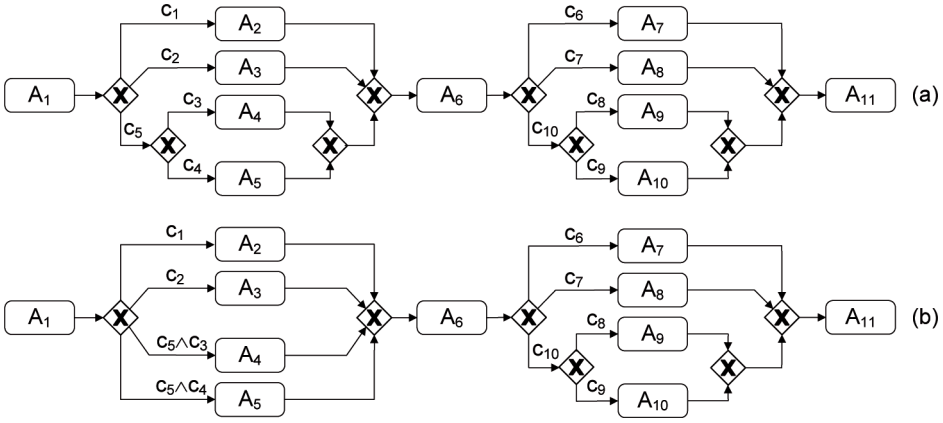

Fig. 6. EI-DXG-equivalent BPMN processes

Proposition 3 *The relation* $\approx^{\mathrm{EI}}_{\mathrm{DXG}}$ *is transitive.*
By similar argument as Proposition 1. □

Proposition 4 *The relation* $\approx^{\mathrm{EI}}_{\mathrm{DXG}}$ *is an equivalence.*
Proof. *Analogous to Proposition 2.* □

As depicted in Figure 6 (b), the inner pair of data-based exclusive gateways is eliminated to produce a semantically equivalent diagram that is less cluttered. The conditions c_3 and c_4 (Figure 6 (a)) are replaced by the conditions $c_5 \wedge c_3$ and $c_5 \wedge c_4$ (Figure 6 (b)).

To capture the fact that the simplification of a BPMN process can be achieved by removing a parallel fork gateway which connects a none start event to a collection of flow objects, the following definition is introduced.

Definition 41 (Start-event-implicit-PFG Form) *Let P_1 be a process where Ω_{G}, Ω_{C}, Ω_{PG}, F_{PFG}, C_{SF} are replaced by $\Omega_{\mathrm{G}^{(\mathrm{P}_1)}}$, $\Omega_{\mathrm{C}^{(\mathrm{P}_1)}}$, $\Omega_{\mathrm{PG}^{(\mathrm{P}_1)}}$, $F_{\mathrm{PFG}^{(\mathrm{P}_1)}}$, $C_{\mathrm{SF}^{(\mathrm{P}_1)}}$, $F_{\mathrm{PG}^{(\mathrm{P}_1)}} = F_{\mathrm{PFG}^{(\mathrm{P}_1)}} \cup F_{\mathrm{PJG}}$, $F_{\mathrm{G}^{(\mathrm{P}_1)}} = \bigcup_{i \in \{\mathrm{XG,IG,CG}\}} F_i \cup F_{\mathrm{PG}^{(\mathrm{P}_1)}}$, $S_{\mathrm{F}^{(\mathrm{P}_1)}} = F_{\mathrm{G}^{(\mathrm{P}_1)}} \cup \bigcup_{i \in \{\mathrm{E,A}\}} F_i$. If $FO_i \in S_{\mathrm{F}^{(\mathrm{P}_1)}}$, $E_1 \in F_{\mathrm{SE}}^{\mathrm{None}}$, $G_1 \in F_{\mathrm{PFG}^{(\mathrm{P}_1)}}$, (EO_1, G_1), (G_1, FO_1), (G_1, FO_2), …, $(G_1, FO_n) \in C_{\mathrm{SF}^{(\mathrm{P}_1)}}$, $CC_{\mathrm{SF}^{(\mathrm{P}_1)}} = \{(EO_1, G_1), (G_1, FO_1), (G_1, FO_2), \ldots, (G_1, FO_n)\}$ for $i = 1, \ldots, n$, then there is a unique process P_2 which is in start-event-implicit-PFG form such that*

- *Ω_{G}, Ω_{C}, Ω_{PG}, F_{PFG}, C_{SF} are replaced by $\Omega_{\mathrm{G}^{(\mathrm{P}_2)}}$, $\Omega_{\mathrm{C}^{(\mathrm{P}_2)}}$, $\Omega_{\mathrm{PG}^{(\mathrm{P}_2)}}$, $F_{\mathrm{PFG}^{(\mathrm{P}_2)}}$, $C_{\mathrm{SF}^{(\mathrm{P}_2)}}$,*
- *$F_{\mathrm{PG}^{(\mathrm{P}_2)}} = F_{\mathrm{PFG}^{(\mathrm{P}_2)}} \cup F_{\mathrm{PJG}}$,*
- *$F_{\mathrm{G}^{(\mathrm{P}_2)}} = \bigcup_{i \in \{\mathrm{XG,IG,CG}\}} F_i \cup F_{\mathrm{PG}^{(\mathrm{P}_2)}}$,*
- *$S_{\mathrm{F}^{(\mathrm{P}_2)}} = F_{\mathrm{G}^{(\mathrm{P}_2)}} \cup \bigcup_{i \in \{\mathrm{E,A}\}} F_i$,*
- *$FO_i \in S_{\mathrm{F}^{(\mathrm{P}_2)}}$,*
- *$E_1 \in F_{\mathrm{SE}}^{\mathrm{None}}$,*
- *$(E_1, FO_1), (E_1, FO_2), \ldots, (E_1, FO_n) \in C_{\mathrm{SF}^{(\mathrm{P}_2)}}$,*
- *$CC_{\mathrm{SF}^{(\mathrm{P}_2)}} = \{(E_1, FO_1), (E_1, FO_2), \ldots, (E_1, FO_n)\}$*
- *$F_{\mathrm{PFG}^{(\mathrm{P}_2)}} = C_{\mathrm{PFG}^{(\mathrm{P}_1)}} \setminus \{G_1\}$ and*
- *$C_{\mathrm{SF}^{(\mathrm{P}_2)}} = C_{\mathrm{SF}^{(\mathrm{P}_1)}} \setminus CC_{\mathrm{SF}^{(\mathrm{P}_1)}} \cup CC_{\mathrm{SF}^{(\mathrm{P}_2)}}$*

for $i = 1, \ldots, n$.

Through the elimination of a parallel fork gateway which its incoming and outgoing sequence flows connect, respectively, to a none start event and a set of flow objects, an equivalent representation is yielded by linking up the none start event with the set of flow objects directly (Definition 41).

Definition 42 (SEImpl-PFG-equivalence) *For any BPMN processes P_1 and P_2, P_1 is SEImpl-PFG-equivalent to P_2, denoted by $P_1 \approx_{\mathrm{PFG}}^{\mathrm{SEImpl}} P_2$, if and only if there is a BPMN process P_3 such that P_3 is a start-event-implicit-PFG form of P_1 and P_2.*

Proposition 5 *The relation $\approx_{\mathrm{PFG}}^{\mathrm{SEImpl}}$ is transitive.*
By similar argument as Proposition 1. □

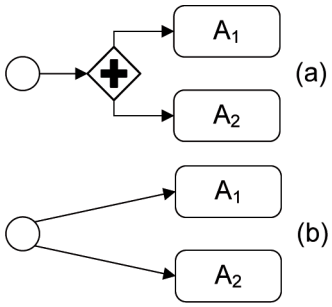

Fig. 7. SEImpl-PFG-equivalent BPMN processes

Proposition 6 *The relation* $\approx_{\mathrm{PFG}}^{\mathrm{SEImpl}}$ *is an equivalence.*
Proof. *Analogous to Proposition 2.* □

Definition 42 specifies SEImpl-PFG-equivalence in terms of start-event-implicit-PFG form. An example of SEImpl-PFG-equivalence is delineated in Figures 7 (a) and (b).

Analogous method for the removal of a data-based exclusive merge gateway which connects a collection of flow objects to a flow object is formally specified as provided below.

Definition 43 (Implicit-DXMG Form) *Let* P_1 *be a process where* Ω_{G}, Ω_{C}, Ω_{XG}, $F^{\mathrm{D}}_{\mathrm{XMG}}$, C_{SF} *are replaced by* $\Omega_{\mathrm{G}^{(\mathrm{P}_1)}}$, $\Omega_{\mathrm{C}^{(\mathrm{P}_1)}}$, $\Omega_{\mathrm{XG}^{(\mathrm{P}_1)}}$, $F^{\mathrm{D}}_{\mathrm{XMG}^{(\mathrm{P}_1)}}$, $C_{\mathrm{SF}^{(\mathrm{P}_1)}}$, $F_{\mathrm{XG}^{(\mathrm{P}_1)}} = \bigcup_{i\in\{\mathrm{D,E}\}} F^{i}_{\mathrm{XDG}} \cup (F^{\mathrm{D}}_{\mathrm{XMG}^{(\mathrm{P}_1)}} \cup F^{\mathrm{E}}_{\mathrm{XMG}})$, $F_{\mathrm{G}^{(\mathrm{P}_1)}} = \bigcup_{i\in\{\mathrm{IG,CG,PG}\}} F_i \cup F_{\mathrm{XG}^{(\mathrm{P}_1)}}$ *and* $S_{\mathrm{F}^{(\mathrm{P}_1)}} = F_{\mathrm{G}^{(\mathrm{P}_1)}} \cup \bigcup_{i\in\{\mathrm{E,A}\}} F_i$. *If* $FO_i \in S_{\mathrm{F}^{(\mathrm{P}_1)}}$, $A_1 \in F_{\mathrm{A}}$, $G_1 \in F^{\mathrm{D}}_{\mathrm{XMG}^{(\mathrm{P}_1)}}$, (FO_1, G_1), (FO_2, G_1), …, (FO_{n-1}, G_1), (G_1, A_1), $(A_1, FO_n) \in C_{\mathrm{SF}^{(\mathrm{P}_1)}}$, $CC_{\mathrm{SF}^{(\mathrm{P}_1)}} = \{(FO_1, G_1), (FO_2, G_1), \ldots, (FO_{n-1}, G_1), (G_1, A_1)\}$ *for* $i = 1, \ldots, n$, *then there is a unique process* P_2 *which is in implicit-DXMG form such that*

- Ω_{G}, Ω_{C}, Ω_{XG}, $F^{\mathrm{D}}_{\mathrm{XMG}}$, C_{SF} *are replaced by* $\Omega_{\mathrm{G}^{(\mathrm{P}_2)}}$, $\Omega_{\mathrm{C}^{(\mathrm{P}_2)}}$, $\Omega_{\mathrm{XG}^{(\mathrm{P}_2)}}$, $F^{\mathrm{D}}_{\mathrm{XMG}^{(\mathrm{P}_2)}}$, $C_{\mathrm{SF}^{(\mathrm{P}_2)}}$,
- $F_{\mathrm{XG}^{(\mathrm{P}_2)}} = \bigcup_{i\in\{\mathrm{D,E}\}} F^{i}_{\mathrm{XDG}} \cup (F^{\mathrm{D}}_{\mathrm{XMG}^{(\mathrm{P}_2)}} \cup F^{\mathrm{E}}_{\mathrm{XMG}})$,
- $F_{\mathrm{G}^{(\mathrm{P}_2)}} = \bigcup_{i\in\{\mathrm{IG,CG,PG}\}} F_i \cup F_{\mathrm{XG}^{(\mathrm{P}_2)}}$,
- $S_{\mathrm{F}^{(\mathrm{P}_2)}} = F_{\mathrm{G}^{(\mathrm{P}_2)}} \cup \bigcup_{i\in\{\mathrm{E,A}\}} F_i$,
- $FO_i \in S_{\mathrm{F}^{(\mathrm{P}_2)}}$,
- $A_1 \in F_{\mathrm{A}}$,
- $(FO_1, A_1), (FO_2, A_1), \ldots, (FO_{n-1}, A_1), (A_1, FO_n) \in C_{\mathrm{SF}^{(\mathrm{P}_2)}}$,
- $CC_{\mathrm{SF}^{(\mathrm{P}_2)}} = \{(FO_1, A_1), (FO_2, A_1), \ldots, (FO_{n-1}, A_1)\}$,

— $F^{\mathrm{D}}_{\mathrm{XMG}^{(\mathrm{P}_2)}} = F^{\mathrm{D}}_{\mathrm{XMG}^{(\mathrm{P}_1)}} \setminus \{G_1\}$ *and*
— $C_{\mathrm{SF}^{(\mathrm{P}_2)}} = C_{\mathrm{SF}^{(\mathrm{P}_1)}} \setminus CC_{\mathrm{SF}^{(\mathrm{P}_1)}} \cup CC_{\mathrm{SF}^{(\mathrm{P}_2)}}$

for $i = 1, \ldots, n$.

A BPMN process consisting of a data-based exclusive merge gateway with $n-1$ incoming sequence flows and an outgoing sequence flow connecting to an activity can be simplified by removing the data-based exclusive merge gateway such that the $n-1$ incoming sequence flows link up directly with the activity (Definition 43).

Definition 44 (Impl-DXMG-equivalence) *For any BPMN processes P_1 and P_2, P_1 is Impl-DXMG-equivalent to P_2, denoted by $P_1 \approx^{\mathrm{Impl}}_{\mathrm{DXMG}} P_2$, if and only if there is a BPMN process P_3 such that P_3 is an implicit-DXMG form of P_1 and P_2.*

Proposition 7 *The relation $\approx^{\mathrm{Impl}}_{\mathrm{DXMG}}$ is transitive.*
Proof. *By similar argument as Proposition 1.* □

Proposition 8 *The relation $\approx^{\mathrm{Impl}}_{\mathrm{DXMG}}$ is an equivalence.*
Proof. *Analogous to Proposition 2.* □

Definition 44 sets out the interchangeability of a BPMN process comprising an activity and a data-based exclusive merge gateway with multiple incoming sequence flows as well as an outgoing sequence flow and a BPMN process consisting of the activity with the multiple incoming sequence flows. Figures 8 (a) and (b) illustrate the concept of Impl-DXMG-equivalence.

In addition to Definitions 39, 41 and 43, another technique for the simplification of a BPMN process is defined below. Conceptually, a none start event, which connects to a data-based exclusive decision gateway, is removable.

Definition 45 (DXDG-implicit-start-event Form) *Let P_1 be a process where Ω_{E}, Ω_{C}, Ω_{SE}, $F^{\mathrm{None}}_{\mathrm{SE}}$, C_{SF} are replaced by $\Omega_{\mathrm{E}^{(\mathrm{P}_1)}}$, $\Omega_{\mathrm{C}^{(\mathrm{P}_1)}}$, $\Omega_{\mathrm{SE}^{(\mathrm{P}_1)}}$, $F^{\mathrm{None}}_{\mathrm{SE}^{(\mathrm{P}_1)}}$, $C_{\mathrm{SF}^{(\mathrm{P}_1)}}$, $F_{\mathrm{SE}^{(\mathrm{P}_1)}} = F^{\mathrm{None}}_{\mathrm{SE}^{(\mathrm{P}_1)}} \cup \bigcup_{i \in \Gamma_{\mathrm{SE}} \setminus \{\mathrm{None}\}} F^{i}_{\mathrm{SE}}$, $F_{\mathrm{E}^{(\mathrm{P}_1)}} = F_{\mathrm{SE}^{(\mathrm{P}_1)}} \cup \bigcup_{i \in \{\mathrm{EE},\mathrm{IE}\}} F_i$ and $S_{\mathrm{F}^{(\mathrm{P}_1)}} = F_{\mathrm{E}^{(\mathrm{P}_1)}} \cup \bigcup_{i \in \{\mathrm{G},\mathrm{C}\}} F_i$. If $FO_i \in S_{\mathrm{F}^{(\mathrm{P}_1)}}$, $E_1 \in F^{\mathrm{None}}_{\mathrm{SE}^{(\mathrm{P}_1)}}$, $G_1 \in F^{\mathrm{D}}_{\mathrm{XDG}}$, (E_1, G_1), (G_1, FO_1), (G_1, FO_2), …, $(G_1, FO_n) \in C_{\mathrm{SF}^{(\mathrm{P}_1)}}$, $\Phi_{\mathrm{Cond}}((G_1, FO_i)) = c_i$ for $i = 1, \ldots, n$, then there is a unique process P_2 which is in DXDG-implicit-start-event form such that*

— Ω_{E}, Ω_{C}, Ω_{SE}, $F^{\mathrm{None}}_{\mathrm{SE}}$, C_{SF} *are replaced by* $\Omega_{\mathrm{E}^{(\mathrm{P}_2)}}$, $\Omega_{\mathrm{C}^{(\mathrm{P}_2)}}$, $\Omega_{\mathrm{SE}^{(\mathrm{P}_2)}}$, $F^{\mathrm{None}}_{\mathrm{SE}^{(\mathrm{P}_2)}}$, $C_{\mathrm{SF}^{(\mathrm{P}_2)}}$,
— $F_{\mathrm{SE}^{(\mathrm{P}_2)}} = F^{\mathrm{None}}_{\mathrm{SE}^{(\mathrm{P}_2)}} \cup \bigcup_{i \in \Gamma_{\mathrm{SE}} \setminus \{\mathrm{None}\}} F^{i}_{\mathrm{SE}}$,
— $F_{\mathrm{E}^{(\mathrm{P}_2)}} = F_{\mathrm{SE}^{(\mathrm{P}_2)}} \cup \bigcup_{i \in \{\mathrm{EE},\mathrm{IE}\}} F_i$,
— $S_{\mathrm{F}^{(\mathrm{P}_2)}} = F_{\mathrm{E}^{(\mathrm{P}_2)}} \cup \bigcup_{i \in \{\mathrm{G},\mathrm{C}\}} F_i$,

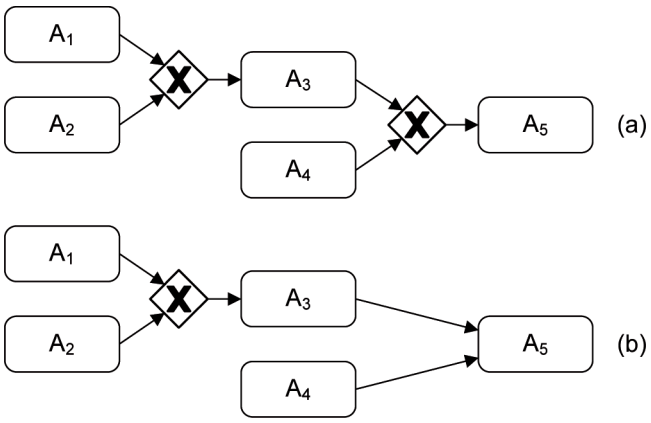

Fig. 8. Impl-DXMG-equivalent BPMN processes

— $FO_i \in S_{\mathrm{F}^{(\mathrm{P}_2)}}$,
— $G_1 \in F^{\mathrm{D}}_{\mathrm{XDG}}$,
— $(G_1, FO_1), (G_1, FO_2), \ldots, (G_1, FO_n) \in C_{\mathrm{SF}^{(\mathrm{P}_2)}}$,
— $\Phi_{\mathrm{Cond}}((G_1, FO_i)) = c_i$,
— $F^{\mathrm{None}}_{\mathrm{SE}^{(\mathrm{P}_2)}} = F^{\mathrm{None}}_{\mathrm{SE}^{(\mathrm{P}_1)}} \setminus \{E_1\}$ *and*
— $C_{\mathrm{SF}^{(\mathrm{P}_2)}} = C_{\mathrm{SF}^{(\mathrm{P}_1)}} \setminus \{(E_1, G_1)\}$

for $i = 1, \ldots, n$.

Definition 45 says that a data-based exclusive decision gateway without an incoming sequence flow is an alternative representation of a none start event connecting to the data-based exclusive decision gateway.

Definition 46 (DXDG-ImplSE-equivalence) *For any BPMN processes P_1 and P_2, P_1 is DXDG-ImplSE-equivalent to P_2, denoted by $P_1 \approx^{\mathrm{DXDG}}_{\mathrm{ImplSE}} P_2$, if and only if there is a BPMN process P_3 such that P_3 is a DXDG-implicit-start-event form of P_1 and P_2.*

Proposition 9 *The relation $\approx^{\mathrm{DXDG}}_{\mathrm{ImplSE}}$ is transitive.*
Proof. *By similar argument as Proposition 1.* □

Proposition 10 *The relation $\approx^{\mathrm{DXDG}}_{\mathrm{ImplSE}}$ is an equivalence.*
Proof. *Analogous to Proposition 2.* □

As stipulated by Definition 46, two BPMN processes are DXDG-ImplSE-equivalent provided that there exists a sequence of transformations for generating a BPMN process

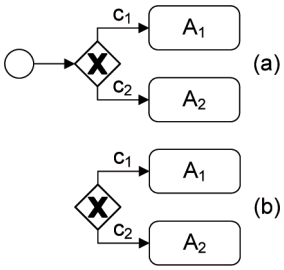

Fig. 9. DXDG-ImplSE-equivalent BPMN processes

in DXDG-implicit-start-event form. Figures 9 (a) and (b) are an example of two BPMN processes which are DXDG-ImplSE-equivalent.

Before the behavioural equivalence of business process diagrams is discussed, a formal definition of structural equivalence is presented.

Definition 47 (Structural Equivalence) *Let* $P_i = (\Omega_{\mathrm{E}^{(\mathrm{P}_i)}}, \Omega_{\mathrm{A}^{(\mathrm{P}_i)}}, \Omega_{\mathrm{G}^{(\mathrm{P}_i)}}, \Omega_{\mathrm{C}^{(\mathrm{P}_i)}})$, $\Omega_{\mathrm{E}^{(\mathrm{P}_i)}} = (\Omega_{\mathrm{SE}^{(\mathrm{P}_i)}}, \Omega_{\mathrm{IE}^{(\mathrm{P}_i)}}, \Omega_{\mathrm{EE}^{(\mathrm{P}_i)}}, \Phi^{\mathrm{Att}}_{\mathrm{E}^{(\mathrm{P}_i)}})$, $\Omega_{\mathrm{A}^{(\mathrm{P}_i)}} = (\Omega_{\mathrm{T}^{(\mathrm{P}_i)}}, \Omega_{\mathrm{SP}^{(\mathrm{P}_i)}}, \Phi^{\mathrm{Bdy[-TX]}}_{\mathrm{IE}^{(\mathrm{P}_i)}}$, $\Phi^{\mathrm{Bdy[TX]}}_{\mathrm{IE}^{(\mathrm{P}_i)}}$, $\Phi^{\mathrm{Att}}_{\mathrm{A}^{(\mathrm{P}_i)}})$, $\Omega_{\mathrm{G}^{(\mathrm{P}_i)}} = (\Omega_{\mathrm{XG}^{(\mathrm{P}_i)}}, \Omega_{\mathrm{IG}^{(\mathrm{P}_i)}}, \Phi_{\mathrm{CG}^{(\mathrm{P}_i)}}, \Phi_{\mathrm{PG}^{(\mathrm{P}_i)}}, \Phi^{\mathrm{Att}}_{\mathrm{G}^{(\mathrm{P}_i)}})$, $\Omega_{\mathrm{C}^{(\mathrm{P}_i)}} = (A_{\mathrm{DO}^{(\mathrm{P}_i)}}, C_{\mathrm{SF}^{(\mathrm{P}_i)}}, C_{\mathrm{DA}^{(\mathrm{P}_i)}}, S_{\mathrm{Cond}^{(\mathrm{P}_i)}}, \Phi_{\mathrm{Cond}^{(\mathrm{P}_i)}}, \Phi_{\mathrm{IsDf}^{(\mathrm{P}_i)}}, \Phi^{\mathrm{Att}}_{\mathrm{C}^{(\mathrm{P}_i)}})$, $\Omega_{\mathrm{SE}^{(\mathrm{P}_i)}} = (F^{\mathrm{None}}_{\mathrm{SE}^{(\mathrm{P}_i)}}, F^{\mathrm{Msg}}_{\mathrm{SE}^{(\mathrm{P}_i)}}, F^{\mathrm{Timer}}_{\mathrm{SE}^{(\mathrm{P}_i)}}, F^{\mathrm{Cond}}_{\mathrm{SE}^{(\mathrm{P}_i)}}, F^{\mathrm{Sign}}_{\mathrm{SE}^{(\mathrm{P}_i)}}, F^{\mathrm{Multi}}_{\mathrm{SE}^{(\mathrm{P}_i)}})$, $\Omega_{\mathrm{IE}^{(\mathrm{P}_i)}} = (F^{\mathrm{None}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\mathrm{Msg}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\overline{\mathrm{Msg}}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\mathrm{Timer}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\mathrm{Err}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\mathrm{Cncl}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\mathrm{Cmpen}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\overline{\mathrm{Cmpen}}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\mathrm{Cond}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\mathrm{Link}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\overline{\mathrm{Link}}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\mathrm{Sign}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\overline{\mathrm{Sign}}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\mathrm{Multi}}_{\mathrm{IE}^{(\mathrm{P}_i)}}, F^{\overline{\mathrm{Multi}}}_{\mathrm{IE}^{(\mathrm{P}_i)}})$, $\Omega_{\mathrm{EE}^{(\mathrm{P}_i)}} = (F^{\overline{\mathrm{None}}}_{\mathrm{EE}^{(\mathrm{P}_i)}}, F^{\overline{\mathrm{Msg}}}_{\mathrm{EE}^{(\mathrm{P}_i)}}, F^{\overline{\mathrm{Err}}}_{\mathrm{EE}^{(\mathrm{P}_i)}}, F^{\overline{\mathrm{Cncl}}}_{\mathrm{EE}^{(\mathrm{P}_i)}}, F^{\overline{\mathrm{Cmpen}}}_{\mathrm{EE}^{(\mathrm{P}_i)}}, F^{\overline{\mathrm{Sign}}}_{\mathrm{EE}^{(\mathrm{P}_i)}}, F^{\overline{\mathrm{Term}}}_{\mathrm{EE}^{(\mathrm{P}_i)}}, F^{\overline{\mathrm{Multi}}}_{\mathrm{EE}^{(\mathrm{P}_i)}})$, $\Omega_{\mathrm{T}^{(\mathrm{P}_i)}} = (F_{\mathrm{T}^{(\mathrm{P}_i)}}, \Phi_{\mathrm{TM}^{(\mathrm{P}_i)}}, \Phi_{\mathrm{Ttype}^{(\mathrm{P}_i)}}, \Phi_{\mathrm{TName}^{(\mathrm{P}_i)}})$, $\Omega_{\mathrm{SP}^{(\mathrm{P}_i)}} = (F^{\mathrm{Embed}}_{\mathrm{SP}^{(\mathrm{P}_i)}}, F^{\mathrm{Reuse}}_{\mathrm{SP}^{(\mathrm{P}_i)}}, F^{\mathrm{Ref}}_{\mathrm{SP}^{(\mathrm{P}_i)}}, \Phi_{\mathrm{IsTX}^{(\mathrm{P}_i)}}, \Phi_{\mathrm{SPM}^{(\mathrm{P}_i)}}, \Phi^{\mathrm{Bdy}}_{\mathrm{SE}^{(\mathrm{P}_i)}}, \Phi^{\mathrm{Bdy}}_{\mathrm{EE}^{(\mathrm{P}_i)}}, \Phi_{\mathrm{NP}^{(\mathrm{P}_i)}}, \Phi_{\mathrm{P}^{(\mathrm{P}_i)}}, \Phi_{\mathrm{RP}^{(\mathrm{P}_i)}})$, $\Omega_{\mathrm{XG}^{(\mathrm{P}_i)}} = (F^{\mathrm{D}}_{\mathrm{XDG}^{(\mathrm{P}_i)}}, F^{\mathrm{D}}_{\mathrm{XMG}^{(\mathrm{P}_i)}}, F^{\mathrm{E}}_{\mathrm{XDG}^{(\mathrm{P}_i)}}, F^{\mathrm{E}}_{\mathrm{XMG}^{(\mathrm{P}_i)}})$, $\Omega_{\mathrm{IG}^{(\mathrm{P}_i)}} = (F_{\mathrm{IDG}^{(\mathrm{P}_i)}}, F_{\mathrm{IMG}^{(\mathrm{P}_i)}})$, $\Omega_{\mathrm{CG}^{(\mathrm{P}_i)}} = (F_{\mathrm{CDG}^{(\mathrm{P}_i)}}, F_{\mathrm{CMG}^{(\mathrm{P}_i)}})$, $\Omega_{\mathrm{PG}^{(\mathrm{P}_i)}} = (F_{\mathrm{PFG}^{(\mathrm{P}_i)}}, F_{\mathrm{PJG}^{(\mathrm{P}_i)}})$, $\Gamma_{\mathrm{Cat}} = \{\mathrm{E}, \mathrm{A}, \mathrm{G}, \mathrm{C}\}$, $\Gamma_{\mathrm{BdyTX}} = \{\mathrm{Bdy[-TX]}, \mathrm{Bdy[TX]}\}$ *and* $\Gamma_{\mathrm{Misc}} = \{\mathrm{Cond}, \mathrm{IsDf}, \mathrm{TM}, \mathrm{Ttype}, \mathrm{TName}, \mathrm{IsTX}, \mathrm{SPM}, \mathrm{NP}, \mathrm{P}, \mathrm{RP}\}$ *for* $i = 1, 2$. P_1 *and* P_2 *are structural equivalent, written* $P_1 \equiv P_2$*, if and only if* $(A_{\mathrm{DO}^{(\mathrm{P}_1)}} = A_{\mathrm{DO}^{(\mathrm{P}_2)}}) \wedge (\bigwedge_{i\in\{\mathrm{SF,DA}\}} C_{i^{(\mathrm{P}_1)}} = C_{i^{(\mathrm{P}_2)}}) \wedge (S_{\mathrm{Cond}^{(\mathrm{P}_1)}} = S_{\mathrm{Cond}^{(\mathrm{P}_2)}}) \wedge (\bigwedge_{i\in\Gamma_{\mathrm{SE}}} F^{i}_{\mathrm{SE}^{(\mathrm{P}_1)}} = F^{i}_{\mathrm{SE}^{(\mathrm{P}_2)}}) \wedge (\bigwedge_{i\in\Gamma_{\mathrm{IE}}\cup\Gamma_{\overline{\mathrm{IE}}}} F^{i}_{\mathrm{IE}^{(\mathrm{P}_1)}} = F^{i}_{\mathrm{IE}^{(\mathrm{P}_2)}}) \wedge (\bigwedge_{i\in\Gamma_{\mathrm{EE}}} F^{\overline{i}}_{\mathrm{EE}^{(\mathrm{P}_1)}} = F^{\overline{i}}_{\mathrm{EE}^{(\mathrm{P}_2)}}) \wedge (F_{\mathrm{T}^{(\mathrm{P}_1)}} = F_{\mathrm{T}^{(\mathrm{P}_2)}}) \wedge (\bigwedge_{i\in\Gamma_{\mathrm{SP}}} F^{i}_{\mathrm{SP}^{(\mathrm{P}_1)}} = F^{i}_{\mathrm{SP}^{(\mathrm{P}_2)}}) \wedge (\bigwedge_{i\in\Gamma_{\mathrm{XT}}} \bigwedge_{j\in\Gamma_{\mathrm{XG}}} F^{i}_{\mathrm{j}^{(\mathrm{P}_1)}} = F^{i}_{\mathrm{j}^{(\mathrm{P}_2)}}) \wedge (\bigwedge_{i\in(\Gamma_{\mathrm{IG}}\cup\Gamma_{\mathrm{CG}}\cup\Gamma_{\mathrm{PG}})} F_{i^{(\mathrm{P}_1)}} = F_{i^{(\mathrm{P}_2)}}) \wedge (\bigwedge_{i\in\Gamma_{\mathrm{Cat}}} \Phi^{\mathrm{Att}}_{i^{(\mathrm{P}_1)}} = \Phi^{\mathrm{Att}}_{i^{(\mathrm{P}_2)}}) \wedge (\bigwedge_{i\in\Gamma_{\mathrm{BdyTX}}} \Phi^{i}_{\mathrm{IE}^{(\mathrm{P}_1)}} = \Phi^{i}_{\mathrm{IE}^{(\mathrm{P}_2)}}) \wedge (\bigwedge_{i\in\Gamma_{\mathrm{Misc}}} \Phi_{i^{(\mathrm{P}_1)}} = \Phi_{i^{(\mathrm{P}_2)}}) \wedge (\bigwedge_{i\in\{\mathrm{SE,EE}\}} \Phi^{\mathrm{Bdy}}_{i^{(\mathrm{P}_1)}} = \Phi^{\mathrm{Bdy}}_{i^{(\mathrm{P}_2)}})$.

Two BPMN processes are regarded as structural equivalent if they have (i) the same sets of data objects, sequence flows, directed associations, conditions, events, tasks, subprocesses, gateways and attributes; and (ii) the same results for all functions.

Proposition 11 *The relation $\equiv$ is transitive.*
Proof. *By similar argument as Proposition 1.* □

Proposition 12 *The relation $\equiv$ is an equivalence.*
Proof. *Analogous to Proposition 2.* □

We introduce a notation $P \rightarrow^* P'$ to represent the transformation of process P into a unique process P' through one or more applications of the inner-outer-DXG form, eliminated-inner-DXG form, start-event-implicit-PFG form, implicit-DXMG form or DXDG-implicit-start-event form.

Definition 48 (Behavioural Equivalence) *Let* $BPD_1 = (S^1_{\mathrm{POOL}}, S^1_{\mathrm{P}}, \Phi^1_{\mathrm{POOL}\rightarrow\mathrm{P}}, C^1_{\mathrm{MF}})$, $POOL_i \in S^1_{\mathrm{POOL}}$, $P_{i^{(\mathrm{D}_1)}} \in S^1_{\mathrm{P}}$, $\Phi^1_{\mathrm{POOL}\rightarrow\mathrm{P}}(POOL_i) = P_{i^{(\mathrm{D}_1)}}$, $C^1_{\mathrm{MF}} = \bigcup_{POOL_1, POOL_2 \in S^1_{\mathrm{POOL}}} (\{PName^{(POOL_1)}\} \times \{PName^{(POOL_2)}\})$, $BPD_2 = (S^2_{\mathrm{POOL}}, S^2_{\mathrm{P}}, \Phi^2_{\mathrm{POOL}\rightarrow\mathrm{P}}, C^2_{\mathrm{MF}})$, $POOL_i \in S^2_{\mathrm{POOL}}$, $P_{i^{(\mathrm{D}_2)}} \in S^2_{\mathrm{P}}$, $\Phi^2_{\mathrm{POOL}\rightarrow\mathrm{P}}(POOL_i) = P_{i^{(\mathrm{D}_2)}}$, $C^2_{\mathrm{MF}} = \bigcup_{POOL_1, POOL_2 \in S^2_{\mathrm{POOL}}} (\{PName^{(POOL_1)}\} \times \{PName^{(POOL_2)}\})$ *and* $C^1_{\mathrm{MF}} = C^2_{\mathrm{MF}}$ *for* $i = 1, \ldots, n$. BPD_1 *and* BPD_2 *are behavioural equivalent, written,* $BPD_1 \doteqdot BPD_2$, *if and only if* $(P_{i^{(\mathrm{D}_1)}} \equiv P_{i^{(\mathrm{D}_2)}}) \vee (P'_{i^{(\mathrm{D}_1)}} \equiv P'_{i^{(\mathrm{D}_2)}})$ *where* $P_{i^{(\mathrm{D}_1)}} \rightarrow^* P'_{i^{(\mathrm{D}_1)}}$ *and* $P_{i^{(\mathrm{D}_2)}} \rightarrow^* P'_{i^{(\mathrm{D}_2)}}$ *for* $i = 1, \ldots, n$.

The motivation for introducing behavioural equivalence is to formally describe when a business process model is a substitute for another business process model.

Proposition 13 *The relation $\doteqdot$ is transitive.*
Proof. *By similar argument as Proposition 1.* □

Proposition 14 *The relation $\doteqdot$ is an equivalence.*
Proof. *Analogous to Proposition 2.* □

Having established a formal foundation for the equivalences of BPMN models, we shift the emphasis away from theoretical aspect to the applicability and practicality of the mathematical framework. In Figures 10 and 11, two business process diagrams which are structurally different are delineated. The discrepancies lie in the fact that they have different numbers of data-based exclusive decision gateways, data-based exclusive merge gateways and sequence flows.

Business process diagram 1 (Figure 10) is composed of two pools: $Pool_1$ and $Pool_2$. The pool $Pool_1$ contains a BPMN process that can be restructured and simplified through a sequence of transformations based on Definitions 37, 39, 43 and 45. The application of Definition 45 with the objective of eliminating the none start event results in an equivalent BPMN process in DXDG-implicit-start-event form as shown in Figure 12.

By swapping the pair of inner data-based exclusive decision gateway and data-based exclusive merge gateway consisting of outgoing and incoming sequence flows to and

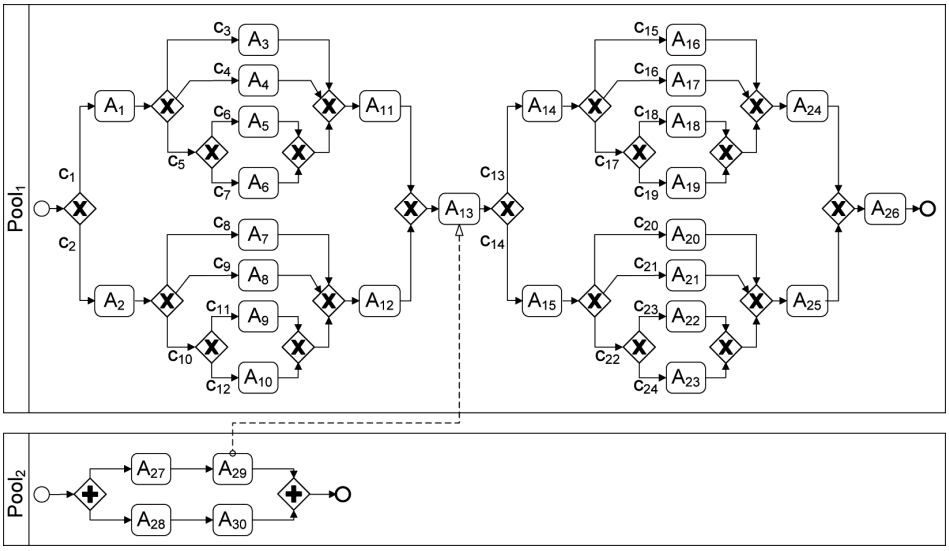

Fig. 10. Business process diagram 1

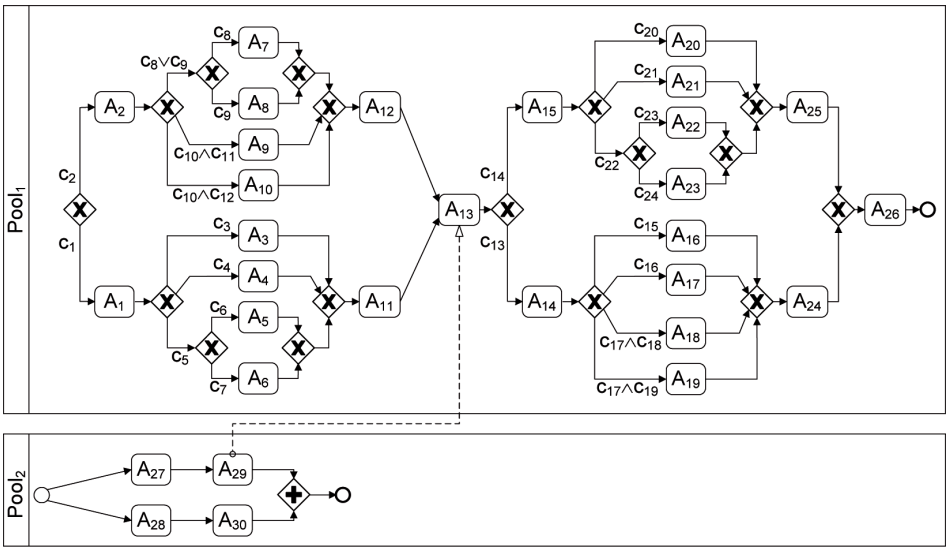

Fig. 11. Business process diagram 2

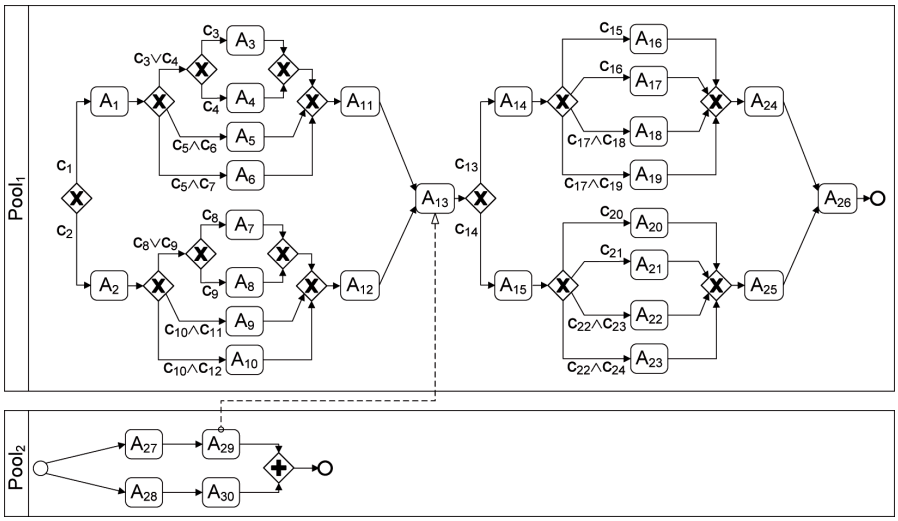

Fig. 12. Business process diagram 3

from the activities A_5 and A_6 with the pair of outer data-based exclusive decision gateway and data-based exclusive merge gateway comprising outgoing and incoming sequence flows to and from the activities A_3 and A_4, an equivalent BPMN process is obtained as depicted in Figure 12. The condition expressions c_6 and c_7 are changed to $c_5 \wedge c_6$ and $c_5 \wedge c_7$, respectively. A new conditional expression $c_3 \vee c_4$ is added to the incoming sequence flow of the data-based exclusive decision gateway (Figure 12). The conditional expression c_5 (Figure 10) is deleted in accordance to Definition 37.

Consider the inner data-based exclusive decision gateway associated with conditional expressions c_{11} and c_{12} and the corresponding data-based exclusive merge gateway as well as the outer data-based exclusive decision gateway associated with conditional expresssions c_8, c_9 and c_{10} and the corresponding data-based exclusive merge gateways (Figure 10). An application of Definition 37 yields an equivalent BPMN process in inner-outer-DXG form as illustrated in Figure 12.

Two successive applications of Definition 43 eliminate the two data-based exclusive merge gateways that connect, respectively, to the activities A_{13} and A_{26}. The resulting BPMN process, which is in implicit-DXMG form, is produced as delineated in Figure 12.

Applying Definition 39, the inner data-based exclusive gateway associated with conditional expressions c_{18} and c_{19} and the respective data-based exclusive merge gateway are eliminated to generate an equivalent BPMN process in eliminated-inner-DXG form (Figure 12). A repeated application of Definition 39 to remove the inner data-based exclusive decision gateway associated with conditional expressions c_{23} and c_{24} along with the corresponding data-based exclusive merge gateway, we get an equivalent BPMN process in eliminated-inner-DXG form as shown in Figure 12.

The parallel fork gateway with incoming sequence flow from the none start event in pool $Pool_2$ is removed to obtain an equivalent BPMN process in start-event-implicit-PFG form as depicted in Figure 12 in accordance to Definition 41. Combining all these transformations together gives the business process diagram in Figure 12.

In the same spirit, the inner data-based exclusive decision gateway associated with conditional expressions c_6 and c_7 (Figure 11) as well as the respective data-based exclusive merge gateway are swapped with the outer data-based exclusive decision gateway associated with conditional expressions c_3, c_4 and c_5 along with the corresponding data-based exclusive merge gateway based on Definition 37. The pair of inner data-based exclusive gateways connecting the activities A_{22} and A_{23} is eliminated in accordance to Definition 39. The data-based exclusive merge gateway with incoming sequence flows from the activities A_{24} and A_{25} is removed through the use of Definition 43. The transformed diagram is depicted in Figure 13.

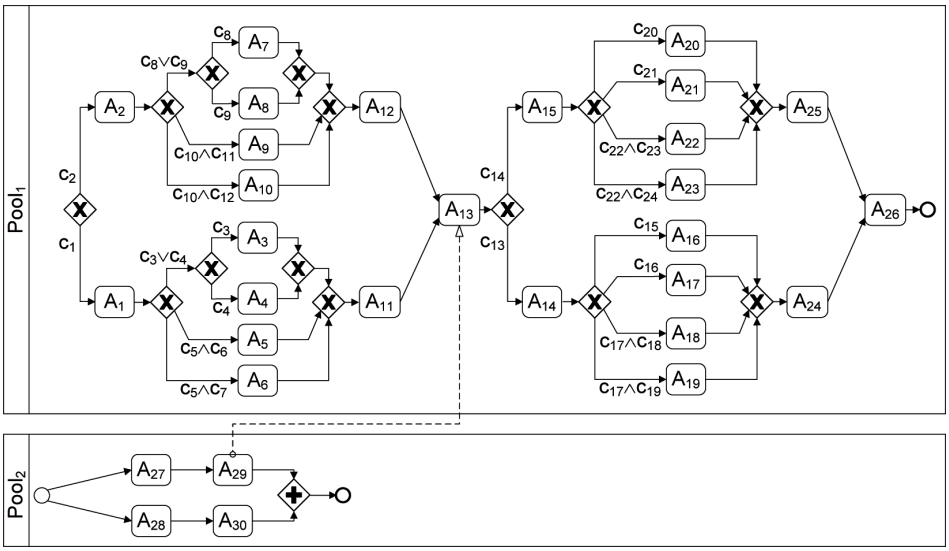

Fig. 13. Business process diagram 4

According to Definition 47, the two BPMN processes in $Pool_1$ of Figures 12 and 13 are structural equivalent. Likewise, the two BPMN processes in $Pool_2$ of Figures 12 and 13 are also considered as structural equivalent. The business process diagrams in Figures 12 and 13 are behavioural equivalence as stipulated by Definition 48. Informally, we say that the two business process diagrams have the same behaviour even though they are represented in different forms.

6. Conclusions and Future Work

The theoretical and practical aspects of the equivalences of UML activity diagrams and BPMN processes are examined in our two recent studies [3, 4]. Nonetheless, there remains an unsettled question on how to determine the equivalence of two BPMN models. This paper closes the gap by advancing a theory of substituting equals for equals.

In summary, we have built a mathematical framework for BPMN. Various sorts of equivalences including IO-DXG-equivalence, EI-DXG-equivalence, SEImpl-PFG-equivalence, Impl-DXMG-equivalence, DXDG-ImplSE-equivalence and structural equivalence have been furnished. The behavioural equivalence of business process diagrams has been explored from a formal perspective. An illustrative example has been utilized to demonstrate the practicality of the proposed framework.

Following this thread of work, we aim to further develop it along several tracks:

(i) the automation of the detection for BPMN processes and BPMN models that are interchangeable;

(ii) the construction of software tools for yielding equivalent BPMN processes and BPMN models;

(iii) the assessment of the disciplined approach by means of real-life business processes; and

(iv) the identification of other types of equivalences.

References

1. OMG: *Business process modeling notation*, v1.2, January 2009. http: //www.bpmn.org/; accessed February 13, 2010.
2. OMG: *UML 2.0 superstructure specification*, August 2005. http:// www.omg.org; accessed July 28, 2006.
3. V.S.W. Lam: *Equivalences of BPMN processes.* Service Oriented Computing and Applications, 3(3):189–204, 2009.
4. V.S.W. Lam: *Theory for classifying equivalences of UML activity dia- grams*. IET Software, 2(5):391–403, 2008.
5. OMG: *Business process modeling notation specification*, February 2006. http://www.bpmn.org/; accessed December 28, 2007.
6. R. Milner, J. Parrow, and D. Walker: *A calculus of mobile process (Parts I and II)*. Information and Computation, 100:1–77, 1992.
7. R. Milner: *The polyadic π-calculus: A tutorial*. In Logic and Algebra of Specification, Proceedings of International NATO Summer School, volume 94, pages 203–246. Springer-Verlag, 1993.

8. J. Parrow: *An introduction to the π-calculus.* In A. Bergstra, J.A. Ponse and S.A. Smolka, editors, Handbook of Process Algebra, chapter 8, pages 479–543. Elsevier Science, 2001.

9. C.A.R. Hoare: *Communicating Sequential Processes*. Prentice-Hall, 1985.

10. A. Bog, F. Puhlmann, and M. Weske: *The PiVizTool: Simulating choreographies with dynamic binding*. In Demo Session of the 5th International Conference on Business Process Management, 2007. http://bpt.hpi.uni-potsdam.de/pub/Public/Frank Puhlmann/bpm2007-piviztool.pdf; accessed February 17, 2008.

11. A. Bog, and F. Puhlmann: *A tool for the simulation of π-calculus systems*. In Open.BPM 2006: Geschäftsprozessmanagement mit Open Source-Technologien, 2006. http://bpt.hpi.uni-potsdam.de/pub/Public/FrankPuhlmann/PiSimulator_openBPM.pdf; accessed January 9, 2009.

12. A. Bog: *Introduction to PiVizTool*. Hasso Plattner Institute, University of Potsdam, 2006. http://bpt.hpi.uni-potsdam.de/pub/ Piworkflow/Simulator/piviztool-intro.pdf; accessed January 13, 2009.

13. A. Bog: *A visual environment for the simulation of business processes based on the pi-calculus.* Master's thesis, Hasso Plattner Institute, University of Potsdam, 2006. http://bpt.hpi.uni-potsdam.de/pub/ Public/FrankPuhlmann/AnjaBogThesisFinal.pdf; accessed January 13, 2009.

14. R.M. Dijkman, M. Dumas, and C. Ouyang: *Semantics and analysis of business process models in BPMN*. Information and Software Technology, 50(12):1281–1294, 2008.

15. P.Y.H. Wong and J. Gibbons: *A process semantics for BPMN*. In Proceedings of the 10th International Conference on Formal Engineering Methods, LNCS 5256, pages 355–374, 2008.

16. P.Y.H. Wong and J. Gibbons: *Verifying business process compatibility*. In Proceedings of the 8th International Conference on Quality Software, pages 126–131, 2008.

17. Formal Systems (Europe) Ltd. *Failures-Divergence Refinement: FDR2 User Manual*, May 2003. http://www.fsel.com/fdr2_download. html; accessed January 20, 2005.

18. P.Y.H. Wong and J. Gibbons: *A relative timed semantics for BPMN*. Electronic Notes in Theoretical Computer Science, 229(2):59–75, 2009.

19. P.Y.H. Wong and J. Gibbons: *Formalisations and applications of BPMN*. Science of Computer Programming, sep 2009.

20. F. Puhlmann: *Soundness verification of business processes specified in the pi-calculus*. In CoopIS 2007, LNCS 4803, pages 6–23, 2007.

21. S. Briais: *The ABC User's Guide*, 2005. http://lamp.epfl.ch/ sbriais/abc/abc_ug.pdf; accessed February 17, 2008.

22. C. Ou-Yang and Y.D. Lin: *BPMN-based business process model feasibility analysis: A Petri Net approach.* International Journal of Production Research, 46(14):3763–3781, 2008.

23. A.V. Ratzer, L. Wells, H.M. Lassen, M. Laursen, J.F. Qvortrup, M.S. Stissing, M. Westergaard, S. Christensen, and K. Jensen: *CPN tools for editing, simulating, and analysing coloured petri nets*. In ICATPN 2003, LNCS 2679, pages 450–462. Springer-Verlag, 2003.

24. I. Raedts, M. Petkovic, Y.S. Usenko, J.M. van der Werf, J.F. Groote, and L. Somers: *Transformation of BPMN models for behaviour analysis.* In MSVVEIS 2007, pages 126–137, 2007.

25. V.S.W. Lam and J. Padget: *Analyzing equivalences of UML state- chart diagrams by structural congruence and open bisimulations.* In Proceedings of 2003 IEEE Symposium on Human Centric Computing Languages and Environments, pages 137–144. IEEE Computer Society, 2003.

26. W. Gruber: *Modelling and Transformation of Workflows with Temporal Constraints*. PhD thesis, Vienna University of Technology, 2003. http: //www.isys.uni-klu.ac.at/PDF/2003-0178-WLG.pdf; accessed January 13, 2009.

27. J. Eder, W. Gruber, and H. Pichler: *Transforming workflow graphs.* In First International Conference on Interoperability of Enterprise Software and Applications, pages 23–25, 2005.

28. S.A.White and D. Miers: *BPMN Modeling and Reference Guide*. Future Strategies Inc., 2008.

29. R. Johnsonbaugh: *Discrete Mathematics*. Macmillan, revised edition, 1986.