On the Formal Semantics of the Extended Compliance Rule Graph *

David Knuplesch[1], Manfred Reichert[1], Linh Thao Ly[1], Akhil Kumar[2], and Stefanie Rinderle-Ma[3]

[1] Institute of Databases and Information Systems, Ulm University, Germany
david.knuplesch,manfred.reichert,thao.ly@uni-ulm.de
[2] Smeal College of Business, Pennsylvania State University, PA, USA
akhilkumar@psu.edu
[3] Faculty of Computer Science, University of Vienna, Austria
stefanie.rinderle-ma@univie.ac.at

Abstract. A fundamental challenge for any process-aware information system is to ensure compliance of modeled and executed business processes with imposed compliance rules stemming from guidelines, standards and laws. Such compliance rules usually refer to multiple process perspectives including control flow, data, time, and resources as well as interactions with business partners. On one hand, compliance rules should be comprehensible for domain experts who must define and apply them. On the other, compliance rules should have precise semantics such that they can be automatically processed. In this context, providing a visual compliance rule language, which hides formal details from rule designers, is crucial in order to enable an intuitive way of modeling. So far, visual compliance rule languages have focused on the control flow perspective, but lack adequate support for the other perspectives. To remedy this drawback, this report introduces the extended Compliance Rule Graph language and its formal semantics.

Keywords: business process compliance, compliance rule graphs, business process modeling, business intelligence, formal semantics

1 Introduction

During the last decade, numerous approaches for ensuring the correctness of business processes have been discussed [2–5]. Most of them focus on syntactical correctness and process model soundness (e.g., absence of deadlocks and lifelocks). However, business processes must also comply with semantic rules stemming from domain-specific requirements such as corporate standards, legal regulations, guidelines or best practices [6, 7]. Summarized under the notion of *business process compliance*, existing approaches have mostly considered

* This work was done within the research project C³Pro funded by the German Research Foundation (DFG), Project number: RE 1402/2-1, and the Austrian Science Fund (FWF) under project number: I743. Parts of this technical report will be published in [1].

compliance issues related to the control flow perspective of single processes. In turn, only a few approaches consider the data, resource, and time perspectives in this context, although they are crucial as well [8–11]. Furthermore, cross-organizational scenarios characterized by interacting and collaborating business processes of various parties have not been properly considered so far [12]. In this context, compliance requirements need to be specified for both local and global processes as well. Note that this requires to consider the data, resource, and time perspectives as well as interactions between business partners (i.e. messages exchanged) as well. As examples consider the compliance rules in Table 1, which are imposed on a cross-organizational process scenario involving the two business partners *reseller* and *manufacturer*. In particular, as shown by the highlighted terms in Table 1, compliance rules may refer to the data, time, and resource perspectives of business processes as well as to interactions with business partners.

Compliance rule c_1 refers to the interactions between a reseller and manufacturer (*request* and *reply*) after a particular point in time (*3rd January, 2013*) as well as the maximum time distance between them (*within three days*). In turn, the data perspective of compliance rules is emphasized by compliance rule c_2 of the manufacturer. It forbids changing an *order* after having started the corresponding *production* task. Compliance rule c_3 combines the interaction, time, and data perspectives. Finally, compliance rule c_4 introduces the resource perspective (*member of the order processing department* and *another member of the same department with supervisor status*). In addition, c_4 considers the data perspective (e.g. *new customer* and *total amount greater than €5,000*) and the time perspective (*at most three days*). Particularly, c_4 shows that the different perspectives might be relevant for the same rule and hence must not be considered in an isolated manner.

When comparing c_4 and c_2 with c_1 and c_3, one can observe two different viewpoints: c_4 and c_2 are expressed from the viewpoint of the manufacturer (i.e., local view), whereas c_1 and c_3 reflect a global view. Note that such distinction

Table 1: Examples of compliance rules for order-to-delivery processes

c_1	Any *request* sent from the reseller to the manufacturer *after January 3rd, 2013* should be *replied* by the manufacturer *within three days*.
c_2	After starting the production related to a particular *order*, the latter must not be changed anymore.
c_3	When the manufacturer *sends a bill* with an *amount lower than €5,000* to the reseller, the latter must make the payment *within 7 days*.
c_4	After receiving a production request message from the reseller, which refers to a *new customer* and has a *total amount greater than €5,000*, the solvency of this customer must be checked by a *member of the order processing department*. Based on the result of this check, *another member of the same department with supervisor status* must approve the request. Finally, the approval result must be sent to the reseller *at most three days* after receiving the original request.

between local and global views is common to cross-organizational collaboration scenarios not only in the context of process compliance [12, 13]. For example, BPMN 2.0 provides collaboration and choreography diagrams to express these different viewpoints.

Several approaches for formally capturing compliance requirements at different abstraction levels (e.g., temporal logics [14]) exist. In particular, they also enable the automatic verification of the conformance of business processes with respective compliance rules. As the use of formal languages for specifying compliance rules would be too intricate, rule patterns hiding formal detail from rule modelers have been proposed [9, 11]. Furthermore, a few approaches also consider more advanced issues like, for example, the use of data conditions in the context of compliance requirements. However, existing approaches are usually restricted to a specific subset of rule patterns. In turn, languages employing visual notations, like the compliance rule graph approach [15] or BPSL [16], combine an intuitive notation with the advantages of a formal language. However, the meta-analyses and case studies, we conducted in domains like higher education [17], medicine [18–21] and automotive engineering [22, 23], have revealed that these visual compliance rule languages still lack support for the time, data, and resource perspective of business processes and do not consider cross-organizational scenarios with interacting partners [12]. Overall, the following fundamental requirements for visual compliance rule languages need to be considered:

- In addition to the control flow perspective, the data, resource and time perspectives of compliance requirements must be properly captured.
- To not only consider process orchestrations, but cross-organizational scenarios as well, it becomes necessary to integrate the interaction perspective with compliance rule languages as well.
- To provide tool support for both the modeling and verification of compliance rules, both their syntax and semantics must be formalized.

To cope with the discussed shortcomings, this report introduces the *extended Compliance Rule Graph* (eCRG) and its formal semantics. The eCRG builds on the *Compliance Rule Graph* (CRG) language developed in previous work [15, 24], but additionally comprises elements enabling the visual modeling of compliance rules with the support of the process, data, time, and resource perspectives. Furthermore, we introduce concepts that allow defining compliance rules in respect to message flows and partner interactions; i.e., the eCRG language is able to specify compliance requirements for cross-organizational process scenarios (i.e. processes choreographies). For defining the formal semantics of the eCRG, we provide a transformation into FOL formula. Note that the latter can be evaluated over process traces to *a posteriori* verify the compliance of business processes.

Altogether, the eCRG allows capturing compliance requirements at an abstract level, while enabling the specification of verifiable compliance rules in the context of cross-organizational scenarios, including the process, data, time, and resource perspectives.

The remainder of this report is structured as follows: Section 2 discusses related work. Section 3 introduces the *extended Compliance Rule Graph* (eCRG). The formal semantics of the eCRG language is specified in Section 4, whereas Section 5 provides a pattern-based evaluation. Finally, Section 6 concludes the report and provides an outlook on future research.

2 Related Work

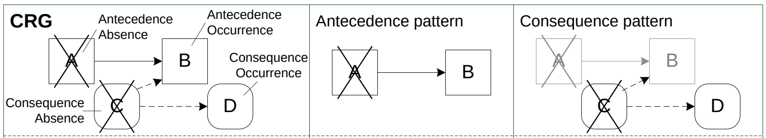
During the last years the interaction, time, resource, and data perspectives have been considered in business process modeling in addition to the control flow perspective (e.g., [25–33]).
The integration of business process compliance throughout the entire process lifecycle has been discussed in [24, 34–36]; [37] examined compliance issues in the context of cross-organizational processes developing a logic-based formalism for describing the semantics of normative specifications as well as compliance checking procedures. This approach allows modeling business obligations regulating the execution of business processes. In turn, [38] introduced a semantic layer that interprets process instances according to an independently designed set of internal controls. Furthermore, there exist approaches using semantic annotations to ensure compliance [39]. An approach for checking the compliance of process models against semantic constraints as well as for ensuring the validity of process change operations based on Mixed-Integer Programming formulation is proposed in [40]. It introduces the notions of *degree of compliance*, *validity of change operations*, and *compliance by compensation*. Further, [9] uses alignments to detect compliance violations in process logs. To verify whether compliance rules are fulfilled by process models at design time, many approaches apply model checking techniques [14, 16, 41–44]; some of them address the data and time perspectives as well. In order ensure *compliability* and *global compliance* of cross-organizational processes, where partners only provide restricted views on their local processes, [45, 46] apply model checking as well. Other approaches for verifying compliance apply the notion of *semantic congruence* [47] or use *petri-nets* [48] and consider the data and time perspectives as well.
The approach described in [41, 49] for visually modeling compliance considers the control flow and data perspectives. It is based on linear temporal logic (LTL), which allows modeling the control flow perspective based on operators like *next*, *eventually*, *always*, and *until*. Other visual approaches for compliance rule modeling [15, 16, 50] focus on control flow and partially the data perspective, but ignore the other perspectives mentioned.

3 Extended Compliance Rule Graphs

This section introduces *extended Compliance Rule Graphs (eCRG)* – a visual notation for compliance rule modeling covering the process, interaction, time, resource, and data perspectives. Secion 3.1 introduces fundamentals of CRGs. Its extensions are introduced stepwise in Sections 3.2–3.6.

3.1 Fundamentals of Compliance Rule Graphs

The compliance rule graph (CRG) language [15, 24] allows visually modeling compliance rules whose semantics is defined over event traces. More precisely, a CRG is an acyclic graph consisting of an *antecedence pattern* as well as at least one related *consequence pattern*. Both patterns are modeled using *occurrence* and *absence nodes*, which indicate the occurrence or absence of events (e.g., related of the execution of a particular task). Edges between such nodes indicate control flow dependencies. As illustrated in Fig. 1, a trace is considered as compliant with a CRG iff for each match of the antecedence pattern there is at least one corresponding match of every consequence pattern. Furthermore, a trace is considered as *trivially compliant* iff there is no match of the antecedence pattern. For example, the CRG from Fig. 1 expresses that for each B not preceded by an A, there must occur a D, which is not preceded by any C also preceding the corresponding B.



< E, **D**, F, G, **B** >	compliant	only match < E, D, F, G, **B** >	< E, **D**, F, G, B >
< **D**, F, **C**, E, **B** >	compliant	only match < D, F, C, E, **B** >	< **D**, F, C, E, B > (C is after D!)
< **A, B**, **C**, E, **D** >	trivially compliant	no match (< ***A*, B**, C, E, D >)	-
< **C**, F, **B**, G, E >	violation	only match < C, F, **B**, G, E >	no match (missing D)
< **B**, **C**, **D**, E, **B** >	violation	1st match < **B**, C, D, E, B >	< B, C, **D**, E, B >
		2nd match< B, C, D, E, **B** >	no match (< B, **C**, **D**, E, B >)

Fig. 1: CRG example and semantics over execution traces

In the following sections, we introduce the *extended Compliance Rule Graph* (eCRG) language, which is based on CRGs. In addition to using nodes and connectors (i.e., edges), eCRG allows for *attachments*. The latter represent constraints to the nodes or edges they are attached to. Furthermore, an eCRG may contain *instance nodes* representing particular instances, which exist independently from the respective rule (e.g. a particular employee *Mr. Smith*, date *3rd January 2013*, or role *supervisor*). Accordingly, instance nodes are neither part of the antecedence nor the consequence pattern.

3.2 Process Perspective

We first consider the process (i.e., control flow) perspective. The eCRG elements for modeling the process (i.e. control flow) perspective of compliance rules are introduced in Fig. 2. Since the extensions are based on the CRG language, there exist four different task elements, i.e., *antecedence occurrence*, *antecedence absence*, *consequence occurrence*, and *consequence absence tasks*. These allow expressing whether or not particular tasks must be executed. In addition, two different kinds of *sequence flow connectors* are provided that may be used to constrain

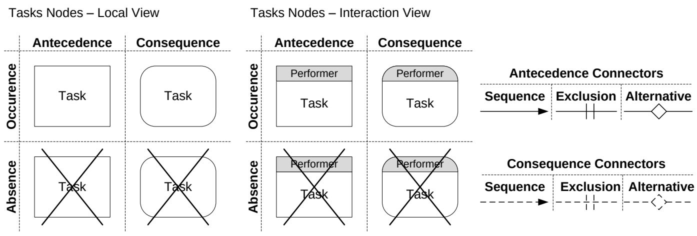


Fig. 2: eCRG elements of the process perspective

the execution sequence of tasks. Note that the absence of sequence flow indicates a parallel flow. To clearly distinguish between *start-start*, *start-end*, *end-start*, and *end-end* constraints on the execution sequence of tasks, sequence flow edges are either connected to the right or left border of a task node. Furthermore, *exclusive connectors* allow modeling mutual excluding tasks. *Alternative connectors*, in turn, express that at least one of the connected tasks must occur. Note that exclusive as well as alternative connectors may only connect nodes that are either part of the antecedence or the consequence pattern.

Fig. 4A shows an example of a start-start constraint on the execution sequence of tasks. It depicts the process perspective of compliance rule c_2 from Table 1 that disallows executing task *change order* after task *production* is started.

3.3 Interaction Perspective

The interaction perspective of business processes is crucial in cross-organizational scenarios [51, 13]. It covers constraints on the messages exchanged between business pratners and the *interaction view* of the eCRG meta-model. Message exchanges are expressed in terms of particular nodes that reflect the events of *sending* and *receiving a message*. In turn, a *message flow* denotes the dependency between the events representing the sending and receiving of a particular message (cf. Fig. 3).

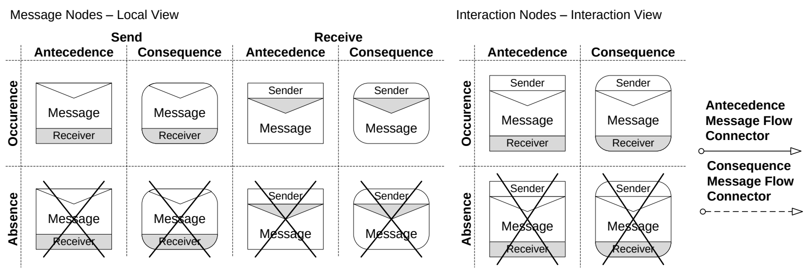


Fig. 3: eCRG elements of the interaction perspective

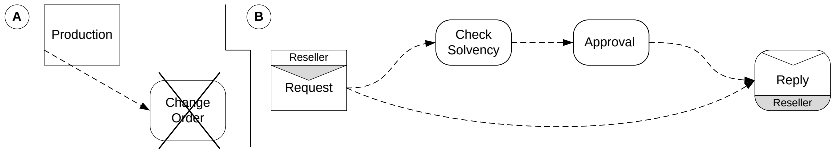


Fig. 4: Local view on c_2 and c_4 with process and interaction perspectives

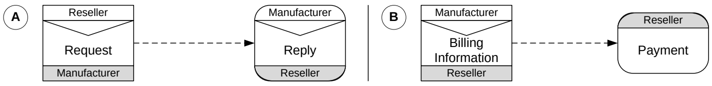


Fig. 5: Interaction view on c_1 and c_3 with process and interaction perspectives

In Fig. 4B, the elements from Fig. 3 are used to model the process and interaction perspective of compliance rule c_4. This rule requires that after receiving message *request* from a reseller, a *solvency check* must be performed first. Then, a decision about *approval* has to be made before replying the *request*. Although the rule modeled in Fig. 4B considers the interaction perspective, using the two message nodes *request* and *reply*, it still represents the view of a particular business partner on its local business processes. We refer to this traditional point of view as the *local view* of a compliance rule. However, when considering the compliance rules c_1 and c_3 from Table 1, one can easily discover a global point of view on cross-organizational processes and related interactions (i.e., the messages exchanged) that corresponds to the BPMN 2.0 *choreography diagram*. In this *interaction view*, interaction nodes (cf. Fig. 3) are used to denote the exchange of a message between two business partners. Since the interaction view spans multiple business partners, task nodes may be annotated with the the business partner responsible (cf. Fig. 2).
Fig. 5A provides an interaction view on compliance rule c_1 from Table 1: After the reseller sends a *request* to the manufacturer, eventually, the manufacturer must *reply*. Furthermore, Fig. 5B provides an interaction view on compliance rule c_3 from Table 1. This rule requires that the reseller must perform task *payment* after having received *billing information* from the manufacturer.

3.4 Time Perspective

The time perspective of a business process deals with temporal constraints that need to be obeyed by instances of the process [28, 29].
Having a closer look on the original definition of compliance rules c_1 and c_3 from Table 1, it becomes clear that the eCRGs from Figs. 5A and 5B do not fully specify them yet. In particular, the time distances between the interactions and tasks have not been modeled. Fig. 6 provides elements for modeling *points in time* and *time conditions* in compliance rules. The latter may be attached to task nodes as well as sequence or message flow connectors to either constrain the duration of a task or the time distance between tasks, messages or points in time. Additionally, *time distance connectors* are introduced that must be attached

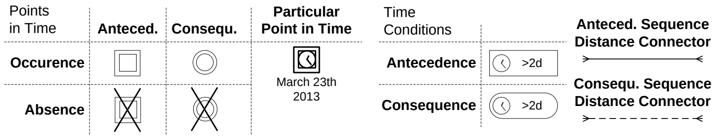


Fig. 6: eCRG elements of the time perspective

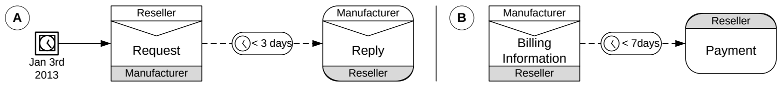


Fig. 7: Interaction view on c_1 and c_3 with process, interaction and time perspectives

with a time condition. Respective time distance connectors and related time conditions then allow constraining the time distance between tasks, messages or points in time without implying a particular sequence.

Fig. 7A combines the interaction and time perspectives of compliance rule c_1. This visual representation of c_1 covers exactly the semantics of the compliance rule described in Table 1. In Fig. 7B, the interaction and time perspectives of c_3 are provided. This compliance rule requires that at most seven days after the manufacturer sends *billing information* to the reseller, the latter must perform task *payment*.

3.5 Resource Perspective

The resource perspective covers the different kinds of human resources as well as their inter-relations [25, 52]. Further, it allows constraining the assignment of resources to tasks. In the context of our work, we consider resources like *staff member*, *role*, *group*, and *organizational unit* as well as their relations to tasks. Furthermore, we support *resource conditions* and *relations* among resources (cf. Fig. 8). Similar to task nodes, *resource nodes* may be part of the antecedence or consequence patterns. Alternatively, they may represent a particular resource instance (e.g. staff member *Mr. Smith*, or role *supervisor*). In turn, *resource conditions* may constrain resource nodes. Furthermore, the *performing relation* indicates the performer of a task. Finally, *resource relation connectors* express relations between resources. Note that the resource perspective can be easily extended with other kinds of resources if required.

Fig. 9 combines the process, interaction, time, and resource perspectives of compliance rule c_4. This rule requires that at least three days after receiving a *request* of the reseller, a *replay* must be sent to him. Before sending this reply, first of all, task *solvency check* must be performed by a staff member assigned to the particular organizational unit *order processing department*. Following this task, another staff member of the same department with *supervisor* status (i.e., role) must decide whether to grant approval before sending the *reply*.

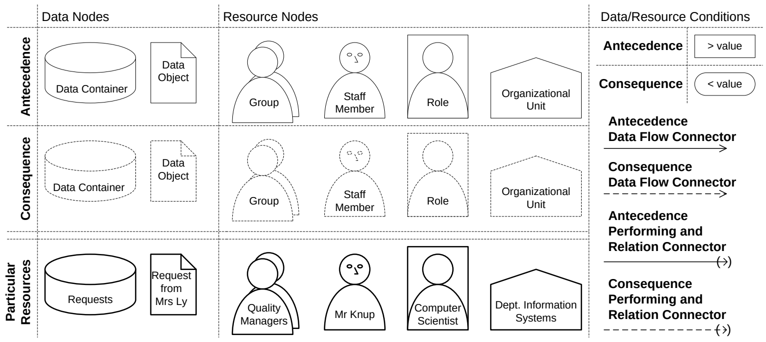


Fig. 8: eCRG elements of the resource and data perspectives

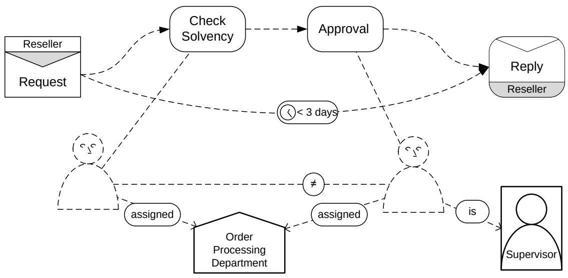


Fig. 9: Local view on c_4 with process, interaction, time, and resource perspectives

3.6 Data Perspective

The data perspective of business processes covers the data objects processed as well as the data flow between the tasks of the process [32, 53, 33]. Fig. 8 introduces eCRG elements for modeling data containers and data objects as well as connectors representing data flow. Thereby, *data containers* refer to process data elements or global data stores. By contrast, *data objects* refer to particular data values and object instances. Similar to resource nodes, *data nodes* may be part of the antecedence or consequence pattern, or represent a particular data container or data object (e.g., data container *student credit points*, document *1st order from Mr. Smith*). Furthermore, *data flow* defines which process tasks read or write which data objects or data container. To constrain data container, data objects, and data flow, *data conditions* may be attached. Finally, *data relation connectors* may be used either to compare different data objects or to constrain the value of data containers at particular points in time.

Figs. 10A-C show the visual modeling of compliance rules c_2, c_3, and c_4 covering the data perspective as well as the other perspectives discussed. Note that each of the depicted eCRG covers the informal semantics described in Table 1.

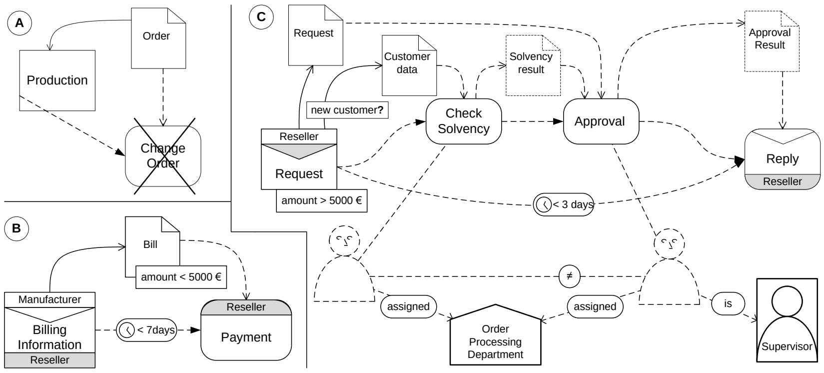


Fig. 10: Local view on c_2 and interaction view on c_3 and c_4, considering the process, interaction, time, resource, and data perspectives

4 Formal Semantics of the eCRG

In the previous section, we have only informally described eCRG semantics. In order to enable automated compliance checking of process logs against eCRG, however, an unambiguous formal semantics is required. For this purpose, this section provides a transformation from eCRG to first-order predicate logic (FOL), which defines how to interpret and evaluate an eCRG over process logs.
How this transformation can be accomplished is described in the following. First, Section 4.1 introduces sets representing fundamental business process concepts (e.g., tasks types, resources, or data values). Second, process logs are formally defined as set of predicates in Section 4.2. Third, Section 4.3 provides a set-based formal description of an eCRG. Fourth, Section 4.4 defines the transformation translating the eCRG into logic formula based on the defined predicates.

4.1 Environmental Sets and Concepts

The eCRG language relates to different perspectives of business processes, i.e., the process, time, data, resource, and interaction perspectives. The formalization of the eCRG language requires a formal definition of theses fundamental concepts defining the *process environment* in Def. 1 as follows:

Definition 1 (Process Environment).

The environment of a business process comprises the following sets:

- $\mathcal{T}$ *is the set of task types*
- $\mathcal{M}$ *is set of message types*
- $\mathfrak{T}$ *is the discrete and ordered set of points in time*
- $\mathcal{R} := \mathcal{R}_S \cup \mathcal{R}_R \cup \mathcal{R}_G \cup \mathcal{R}_U$ *is the set of resources, with* $\mathcal{R}_S$ *being the set of staff members,* $\mathcal{R}_R$ *being the set of roles,* $\mathcal{R}_G$ *being the set of groups,* $\mathcal{R}_U$ *being the set of business units and departments.*
- $\mathcal{D}$ *set of data containers,* $\mathcal{P}$ *set of activity and message parameters,*
- Ω *set of data values and objects (e.g.* 5*,* 1.345*, color red, document-A, report-123, ...),*
- $\mathcal{I}_\mathcal{T}$ *set of task instance identifiers, and*
- $\mathcal{I}_\mathcal{M}$ *set of message instance identifier.*

Further, based on the definition of the environment, we introduce the following sets of conditions and relations for time, data and resources:

- $\mathfrak{C}_T \subseteq \{con_t | con_t \subseteq \mathfrak{T} \to \mathbb{B}\}$ *is the set of time conditions,*
- $\mathfrak{C}_D \subseteq \{con_d | con_d \subseteq \Omega \to \mathbb{B}\}$ *is the set of data conditions,*
- $\mathfrak{C}_R \subseteq \{con_r | con_r \subseteq \mathcal{R} \to \mathbb{B}\}$ *is the set of resource conditions,*
- $\mathfrak{R}_T \subseteq \{rel_t | rel_t \subseteq \mathfrak{T}^2 \to \mathbb{B}\}$ *is the set of time relations,*
- $\mathfrak{R}_D \subseteq \{rel_d | rel_d \subseteq \Omega^2 \to \mathbb{B}\}$ *is the set of data relations, and*
- $\mathfrak{R}_R \subseteq \{rel_r | rel_r \subseteq \mathcal{R}^2 \to \mathbb{B}\}$ *is the set of resource relations.*

InStaff $\in \mathfrak{R}_R$ is a resource relation and InStaff(r, s) indicates that staff member s belongs or is assigned to resource r. Furthermore, we do not explicitly distinguish between time conditions and time relations, since each time condition con_t constitutes a time relation rel^{con_t} with

$$rel^{con_t}(t_1, t_2) :\Leftrightarrow con_t(|t_1 - t_2|).$$

4.2 Process Logs

A common way to capture the execution of a business process is to store execution events in process logs and traces [2]. Usually, the latter contain multiple information about the tasks executed, messages exchanged, or data objects accessed. However, not all logged information is of interest for evaluating whether or not a particular process instance complies with a given compliance rule. Independent from the concrete format of a process log, we introduce a set of predicates describing its relevant information in Def. 2. Obviously, the values of these predicates can be easily derived by processing a concrete process log:

Definition 2 (Execution Log Entries/Predicates).

Let $\mathcal{T}, \mathcal{M}, \mathfrak{T}, \mathcal{R}, \mathcal{D}, \mathcal{I}_\mathcal{T}, \mathcal{I}_\mathcal{M}$ *be defined as in Def. 1. Then, predicate*

- $Start(t, i, tt)$ *expresses that task* $i \in \mathcal{I}_\mathcal{T}$ *of type* $tt \in \mathcal{T}$ *is* **started** *at* $t \in \mathfrak{T}$.
- $End(t, i, tt)$ *expresses that task* $i \in \mathcal{I}_\mathcal{T}$ *of type* $tt \in \mathcal{T}$ *is* **completed** *at* $t \in \mathfrak{T}$.
- $Perform(i, s)$ *expresses that task* $i \in \mathcal{I}_\mathcal{T}$ *was* **performed** *by the staff member* $s \in \mathcal{R}_S$.
- $Receive(t, i, mt, b)$ *expresses that message* $i \in \mathcal{I}_\mathcal{M}$ *of type* $mt \in \mathcal{M}$ *is* **received** *from business partner* $b \in \mathcal{B}$ *at* $t \in \mathfrak{T}$.
- $Send(t, i, mt, b)$ *expresses that message* $i \in \mathcal{I}_\mathcal{M}$ *of type* $mt \in \mathcal{M}$ *is* **sent** *to business partner* $b \in \mathcal{B}$ *at* $t \in \mathfrak{T}$.
- $Parameter(i, p, v)$ *expresses that execution parameter* $p \in \mathcal{P}$ *of task or message* $i \in \mathcal{I}_\mathcal{T} \cup \mathcal{I}_\mathcal{M}$ *has* **parameter value** $v \in \Omega$.
- $Write(t, i, p, v, d)$ *expresses that task or message* $i \in \mathcal{I}_\mathcal{T} \cup \mathcal{I}_\mathcal{M}$ **writes** *value* $v \in \Omega$ *to data container* $d \in \mathcal{D}$ *via output parameter* $p \in \mathcal{P}$ *at* $t \in \mathfrak{T}$.
- $Read(t, i, p, v, d)$ *expresses that task or message* $i \in \mathcal{I}_\mathcal{T} \cup \mathcal{I}_\mathcal{M}$ **reads** *value* $v \in \Omega$ *to data container* $d \in \mathcal{D}$ *via input parameter* $p \in \mathcal{P}$ *at* $t \in \mathfrak{T}$.

Based on predicate **Write***, we define predicate* **Value***:*

$$Value(t_0, o, v) :\Leftrightarrow \exists t_1 \in \mathfrak{T} : t_1 < t_0 \wedge Write(t_1, \cdot, \cdot, v, o) \wedge \nexists t_2 : t_1 < t_2 < t_0 \wedge Write(t_2, \cdot, \cdot, \cdot, o)$$

Note that we use dots $(\cdot)$ *as wildcards, i.e.:*

$$Write(t_1, \cdot, \cdot, \cdot, o) \Leftrightarrow \exists x_1, x_2, x_3 : Write(t_1, x_1, x_2, x_3, o)$$

4.3 Formal Description of eCRG

We now introduce a set-based formal description of the eCRG. Since the latter constitutes a graph, it consists of nodes (e.g. task nodes) and edges (e.g. sequence flow edge). Furthermore, the eCRG nodes and edges may have attachments that specify additional constraints. Thus, we can roughly define an eCRG as specified in Def. 3:

Definition 3 (eCRG).

An eCRG Ψ is a graph $\Psi = (N, E, \diamond)$, where

- *N is the set of nodes,*
- *E is the set of edges, and*
- *$\diamond$ is the set of attachments to nodes and edges.*

As described in Section 3, each element of an eCRG (i.e., node, edge, or attachment) can either be element of the *antecedence pattern* (A) or the *consequence pattern* (C) or be a particular *instance* (I) (e.g., a particular employee *Mr. Smith*, date *3rd January 2013*, or role *supervisor*). In the cases of nodes, the antecedence pattern is partitioned into the *antecedence occurrence pattern* (AO) and the *antecedence absence pattern* (AA), while the consequence pattern is partitioned into the *consequence occurrence pattern* (CO) and the *consequence absence pattern* (CA). Before formally introducing these pattern classes in Def. 7, we first define the various elements of an eCRG (i.e., nodes, edges and attachments) in more detail.

Nodes. The most fundamental elements of the eCRG are nodes. According to the different compliance perspectives the set of nodes N can be partitioned into nodes corresponding to tasks, receiving and sending messages, points in time, data containers, data objects, and resources as follows:

Definition 4 (Nodes of an eCRG).

Let N be the set of nodes of an eCRG $\Psi = (N, E, \diamond)$. Then, N can be partitioned as follows:

$N := T \cup M_R \cup M_S \cup P \cup D \cup O \cup R$, where

- *T is the set of task nodes,*
- *M_R/M_S is the set of receiving/sending message nodes,*
- *P is the set of nodes representing points in time,*
- *D is the set of data container nodes,*
- *O is the set of data object nodes.*
- *$R := R_S \cup R_G \cup R_R \cup R_U$ are the sets of resource nodes for staff members, groups, roles, and organizational units.*

In the following, we use $T_S \cup T_E$ instead of T in order to distinguish whether edges are connected to the left and right side of task nodes. Thereby, T_S is the left side, i.e. it corresponds to the start of a task, whereas T_E is the right side corresponding to completion end of a task. Furthermore, we introduce the function $s : T \to T_S$ ($e : T \to T_E$) to get the start (end) of a task node, and function $t : T_S \cup T_E \to T$ for the opposite direction. Function *type* : $(T \to \mathcal{T}) \cup (M \to \mathcal{M})$ assigns to each task and message node the associated task or message

type. $bp : (M_R \cup M_S) \rightarrow \mathcal{B}$ denotes the business partner that sends or receives a message. Finally, $TMP := T \cup M_R \cup M_S \cup P \subseteq N$ is the set of a all task, message, and points in time nodes, whereas $DR := D \cup O \cup R \subseteq N$ comprises all data container, data object, and resource nodes.
We can partition the set of nodes and each of its subsets into distinct sets of particular instance nodes, antecedence nodes, and absence nodes as well as into particular instance nodes, antecedence occurrence nodes, antecedence absence nodes, consequence occurrence nodes, and consequence absence nodes. We use the exponents ^{I}, ^{A}, ^{C}, ^{AO}, ^{AA}, ^{CO} and ^{CA} to indicate the corresponding subset. Further, the exponent $^{-I}$ denotes that all nodes are considered except particular instance ones.

Edges. The nodes of an eCRG can be connected through a wide range of edges indicating relations and correlations between these nodes. In particular, the set of edges can be partitioned into distinct subsets (cf. Def. 5). The latter contain edges for sequence flow, message flow, and distance in time. Other edge subsets refer to read/write data flow from/to data containers or data objects as well as to the performers of tasks and relations among data objects or resources. Finally, the two last subsets contain edges specifying correlations between conditions to data containers and the particular points in time, when these conditions are evaluated:

Definition 5 (Edges of an eCRG).

We consider E as the set of edges of an eCRG $\Psi = (N, E, \diamond)$. Then, E can be partitioned as follows:

$E := \overrightarrow{sf} \cup \overrightarrow{mf} \cup \overrightarrow{tde} \cup \overrightarrow{wdfc} \cup \overrightarrow{rdfc} \cup \overrightarrow{wdfo} \cup \overrightarrow{rdfo} \cup \overrightarrow{pfm} \cup \overrightarrow{rr} \cup \overrightarrow{dr} \cup \overrightarrow{dcec} \cup \overrightarrow{dceo}$, *where*

- $\overrightarrow{sf} \subseteq (T_S \cup T_E \cup M_S \cup M_R \cup P)^2 \times \mathfrak{R}_T$ *is the set of sequence flow edges including a time condition,*
- $\overrightarrow{mf} \subseteq M_S \times M_R$ *is the set of message flow edges,*
- $\overrightarrow{tde} \subseteq (T \cup M_R \cup M_S \cup P)^2 \times \mathfrak{C}_T$ *is the set of time distance edges including a time condition,*
- $\overrightarrow{wdfc} \subseteq (T \cup M_R \cup M_S) \times (\mathcal{P} \cup \{\cdot\}) \times D$ *is the set of data flow edges writing to a data container,*
- $\overrightarrow{rdfc} \subseteq D \times (T \cup M_R \cup M_S) \times (\mathcal{P} \cup \{\cdot\})$ *is the set of data flow edges reading from a data container,*
- $\overrightarrow{wdfo} \subseteq (T \cup M_R \cup M_S) \times (\mathcal{P} \cup \{\cdot\}) \times O$ *is the set of data flow edges writing a data object,*
- $\overrightarrow{rdfo} \subseteq O \times (T \cup M_R \cup M_S) \times (\mathcal{P} \cup \{\cdot\})$ *is the set of data flow edges reading a data object,*
- $\overrightarrow{pfm} \subseteq T \times R$ *is the set of performing relation edges denoting the performer of a task,*
- $\overrightarrow{rr} \subseteq R^2 \times \mathfrak{R}_R$ *is the set of resource relation edges,*
- $\overrightarrow{dr} \subseteq O \times O \times \mathfrak{R}_D$ *is the set of data relation edges,*
- $\overrightarrow{dcec} \subseteq (T_S \cup T_E \cup M_R \cup M_S \cup P) \times D \times \mathfrak{C}_D$ *is the set of data condition edges constraining a data container at a particular point in time by a data condition,*
- $\overrightarrow{dceo} \subseteq (T_S \cup T_E \cup M_R \cup M_S \cup P) \times D \times O$ *is the set of data condition edges requiring a data container to contain a data object at a particular point in time.*

Further, $src : E \rightarrow N$ ($tgt : E \rightarrow N$) is a function that maps each edge to its source node (target node), whereas $nd : E \rightarrow N, e \mapsto nd(e) := \{src(e), tgt(e)\}$ is a function that maps each edge to the set containing its source and target node. Edges are either part of the antecedence or the consequence pattern. Hence, we can further partition the set of edges and each of its subsets into antecedence and consequence edges. We use exponents ^{A} and ^{C} to indicate when only considering the particular subset.

According to the definition of $\overrightarrow{sf}$, each sequence flow requires a time relation. Thereby, if the graphical representation does not provide a time relation, we just assume the time condition to be $\mathfrak{T}^2$, i.e., true for each input. Further an antecedence edge with a consequence time condition is interpreted as two edges: an antecedence edge without a time condition and a consequence edge with the respective time condition.

Attachments. Nodes and edges of an eCRG may be refined by attachments. In turn, these attachments constitute additional conditions for the respective node or edge. In particular, the set of attachments $\diamond$ can be partitioned into distinct subsets. The latter refer to different data conditions constraining data flow or parameters of tasks or messages. Other attachments express time conditions that constrain the duration of tasks or resource conditions that constrain resources:

Definition 6 (Attachments of an eCRG).

We consider $\diamond$ *as the set of attachments of an eCRG* $\Psi = (N, E, \diamond)$*. Then,* $\diamond$ *can be partitioned as follows:*

$\diamond := {}^{\diamond}dcp \cup {}^{\diamond}doc \cup {}^{\diamond}dfc \cup {}^{\diamond}dfo \cup {}^{\diamond}td \cup {}^{\diamond}rc$*, where*

- ${}^{\diamond}dcp \subseteq (T \cup M) \times \mathcal{P} \times \mathfrak{C}_D$ *attaches a task or message node with a data condition on a particular parameter,*
- ${}^{\diamond}doc \subseteq O \times \mathfrak{C}_D$ *attaches data object nodes with a data condition,*
- ${}^{\diamond}dfc \subseteq (\overrightarrow{wdfc} \cup \overrightarrow{rdfc} \cup \overrightarrow{wdfo} \cup \overrightarrow{rdfo}) \times \mathfrak{C}_D$ *attaches data flow edges with a data condition,*
- ${}^{\diamond}dfo \subseteq (\overrightarrow{wdfc} \cup \overrightarrow{rdfc}) \times O$ *attaches data flow edges from/to a data container with a data object serving as placeholder for the value of the data flow,*
- ${}^{\diamond}td : T \times \mathfrak{R}_T$ *attaches task nodes with a time relation that constraints the duration of the task, and*
- ${}^{\diamond}rc : R \times \mathfrak{C}_R$ *attaches resource nodes with a resource condition.*

Further, $at : \diamond \to N \cup E$ is a function mapping each attachment to the node or edge it corresponds to. Attachments are either part of the antecedence or the consequence pattern. Consequently, we can partition the set of attachments and each of its subsets into antecedence and consequence attachments. Again, we use exponents ^{A} and ^{C} when solely referring to the respective subset.

Pattern Classes. As mentioned in Section 3, nodes, edges, and attachments can be classified as elements of the instance, the antecedence, or the consequence pattern. In the context of nodes, the antecedence and consequence pattern are further partitioned into the antecedence occurrence and the antecedence absence pattern as well as the consequence occurrence and the consequence absence pattern. Def. 7 formally introduces the pattern classes and transfers the finer classification to edges and attachments (i.e., the classification into antecedence occurrence, antecedence absence, consequence occurrence, and consequence absence patterns). For this purpose, we consider pattern classes of the nodes connected by an edge as well as the pattern class of the element an attachment corresponds to.

Definition 7 (Classes of an eCRG).

Let $\Psi = (N, E, \diamond)$ be an eCRG. Then:

- $I := N^I$ *is the instance pattern,*
- $A := N^{AO} \cup N^{AA} \cup E^A \cup \diamond^A$ *is the antecedence pattern,*
- $C := N^{CO} \cup N^{CA} \cup E^C \cup \diamond^C$ *is the consequence pattern,*

- $E^{AO} := \{e \in E^A | nd(e) \subseteq N^{AO}\}$ *is the set of antecedence occurrence edges,*
- $E^{AA} := \{e \in E^A | nd(e) \cap N^{AA} \neq \varnothing \wedge nd(e) \cap N^{AO} \neq \varnothing\}$ *is the set of antecedence absence edges,*
- $E^{CO} := \{e \in E^C | nd(e) \cap N^C \neq \varnothing \wedge nd(e) \subseteq (N^{CO} \cup N^{AO})\}$ *is the set of consequence occurrence edges,*
- $E^{CA} := \{e \in E^C | nd(e) \cap N^{CA} \neq \varnothing \ \wedge \ nd(e) \cap (N^{CO} \cup N^{AO}) \neq \varnothing\}$ *is the set of consequence absence edges,*

- $\diamond^{AO} := \{a \in \diamond^A | at(a) \in N^{AO} \cup E^{AO}\}$ *is the set of antecedence occurrence attachments,*
- $\diamond^{AA} := \{a \in \diamond^A | at(a) \in N^{AA} \cup E^{AA}\}$ *is the set of antecedence absence attachments,*
- $\diamond^{CO} := \{a \in \diamond^C | at(a) \in N^{AO} \cup E^{AO} \cup N^{CO} \cup E^{CO}\}$ *is the set of consequence occurrence attachments,*
- $\diamond^{CA} := \{a \in \diamond^C | at(a) \in N^{CA} \cup E^{CA}\}$ *is the set of consequence absence attachments,*

- $AO := N^{AO} \cup E^{AO} \cup \diamond^{AO}$ *is the antecedence occurrence pattern,*
- $AA := N^{AA} \cup E^{AA} \cup \diamond^{AA}$ *is the antecedence absence pattern,*
- $CO := N^{CO} \cup E^{CO} \cup \diamond^{CO}$ *is the consequence occurrence pattern,*
- $CA := N^{CA} \cup E^{CA} \cup \diamond^{CA}$ *is the consequence absence pattern, and*
- $\mathcal{C} := \{I, AO, AA, CO, CA\}$ *is the set of pattern classes.*

4.4 Transformation of eCRG

After having defined a formal representation of the process environment, process logs and compliance rules, we are able to define the semantics of a particular compliance rule. For this purpose, we transform the eCRG into a first-order predicate logic formula in this subsection. Instead of nodes, edges and attachments, the resulting logic formula comprises variables derived from the eCRG elements. However, note that the set of variables is not isomorphic to the sets of nodes and edges. Thus, Def. 8 introduces the variables required, before describing the transformation in detail. In this context, we use relation $\bar{\in}$ to indicate the set of values a variable may take.

Definition 8 (Variables).

Let $\Psi = (N, E, \diamond)$ be an eCRG, then:

- $\nu_x^t \mathrel{\bar{\in}} \mathfrak{T}$ *corresponds for each $x \in T_S \cup T_E \cup M_R \cup M_S \cup P$ to the variable for either the point in time of the start (end) x of a task node, the point of receiving (sending) a message related to message node x, the time related to a point in time node x, or the particular point in time of x iff $x \in P^I$ is a particular point in time instance node.*
- $\nu_x^i \mathrel{\bar{\in}} \mathcal{I}_\mathcal{T} \cup \mathcal{I}_\mathcal{M}$ *corresponds for each $x \in T \cup M_R \cup M_S$ to the the variable for the instance identifier of a task or message node x.*
- $\nu_x^p \mathrel{\bar{\in}} \mathcal{R}_S$ *corresponds for each $x \in T$ to the variable for the performing staff member of a task node x.*

- $\nu_x^r \,\bar{\in}\, \mathcal{R}$ *corresponds for each* $x \in R_S \cup R_G \cup R_R \cup R_U$ *to the variable for the related resource of a resource node* x *or to the particular resource of* x, *iff* $x \in R^I$ *refers to a particular resource instance.*
- $\nu_x^{dc} \,\bar{\in}\, \mathcal{D}$ *corresponds for each* $x \in D$ *to the variable for the associated data container of data container node* x *or to the particular data container of* x, *iff* $x \in D^I$ *refers to particular data container instance.*
- $\nu_x^{do} \,\bar{\in}\, \Omega$ *corresponds for each* $x \in O$ *to the variable for the data object related to a data object node* x *or to the particular data object* x, *iff* $x \in O^I$ *refers to a particular data object instance.*
- $\nu_x^{dv} \,\bar{\in}\, \Omega$ *corresponds for each* $x \in \overrightarrow{wdfo} \cup \overrightarrow{rdfo} \cup \overrightarrow{wdfa} \cup \overrightarrow{rdfa} \cup {}^{\diamond}dcp$ *to the variable for the value of a data flow edge* x *or the value of the parameter of an attached data condition* x.

Furthermore, we define the following sets:

- $V_t := \{\nu_x^t | x \in T_S \cup T_E \cup M_R \cup M_S \cup P\}$ *corresponds to the set of all variables for points in time,*
- $V_i := \{\nu_x^i | x \in T \cup M_R \cup M_S\}$ *corresponds to the set of all variables for instance identifiers,*
- $V_p := \{\nu_x^p | x \in R_S\}$ *corresponds to the set of all variables for performing staff members,*
- $V_r := \{\nu_x^r | x \in R_S \cup R_G \cup R_R \cup R_U\}$ *corresponds to the set of all variables for resources,*
- $V_{dc} := \{\nu_x^{dc} | x \in D\}$ *corresponds to the set of all variables for data containers,*
- $V_{do} := \{\nu_x^{do} | x \in O\}$ *corresponds to the set of all variables for data objects,*
- $V_{dv} := \{\nu_x^{dv} | x \in \overrightarrow{wdfo} \cup \overrightarrow{rdfo} \cup \overrightarrow{wdfa} \cup \overrightarrow{rdfa} \cup {}^{\diamond}dcp\}$ *corresponds to the set of all variables for data flow edges and parameters, and*
- $V := V_t \cup V_i \cup V_p \cup V_r \cup V_{dc} \cup V_{do} \cup V_{dv}$ *corresponds to the set of all variables.*

Finally, for each set $M \subset N \cup E \cup \diamond - N^I$, *each* $y \in \{t, ip, r, dc, do, dv\}$, *and each* $x \in N \cup E \cup \diamond$ *we define:*

- $V_y^M := \{\nu_x^y | \nu_x^y \in V_y \ \wedge \ x \in M\}$,
- $V^M := V_t^M \cap V_i^M \cap V_p^M \cap V_r^M \cap V_{dc}^M \cap V_{do}^M \cap V_{dv}^M$,
- $V_{TMP}^M := V_t^M \cap V_i^M \cap V_p^M$,
- $V_{RD}^M := V_r^M \cap V_{dc}^M \cap V_{do}^M \cap V_{dv}^M$, *and*
- $V_x^M := \{\nu_x^y | \nu_x^y \in V^M\}$.

Based on these variables, Table 2-4 introduce a set of transformation patterns Γ, which map the elements of an eCRG to conditions over predicates. Further, these tables comprise function ϕ, which defines the *affiliation* of each element; i.e. the superordinated element.

For each pattern class $c \in \mathcal{C}$ (except I) and each node $n \in N$, Def. 9 summarizes the conditions, which are derived by applying the transformation patterns Γ to the corresponding elements. Finally, the resulting terms are used to specify the transformation in Def. 10.

Definition 9 (Pattern Conditions).

Let $\Psi = (N, E, \diamond)$ *be an eCRG, let* $c \in \mathcal{C} - \{I\}$ *be a pattern class, and let* $n \in N$ *be a node. Then*

- $\xi^c := \bigwedge_{x \in c} \Gamma(x)$ *corresponds to the conjunction of all conditions of pattern* c,
- $\zeta_n^c := \bigwedge_{\substack{x \in c \\ \phi(x) = n}} \Gamma(x)$ *corresponds to the conjunction of all conditions of pattern* c *for node* n.

Table 2: Transformation Pattern for Nodes

x	$\Gamma(x)$	$\phi(x)$
$\alpha \in T$	$Start(\nu^t_{s(\alpha)}, \nu^i_\alpha, type(\alpha)) \land End(\nu^t_{e(\alpha)}, \nu^i_\alpha, type(\alpha)) \land \nu^t_{s(\alpha)} \leq \nu^t_{e(\alpha)}$	α
$\beta \in M_R$	$Receive(\nu^t_\beta, \nu^i_\beta, type(\beta), bp(\beta))$	β
$\beta \in M_S$	$Send(\nu^t_\beta, \nu^i_\beta, type(\beta), bp(\beta))$	β
$\tau \in P$	`true`	τ
$\gamma \in R$	`true`	γ
$\omega \in D$	`true`	ω
$\delta \in O$	`true`	δ

Table 3: Transformation Pattern for Edges

x	$\Gamma(x)$	$\phi(x)$
$e = (\alpha, \beta, rel_t) \in \overrightarrow{sf}$	$\nu^t_\alpha \leq \nu^t_\beta \land rel_t(\alpha, \beta)$	**if** $\alpha \in N^{AA} \cup N^{CA}$ **then** α **else** β
$e = (\alpha, \beta) \in \overrightarrow{mf}$	$\nu^i_\alpha = \nu^i_\beta$	**if** $\alpha \in N^{AA} \cup N^{CA}$ **then** α **else** β
$e = (\alpha, \beta, con_t) \in \overrightarrow{tde}$	$(\nu^t_{e(\alpha)} \geq \nu^t_{s(\beta)} \land con_t(\lvert\nu^t_{e(\alpha)} - \nu^t_{s(\beta)}\rvert))$ $\lor(\nu^t_{e(\beta)} \geq \nu^t_{s(\alpha)} \land con_t(\lvert\nu^t_{e(\beta)} - \nu^t_{s(\alpha)}\rvert))$ $\lor(\nu^t_{s(\alpha)} < \nu^t_{e(\beta)} \land \nu^t_{s(\beta)} < \nu^t_{e(\alpha)} \land con_t(0))$	**if** $\alpha \in N^{AA} \cup N^{CA}$ **then** α **else** β
$e = (\alpha, p, \omega) \in \overrightarrow{wdfc}$	$Write(\cdot, \nu^i_\alpha, p, \nu^{dv}_e, \nu^{dc}_\omega)$	α
$e = (\omega, \alpha, p) \in \overrightarrow{rdfc}$	$Read(\cdot, \nu^i_\alpha, p, \nu^{dv}_e, \nu^{dc}_\omega)$	α
$e = (\alpha, p, \delta) \in \overrightarrow{wdfo}$	$Write(\cdot, \nu^i_\alpha, p, \nu^{dv}_e, \cdot) \land \nu^{do}_\delta = \nu^{dv}_e$	α
$e = (\delta, \alpha, p) \in \overrightarrow{rdfo}$	$Read(\cdot, \nu^i_\alpha, p, \nu^{dv}_e, \cdot) \land \nu^{do}_\delta = \nu^{dv}_e$	α
$e = (\alpha, \gamma) \in \overrightarrow{pfm}$	$Perform(\nu^i_\alpha, \nu^r_e) \land InStaff(\nu^r_\gamma, \nu^r_e)$	α
$e = (\gamma, \eta, rel_r) \in \overrightarrow{rr}$	$rel_r(\gamma, \eta)$	γ
$e = (\delta, \epsilon, rel_d) \in \overrightarrow{dr}$	$rel_d(\delta, \epsilon)$	δ
$e = (\alpha, \omega, con_d) \in \overrightarrow{dcec}$	$val(\nu^t_\alpha, \omega, \nu^{dv}_e) \land con_d(\nu^{dv}_e)$	α
$e = (\alpha, \omega, \delta) \in \overrightarrow{dceo}$	$val(\nu^t_\alpha, \omega, \nu^{do}_\delta)$	α

Table 4: Transformation Pattern for Attachments

x	$\Gamma(x)$	$\phi(x)$
$a = (\alpha, p, con_d) \in {}^\diamond dcp$	$Parameter(\nu^i_\alpha, p, v^{dv}_a) \land con_d(v^{dv}_a)$	α
$a = (o, con_d) \in {}^\diamond doc$	$con_d(\nu^{do}_o)$	$\phi(o)$
$a = (e, con_d) \in {}^\diamond dfc$	$con_d(v^{dv}_e)$	$\phi(e)$
$a = (e, \delta) \in {}^\diamond dfo$	$v^{dv}_e = \nu^{do}_\delta$	$\phi(e)$
$a = (\alpha, rel_t) \in {}^\diamond td$	$rel_t(\nu^t_{s(\alpha)}, \nu^t_{e(\alpha)})$	α
$a = (\gamma, con_r) \in {}^\diamond rc$	$con_r(\nu^r_\gamma)$	γ

Definition 10 (Transformation).

The semantic of an eCRG $\Psi = (N, E, \diamond)$ is expressed by the following FOL formula $\Lambda(\Psi)$:

$$\begin{aligned}
\Lambda(\Psi) := \forall\Theta(V^{AO}_{TMP}) : &\Bigg(\Big(\exists\Theta(V^{AO}_{RD}) : \xi^{AO}\Big) \\
&\wedge \Big(\bigwedge_{n\in TMP^{AA}} \nexists\Theta(V^{AA}_{n}) : \exists\Theta(V^{AO}_{RD} \cup V^{AA}_{RD}) : \xi^{AO} \wedge \zeta^{AA}_{n}\Big) \Bigg) \\
\Rightarrow \exists\Theta(V^{CO}_{TMP}) : &\Bigg(\Big(\exists\Theta(V^{AO}_{RD} \cup V^{CO}_{RD}) : \xi^{AO} \wedge \xi^{CO} \Big) \\
&\wedge \Big(\bigwedge_{n\in TMP^{CA}} \nexists\Theta(V^{CA}_{n}) : \exists\Theta(V^{AO}_{RD} \cup V^{CO}_{RD} \cup V^{CA}_{RD}) : \xi^{AO} \wedge \xi^{CO} \wedge \zeta^{CA}_{n}\Big)\Bigg)
\end{aligned}$$

Thereby, $\Theta(W)$ is replaced by the comma-separated concatenation of the variables contained by a set $W \subseteq V$. For example '$\exists\Theta(\{\nu_1, \nu_2, \nu_3\})$' would be replaced by '$\exists\nu_1, \nu_2, \nu_3$'. In case of the empty set (i.e., $W = \varnothing$), $\Theta(W)$ is replaced by a dummy variable.

Note that the transformation specified in Def. 3-10 is neither able to deal with exclusive and alternative connectors nor with multiple consequence patterns. Hence, we formally introduce exclusive and alternative connectors as well as multiple consequences in Def. 11.

Definition 11 (eCRG with multiple consequences, exclusive & alternative connectors).

An eCRG with exclusive and alternative connectors is a tuple $\Psi^+ = (N, E, \diamond, \overline{ex}, \overline{alt})$. Thereby $\Psi' := (N, E, \diamond)$ is an eCRG and

- *$\overline{ex} \subseteq 2^{T\cup M_S\cup M_R\cup P}$ is the set of exclusive connectors between task nodes, sending message nodes, receiving message nodes, and points in time and*
- *$\overline{alt} \subseteq 2^{T\cup M_S\cup M_R\cup P}$ is the set of alternative connectors between task nodes, sending message nodes, receiving message nodes, and points in time.*

$T, M_S, M_R, P \subseteq N$ are the distinct sets of task nodes, sending message nodes, receiving message nodes, and points in time of Ψ' (cf. Def. 4). We define

- *$N_{ex} := \bigcup_{\chi\in\overline{ex}} \chi$ as the set of nodes contained in the set of exclusive connectors,*
- *$N_{alt} := \bigcup_{\chi\in\overline{alt}} \chi$ as the set of nodes contained in the set of alternative connectors, and*
- *$N_{ex,alt} := N_{ex} \cup N_{alt}$ as the union of these two sets.*

As opposed to Def. 7,

- *SC is the set of pairwise disjoint consequence patterns, and*
- *$C_i \in SC$ with $C_i \subseteq N \cup E \cup \diamond$ corresponds to a particular consequence pattern and CO_i (CA_i) is the corresponding consequence occurrence pattern (consequence absence pattern).*

Exclusive and alternative connectors either connect elements of the antecedence pattern solely or elements of the same consequence pattern. Consequently, we can partition the set of exclusive and alternative connectors as well as the related sets of nodes (N_{ex}, N_{alt}, $N_{ex,alt}$) into antecedence and consequence connectors. We use exponents ^{A} and ^{C_i} to indicate the corresponding subset.
An exclusive connector is satisfied if the condition related to exactly one of its nodes (and connected edges and attachments) is satisfied. In turn, the conditions related to the other nodes of the exclusive connector must be violated. Thus, to

solve an exclusive connector we can split it into different cases, whereby each case corresponds to the selection of one node. In this context, the selected node remains unchanged, while all other nodes of the exclusive connector are negated (i.e. occurrence nodes become absence nodes and vice versa). Finally, the logic disjunction of all cases expresses the semantics of the exclusive connector. An alternative connector is satisfied, if the condition related to at least one of its nodes (and connected edges and attachments) is satisfied. In turn, the conditions of the remaining nodes are irrelevant. According to exclusive connectors, we can solve an alternative connector by splitting it into its different cases, whereby each case corresponds to the selection of one node. In this context, the selected node remains unchanged, whereas the remaining nodes of the alternative connector are removed. Again, the logic disjunction of all cases expresses the semantics of the alternative connector. In Def. 12 formally introduce *selections of nodes* first, which correspond to the cases mentioned above.

Definition 12 (Selections).

Let $\Psi^+ = (N, E, \diamond, \overline{ex}, \overline{alt})$ be an eCRG with multiple consequences, exclusive and alternative connectors. Then

- $\tau \subseteq N^A$ *is an antecedence selection choosing one node out of each exclusive and alternative antecedence connector; i.e.,* $\forall \chi \in \overline{ex}^A \cup \overline{alt}^A : |\tau \cap \chi| = 1$,
- $\tau^{-1} := N^A_{ex,alt} - \tau$ *is the complement of* τ,
- $\upsilon \subseteq N^{C_i}$ *is a consequence selection choosing one node out of each exclusive and alternative consequence connector of* C_i*; i.e.,* $\forall \chi \in \overline{ex}^{C_i} \cup \overline{alt}^{C_i} : |\tau \cap \chi| = 1$,
- $\upsilon^{-1} := N^{C_i}_{ex,alt} - \upsilon$ *is the complement of* υ,
- $\Pi^A := \{\tau \subseteq N_{ex,alt} \cap (N^{AO} \cup N^{AA}) | \forall \chi \in \overline{ex}^A \cup \overline{alt}^A : |\tau \cap \chi| = 1\}$ *is the set of all antecedence selections, and*
- $\Pi^{C_i} := \{\upsilon \subseteq N_{ex,alt} \cap (N^{CO_i} \cup N^{CA_i}) | \forall \chi \in \overline{ex}^{C_i} \cup \overline{alt}^{C_i} : |\upsilon \cap \chi| = 1\}$ *is the set of all consequence selections of consequence pattern* C_i.

As aforementioned, each selection configures the pattern class of nodes; i.e. selected nodes remain in their class, while the non-selected ones change their pattern class (exclusive connector) or are removed (alternative connector). Def. 13 specifies these *node configurations* in detail.

Definition 13 (Configuration of Nodes).

Let $\Psi^+ = (N, E, \diamond, \overline{ex}, \overline{alt})$ be an eCRG with multiple consequences, exclusive and alternative connectors, and let $\tau \in \Pi^A$ be an antecedence selection and $\upsilon \in \Pi^{C_i}$ be a consequence selection. Then

- $N^{(AO,\tau)} := (N^{AO} - N_{ex,alt}) \cup (\tau \cap N^{AO}_{ex,alt}) \cup (\tau^{-1} \cap N^{AA}_{ex})$ *corresponds to the configuration of antecedence occurrence nodes* N^{AO} *based on* τ,
- $N^{(AA,\tau)} := (N^{AA} - N_{ex,alt}) \cup (\tau \cap N^{AA}_{ex,alt}) \cup (\tau^{-1} \cap N^{AO}_{ex})$ *corresponds to the configuration of antecedence absence nodes* N^{AA} *based on* τ,
- $N^{(CO_i,\upsilon)} := (N^{CO_i} - N_{ex,alt}) \cup (\upsilon \cap N^{CO_i}_{ex,alt}) \cup (\upsilon^{-1} \cap N^{CA_i}_{ex})$ *corresponds to the configuration of antecedence occurrence nodes* N^{CO_i} *based on* τ,
- $N^{(CA_i,\upsilon)} := (N^{CA_i} - N_{ex,alt}) \cup (\upsilon \cap N^{CA_i}_{ex,alt}) \cup (\upsilon^{-1} \cap N^{CO_i}_{ex})$ *corresponds to the configuration of antecedence absence nodes* N^{CA_i} *based on* τ,

Node configurations affect the corresponding edges and attachments as well. Thus, Def. 14 (re-)configures the definition pattern classes from Def. 7:

Definition 14 (Configuration of Classes).

Let $\Psi^+ = (N, E, \diamond, \overline{ex}, \overline{alt})$ be an eCRG with exclusive and alternative connectors, and let $\tau \in \Pi^A$ be an antecedence selection and $\upsilon \in \Pi^{C_i}$ be a consequence selection. Then:

- $E^{(AO,\tau)} := \{e \in E^A | nd(e) \subseteq N^{(AO,\tau)}\}$ *is the configuration of antecedence occurrence edges based on* τ,
- $E^{(AA,\tau)} := \{e \in E^A | nd(e) \cap N^{(AA,\tau)} \neq \varnothing \ \wedge \ nd(e) \cap N^{(AO,\tau)} \neq \varnothing\}$ *is the configuration of antecedence absence edges based on* τ,
- $E^{(CO_i,\tau,\upsilon)} := \{e \in E^{C_i} | nd(e) \subseteq (N^{(CO_i,\upsilon)} \cup N^{(AO,\tau)})\}$ *is the configuration of consequence occurrence edges based on* τ *and* υ,
- $E^{(CA_i,\tau,\upsilon)} := \{e \in E^{C_i} | nd(e) \cap N^{(CA_i,\upsilon)} \neq \varnothing \ \wedge \ nd(e) \cap (N^{(CO_i,\upsilon)} \cup N^{(AO,\tau)}) \neq \varnothing\}$ *is the configuration of consequence absence edges based on* τ *and* υ,

- $\diamond^{(AO,\tau)} := \{a \in \diamond^A | at(a) \in N^{(AO,\tau)} \cup E^{(AO,\tau)}\}$ *is the configuration of antecedence occurrence attachments based on* τ,
- $\diamond^{(AA,\tau)} := \{a \in \diamond^A | at(a) \in N^{(AA,\tau)} \cup E^{(AA,\tau)}\}$ *is the configuration of antecedence absence attachments based on* τ,
- $\diamond^{(CO_i,\tau,\upsilon)} := \{a \in \diamond^{C_i} | at(a) \in N^{(AO,\tau)} \cup E^{(AO,\tau)} \cup N^{(CO_i,\upsilon)} \cup E^{(CO_i,\tau,\upsilon)}\}$ *is the configuration of consequence occurrence attachments based on* τ *and* υ,
- $\diamond^{(CA_i,\tau,\upsilon)} := \{a \in \diamond^{C_i} | at(a) \in N^{(CA_i,\upsilon)} \cup E^{(CA_i,\tau,\upsilon)}\}$ *is the configuration of consequence absence attachments based on* τ *and* υ,

- $AO^\tau := N^{(AO,\tau)} \cup E^{(AO,\tau)} \cup \diamond^{(AO,\tau)}$ *is the configuration of the antecedence occurrence pattern based on* τ,
- $AA^\tau := N^{(AA,\tau)} \cup E^{(AA,\tau)} \cup \diamond^{(AA,\tau)}$ *is the configuration of the antecedence absence pattern based on* τ,
- $CO_i^{(\tau,\upsilon)} := N^{(CO_i,\upsilon)} \cup E^{(CO_i,\tau,\upsilon)} \cup \diamond^{(CO_i,\tau,\upsilon)}$ *is the configuration of the consequence occurrence pattern based on* τ *and* υ,
- $CA_i^{(\tau,\upsilon)} := N^{(CA_i,\upsilon)} \cup E^{(CA_i,\tau,\upsilon)} \cup \diamond^{(CA_i,\tau,\upsilon)}$ *is the configuration of the consequence absence pattern based on* τ *and* υ,

- $A^\tau := AO^\tau \cup AA^\tau$ *is the configuration of the antecedence pattern based on* τ, *and*
- $C_i^{(\tau,\upsilon)} = CO_i^{(\tau,\upsilon)} \cup CA_i^{(\tau,\upsilon)}$ *is the configuration of the consequence pattern based on* τ *and* υ,

- $HA^\tau := A - A^\tau$ *are the elements of the antecedence pattern that are masked during its configuration based on* τ,
- $HC_i^{(\tau,\upsilon)} := C_i - C_i^{(\tau,\upsilon)}$ *are the elements of consequence pattern* C_i *that are masked during its configuration based on* τ *and* υ.

Based on Def. 14, we are able to enrich the transformation from Def. 10 with the support for exclusive and alternative connectors as well as multiple consequences in Def. 15.

Definition 15 (Transformation).

The semantics of an eCRG $\Psi^+ = (N, E, \Diamond, \overline{ex}, \overline{alt})$ with multiple consequences, exclusive and alternative connectors is expressed by the following FOL formula $\Lambda^+(\Psi^+)$:

$$
\begin{aligned}
\Lambda^+(\Psi^+) := \bigvee_{\tau \in \Pi^A} \forall \Theta(V_{TMP}^{AO^\tau}) : \Bigg(\bigg(\Big(& \exists \Theta(V_{RD}^{AO^\tau}) : \xi^{AO^\tau} \Big) \\
\wedge \Big(\bigwedge_{n \in TMP^{AA^\tau}} & \nexists \Theta(V_n^{AA^\tau}) : \exists \Theta(V_{RD}^{AO^\tau} \cup V_{RD}^{AA^\tau}) : \xi^{AO^\tau} \wedge \zeta_n^{AA^\tau} \Big) \bigg) \\
\Rightarrow \bigvee_{\substack{C_i \in SC \\ \upsilon \in \Pi^{C_i}}} \exists \Theta(V_{TMP}^{CO_i^{\tau,\upsilon}}) : \bigg(\Big(& \exists \Theta(V_{RD}^{AO^\tau} \cup V_{RD}^{CO_i^{\tau,\upsilon}}) : \xi^{AO^\tau} \wedge \xi^{CO_i^{\tau,\upsilon}} \Big) \\
\wedge \Big(\bigwedge_{n \in TMP^{CA_i^{\tau,\upsilon}}} & \nexists \Theta(V_n^{CA_i^{\tau,\upsilon}}) : \exists \Theta(V_{RD}^{AO^\tau} \cup V_{RD}^{CO_i^{\tau,\upsilon}} \cup V_{RD}^{CA_i^{\tau,\upsilon}}) : \xi^{AO^\tau} \wedge \xi^{CO_i^{\tau,\upsilon}} \wedge \zeta_n^{CA_i^{\tau,\upsilon}} \Big) \bigg) \Bigg)
\end{aligned}
$$

For example, Def. 10 and Def. 15 translate eCRG c_4 from Fig. 10A into:

$$
\begin{aligned}
\Lambda(c_4) = \forall \nu_{P_S}^t, \nu_{P_E}^t, \nu_P^i, \nu_P^p : \bigg(\Big(& \exists \nu_O^{do}, \nu_{(O,P)}^{dv} : \\
& Start(\nu_{P_S}^t, \nu_P^i, \text{Prod.}) \wedge End(\nu_{P_E}^t, \nu_P^i, \text{Prod.}) \wedge \nu_{P_S}^t \leq \nu_{P_E}^t \\
& \wedge Read(\cdot, \nu_P^i, \cdot, \nu_{(O,P)}^{dv}, \cdot) \wedge \nu_O^{do} = \nu_{(O,P)}^{dv} \Big) \\
& \Rightarrow \Big(\nexists \nu_{C_S}^t, \nu_{C_E}^t, \nu_C^i, \nu_C^p : \exists \nu_O^{do}, \nu_{(O,P)}^{dv}, \nu_{(O,C)}^{dv} : \\
& \qquad Start(\nu_{P_S}^t, \nu_P^i, \text{Prod.}) \wedge End(\nu_{P_E}^t, \nu_P^i, \text{Prod.}) \wedge \nu_{P_S}^t \leq \nu_{P_E}^t \\
& \qquad \wedge Read(\cdot, \nu_P^i, \cdot, \nu_{(O,P)}^{dv}, \cdot) \wedge \nu_O^{do} = \nu_{(O,P)}^{dv} \\
& \qquad \wedge Start(\nu_{C_S}^t, \nu_C^i, \text{Chg.}) \wedge End(\nu_{C_E}^t, \nu_C^i, \text{Chg.}) \wedge \nu_{C_S}^t \leq \nu_{C_E}^t \\
& \qquad \wedge \nu_{P_S}^t \leq \nu_{C_S}^t \\
& \qquad \wedge Read(\cdot, \nu_C^i, \cdot, \nu_{(O,C)}^{dv}, \cdot) \wedge \nu_O^{do} = \nu_{(O,C)}^{dv} \Big) \bigg)
\end{aligned}
$$

5 Pattern-based Evaluation

This section outlines a pattern-based evaluation of the eCRG language. In particular, we model the compliance patterns introduced in [9, 11, 29] with the eCRG language. These patterns result from literature and case studies, and thus constitute a suitable empirical basis for evaluating the appropriateness of our approach.

5.1 Business Process Control Pattern and Property Specification Pattern

27 *business process control patterns* (BPCPs) for modeling compliance rules are introduced in [11]. They include all property specification patterns from [54]. Out of the 27 BPCPs, 15 BPCPs focus on the control flow (i.e., process) perspective, 7 BPCPs focus on the resource perspective, and 5 BPCPs focus on the time perspective. In turn, the data perspective is implicitly supported by each BPCP, since BPCPs do not distinguish between tasks and data conditions. Fig. 11 shows how to model the 15 control flow BPCPs with the eCRG language. In turn, Fig. 12 provides eCRGs that model the 7 resource BPCPs and the 5 time BPCPs.

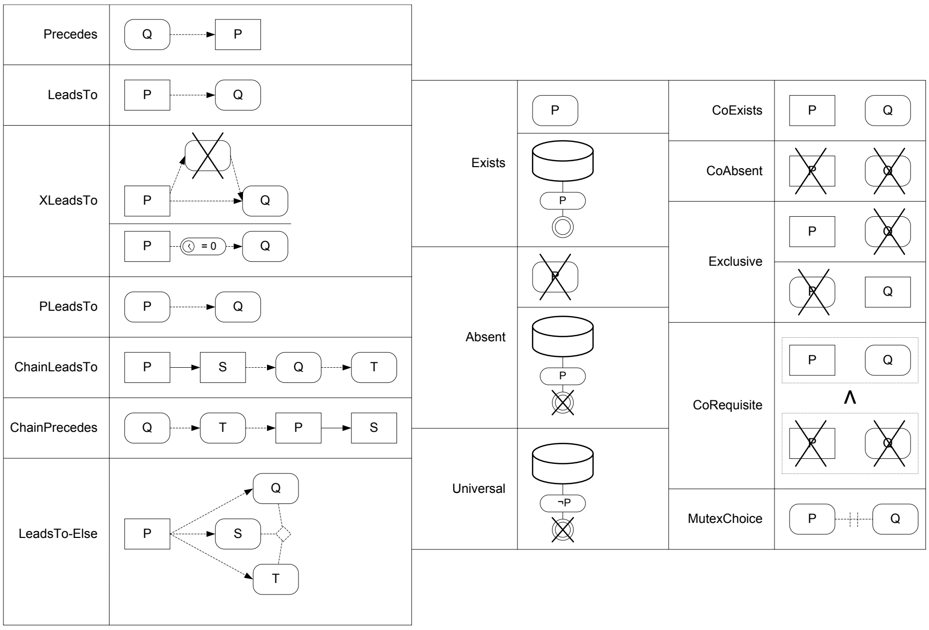


Fig. 11: control flow BPCPs

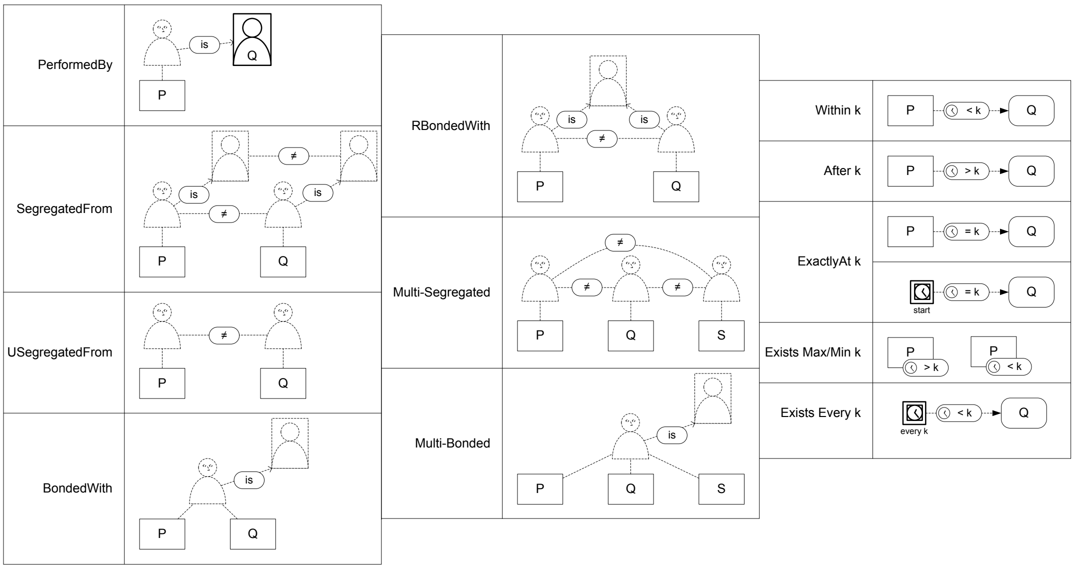


Fig. 12: Resource and time BPCPs

Note that most BPCPs can easily be modeled using the eCRG language. However, from our subjective point of view the modeling of the *XLeadsTo* BPCP, the *Universal* BPCP, the *Exists Max/Min* BPCP, and the *Exists Every k* BPCP was a bit more complicated. In turn, Fig. 12 provides an eCRG for the special case of the *Multi-Segregated* BPCP, which requires the numbers of performers and tasks to be equal. Other cases of the Multi-Segregated BPCP can not be modeled using only one consequence pattern; e.g., in case that the Multi-Segregated BPCP requires 4 tasks to be executed by 3 different performers (i.e., one performer must execute two tasks), $\binom{4}{2} = 6$ consequence patterns are required. Hence the Multi-Segregated BPCP is the only BPCP that cannot always be appropriately modeled using the eCRG language.

5.2 Compliance Rule Pattern

55 control flow *compliance rule patterns* (CRPs) are introduced in [9]. Further, one example of the data perspective and another one of the resource perspective are presented. The 55 control flow CRPs are partitioned into 15 categories. Note that 7 categories and the respective CRPs fully coincide with the aforementioned BPCPs (cf. Table 5). The remaining 8 categories and respective CRPs are modeled with the eCRG language in Figs. 13-16. Fig. 17 contains both examples illustrating the data and resource perspectives.

Table 5: Intersection between BPCPs and CRPs

Compliance rule pattern	**Business process control pattern**
Existence - Event universality	Exists
Existence - Event absence	Absent
Exclusive	Exclusive
Mutual Exclusive	MutexChoice
Prequisite	CoAbsent
Inclusive	CoExists
Substitute	LeadsTo-Else
CoRequisite	CoRequisite

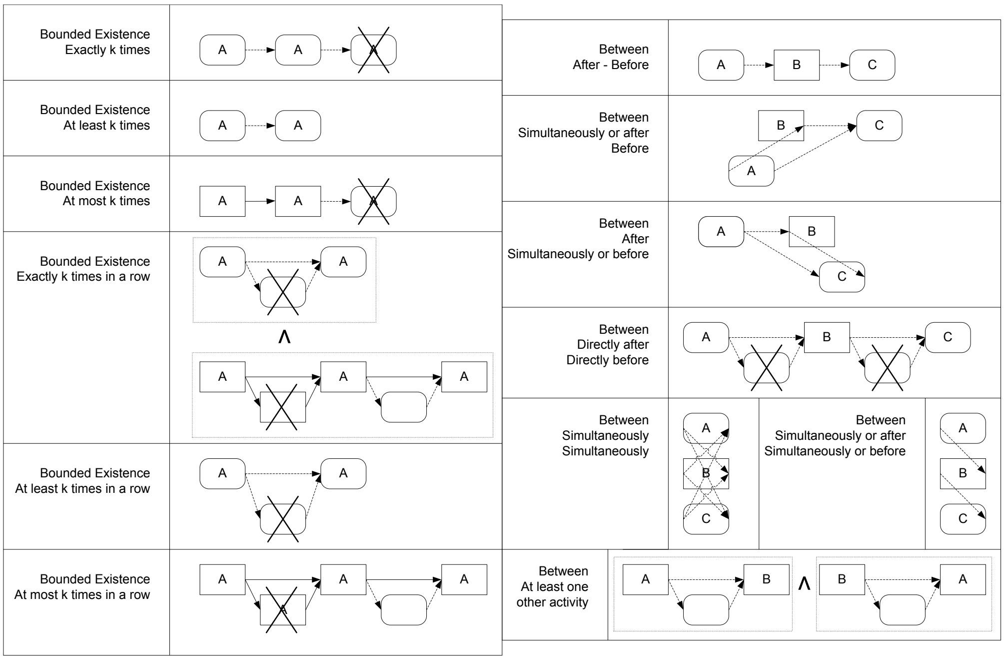


Fig. 13: Bounded existence CRPs and between CRPs

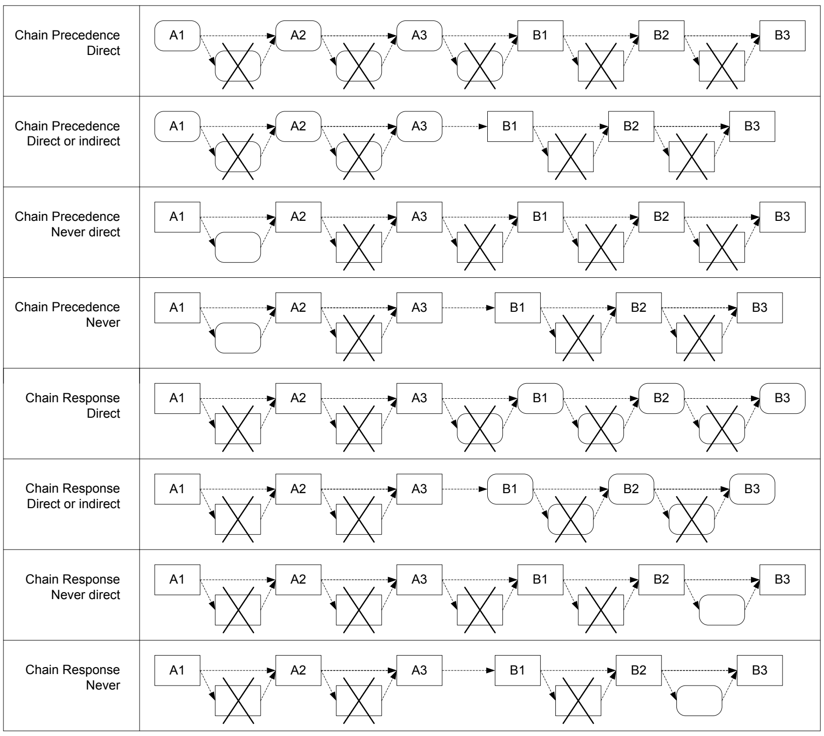


Fig. 14: Chain precedence CRPs and chain response CRPs

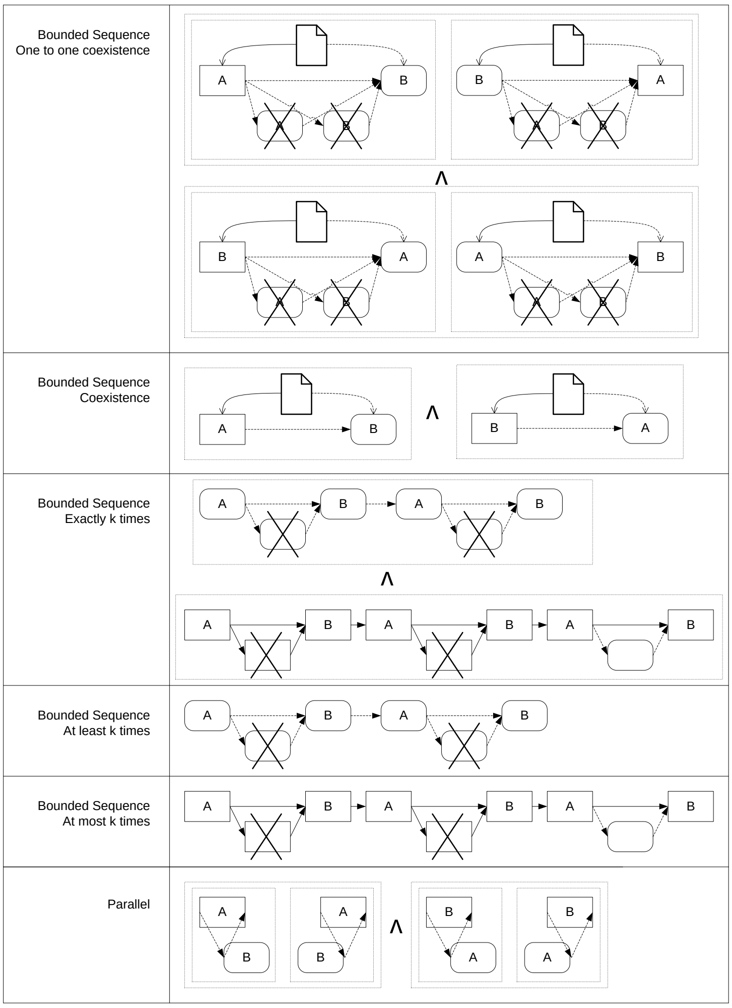


Fig. 15: Bounded sequence CRPs and parallel CRPs

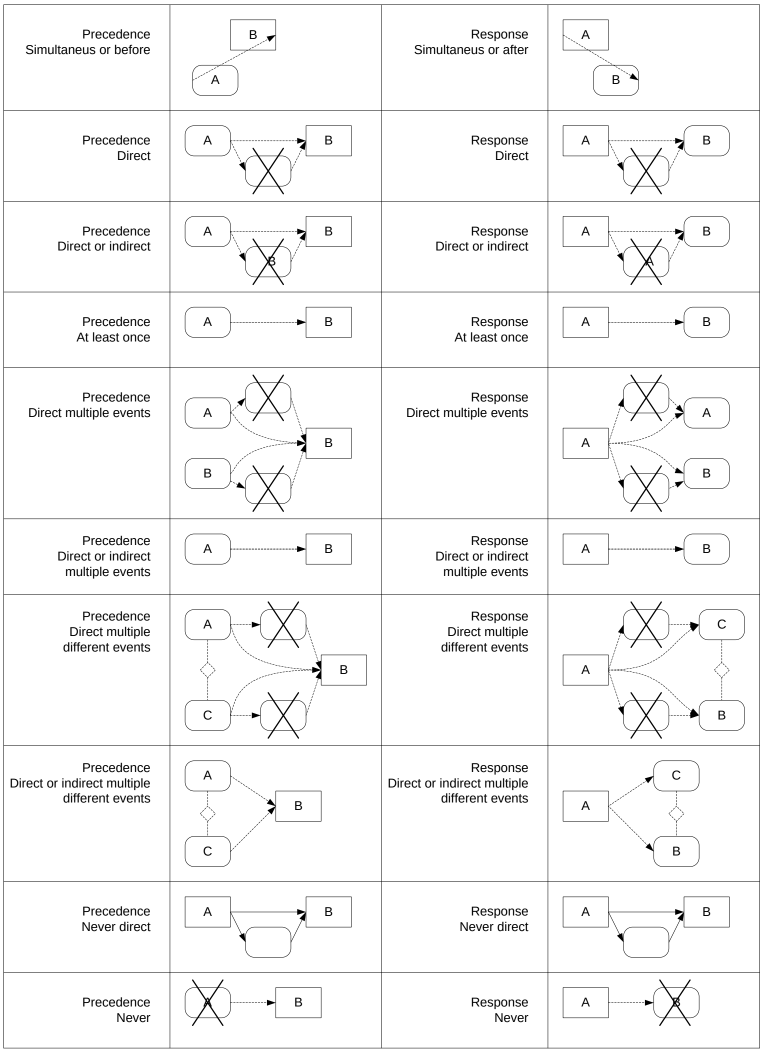


Fig. 16: Precedence CRPs and response CRPs

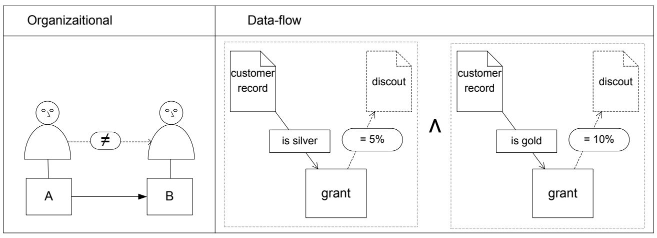


Fig. 17: Data flow CRPs and resource CRPs

Note that most CRPs can easily be modeled using the eCRG language. However, from our subjective point of view a thinking outside the box was required when we modeled the *Bounded Existence - Exactly k times in a row* CRP, the *Bounded Existence - At most k times in a row* CRP, the *Between - Simultaneously, Simultaneously* CRP, and all *Bounded Sequence* CRPs - especially in case of the *Bounded Sequence - One to one coexistence* CRP.

5.3 Time Pattern

In [28, 29], a set of 10 *time patterns* (TPs) is introduced and formally specified. In Fig. 18, the eCRG language is applied to model these TPs.

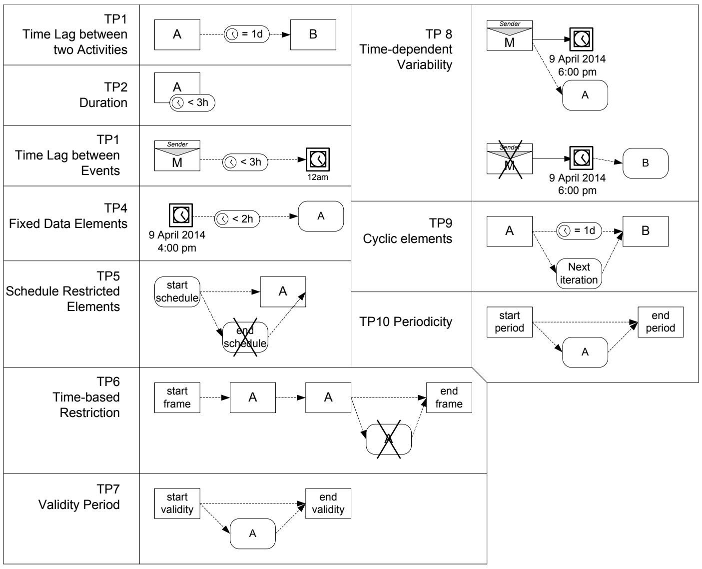


Fig. 18: Time Patterns

Note that most TPs can be easily modeled using the eCRG language. However, from our subjective point of view the modeling of *TP6 Time-based Restrictions* and the modeling of *TP8 Time-dependent Variability* were a bit more complicated.

Fig. 19 summarizes the results of the pattern-based evaluation. Overall, we were able to model 26 out of the 27 business process control patterns (BPCPs) [11], including 5 time BPCPs and 7 resource BPCPs as well. Only the multi-segregation pattern cannot be properly modeled using the eCRG language in any case. Further, the eCRG language supports the 15 categories of compliance rule patterns and respective rules as well as the data flow and the resource example from [9]. Finally, the eCRG language covers well-known time-patterns [29] as well.

Business process control patterns Property specification patterns			
Precedes	++	USegregatedFrom	++
LeadsTo	++	BondedWith	++
XLeadsTo	+	RBondedWith	++
PLeadsTo	++	Multi-Segregated	-
ChainLeadsTo	++	Multi-Bonded	++
Chain Precedes	++	Within k	++
LeadsTo - Else	++	After k	++
Exists	++	ExactlyAt k	++
Absent	++	Exists Max/Min	+
Universal	+	Exists Every k	+
CoExists	++		
CoAbsent	++		
Exclusive	++		
CoRequisite	++		
MutexChoice	++		
PerformedBy	++		
SegregatedFrom	++		

Compliance rule pattern categories	
Existence	++
Bounded existence	+
Bounded sequence	+
Parallel	++
Precedence	++
Chain precedence	++
Response	++
Chain response	++
Between	+
Exclusive	++
Mutual exclusive	++
Inclusive	++
Prequisite	++
Substitute	++
Corequisite	++
Data flow	++
Organizational	++

Time patterns	
Time lags between activities	++
Durations	++
Time lags between events	++
Fixed date elements	++
Schedule restricted elements	++
Time-based restrictions	+
Validity period	++
Time-dependend variability	+
Cyclic elements	++
Periodicity	++

++ full support, + inconvenient support, 0 partial support, - minor support, -- no support

Fig. 19: Support of compliance patterns

6 Summary and Outlook

While compliance rule modeling has been introduced by a plethora of approaches, the data, time, resource and interaction perspectives of compliance rules have not been sufficiently addressed yet [12, 8–11]. This report has introduced the *extended Compliance Rule Graph (eCRG)* language, which is based on the *compliance rule graph (CRG) language* [15, 24]. As opposed to existing visual compliance rule notations, the eCRG supports multiple perspectives; i.e., beyond the control flow perspective, data, time, resources, and interactions with business partners are considered. To provide tool support for both the modeling and verification of compliance rules, we have formalized the syntax and semantics of the eCRG. Finally, we have conducted a pattern-based evaluation to prove the proper expressiveness of the eCRG language.
In future work, we will conduct experiments to evaluate the usability and scalability of the eCRG. Furthermore, we will develop techniques for verifying compliance of business processes and process choreographies with such rules.

References

1. Knuplesch, D., Reichert, M., Ly, L.T., Kumar, A., Rinderle-Ma, S.: Visual modeling of business process compliance rules with the support of multiple perspectives. In: ER'2013. Volume 8217 of LNCS., Springer (2013) 106–120

2. Reichert, M., Weber, B.: Enabling Flexibility in Process-Aware Information Systems - Challenges, Methods, Technologies. Springer (2012)
3. van der Aalst, W.M.P.: Verification of workflow nets. In: ICATPN'97. Volume 1248 of LNCS., Springer (1997) 407–426
4. Reichert, M.: Dynamische ablaufänderungen in workflow-management-systemen, university of ulm, germany (2000)
5. Rinderle, S., Reichert, M., Dadam, P.: Evaluation of correctness criteria for dynamic workflow changes. In: BPM '03. Volume 2678 of LNCS., Springer (2003) 41–57
6. Sadiq, S., Governatori, G., Naimiri, K.: Modeling control objectives for business process compliance. In: BPM'07. Volume 4717 of LNCS., Springer (2007) 149–164
7. Lenz, R., Reichert, M.: It support for healthcare processes - premises, challenges, perspectives. Data and Knowledge Engineering **61**(1) (2007) 39–58
8. Cabanillas, C., Resinas, M., Ruiz-Cortés, A.: Hints on how to face business process compliance. In: JISBD'10. (2010) 26–32
9. Ramezani, E., Fahland, D., van der Aalst, W.M.P.: Where did I misbehave? Diagnostic information in compliance checking. In: BPM'12. Volume 7481 of LNCS., Springer (2012) 262–278
10. Mangler, J., Rinderle-Ma, S.: IUPC: Identification and unification of process constraints. arXiv.org (2011)
11. Turetken, O., Elgammal, A., van den Heuvel, W.J., Papazoglou, M.: Capturing compliance requirements: A pattern-based approach. IEEE Software **29**(3) (2012) 29–36
12. Knuplesch, D., Reichert, M., Mangler, J., Rinderle-Ma, S., Fdhila, W.: Towards compliance of cross-organizational processes and their changes. In: BPM'12 Workshops. Volume 132 of LNBIP., Springer (2013) 649–661
13. Fdhila, W., Rinderle-Ma, S., Reichert, M.: Change propagation in collaborative processes scenarios. In: CollaborateCom'12, IEEE (2012) 452–461
14. Ghose, A.K., Koliadis, G.: Auditing business process compliance. In: ICSOC'07. Volume 4749 of LNCS., Springer (2007) 169–180
15. Ly, L.T., Rinderle-Ma, S., Dadam, P.: Design and verification of instantiable compliance rule graphs in process-aware information systems. In: CAiSE'10. Volume 6051 of LNCS., Springer (2010) 9–23
16. Liu, Y., Müller, S., Xu, K.: A static compliance-checking framework for business process models. IBM Systems Journal **46**(2) (2007) 335–261
17. Ly, L., Indiono, C., Mangler, J., Rinderle-Ma, S.: Data transformation and semantic log purging for process mining. In: CAiSE'12. Volume 7328 of LNCS., Springer (2012) 238–253
18. Schultheiß, B., Meyer, J., Mangold, R., Zemmler, T., Reichert, M., Dadam, P., Kreienberg, R.: Prozessentwurf am Beispiel eines Ablaufs aus dem OP-Bereich - Ergebnisse einer Analyse an der Universitätsfrauenklinik Ulm. Technical Report DBIS-6, University of Ulm (1996)
19. Schultheiß, B., Meyer, J., Mangold, R., Zemmler, T., Reichert, M., Dadam, P., Kreienberg, R.: Prozessentwurf für den Ablauf einer ambulanten Chemotherapie. Technical Report DBIS-7, University of Ulm (1996)
20. Konyen, I., Schultheiß, B., Reichert, M.: Prozessentwurf für den Ablauf einer radiologischen Untersuchung. Technical Report DBIS-15, University of Ulm (1996)
21. Konyen, I., Schultheiß, B., Reichert, M.: Prozessentwurf eines Ablaufs im Labor. Technical Report DBIS-16, University of Ulm (1996)

22. Bestfleisch, U., Herbst, J., Reichert, M.: Requirements for the workflow-based support of release management processes in the automotive sector. In: ECEC'05. (2005) 130–134
23. Müller, D., Herbst, J., Hammori, M., Reichert, M.: It support for release management processes in the automotive industry. In: BPM'06. Volume 4102 of LNCS., Springer (September 2006) 368–377
24. Knuplesch, D., Reichert, M.: Ensuring business process compliance along the process life cycle. Technical Report 2011-06, Ulm University (2011)
25. Russell, N., van der Aalst, W.M.P., ter Hofstede, A.H.M., Edmond, D.: Workflow resource patterns: Identification, representation and tool support. In: CAiSE'05. Volume 3520 of LNCS., Springer (2005) 216–232
26. Kumar, A., Wang, J.: A framework for document-driven workflow systems. In: Handbook on Business Process Management 1. Volume 3649 of LNCS. Springer (2010) 419–440
27. Eder, J., Tahamtan, A.: Temporal conformance of federated choreographies. In: DEXA'08. Volume 5181 of LNCS., Springer (2008) 668–675
28. Lanz, A., Weber, B., Reichert, M.: Workflow time patterns for process-aware information systems. In: BPMDS'10. Volume 50 of LNBIP., Springer (2010) 94–107
29. Lanz, A., Weber, B., Reichert, M.: Time patterns for process-aware information systems. Requirements Engineering **19**(2) (2014) 113–141
30. Decker, G., Weske, M.: Interaction-centric modeling of process choreographies. Information Systems **36**(2) (2011) 292–312
31. Barros, A., Dumas, M., ter Hofstede, A.: Service interaction patterns. In: BPM'05. Volume 3649 of LNCS., Springer (2005) 302–318
32. Künzle, V., Weber, B., Reichert, M.: Object-aware business processes: Fundamental requirements and their support in existing approaches. Journal of Information System Modeling and Design **2**(2) (2011) 19–46
33. Knuplesch, D., Pryss, R., Reichert, M.: Data-aware interaction in distributed and collaborative workflows: Modeling, semantics, correctness. In: CollaborateCom'12, IEEE (2012) 223–232
34. Ly, L.T., Rinderle, S., Dadam, P.: Integration and verification of semantic constraints in adaptive process management systems. Data & Knowledge Engineering **64**(1) (2008) 3–23
35. Ly, L.T., Rinderle-Ma, S., Göser, K., Dadam, P.: On enabling integrated process compliance with semantic constraints in process management systems. Information Systems Frontiers **14**(2) (2012) 195–219
36. Ramezani, E., Fahland, D., van der Werf, J.M., Mattheis, P.: Separating compliance management and business process management. In: BPM'11 Workshops. Volume 100 of LNBIP., Springer (2012) 459–464
37. Governatori, G., Sadiq, S.: The journey to business process compliance. In: Handbook of Research on BPM. IGI Global (2009) 426–454
38. Namiri, K., Stojanovic, N.: Pattern-Based design and validation of business process compliance. In: OTM'07. Volume 4803 of LNCS., Springer (2007) 59–76
39. Governatori, G., Hoffmann, J., Sadiq, S., Weber, I.: Detecting regulatory compliance for business process models through semantic annotations. In: BPM'08 Workshops. Volume 17 of LNBIP., Springer (2009) 5–17
40. Kumar, A., Yao, W., Chu, C.: Flexible process compliance with semantic constraints using mixed-integer programming. INFORMS Journal on Computing **25**(3) (2013) 543–559

41. Awad, A., Weidlich, M., Weske, M.: Specification, verification and explanation of violation for data aware compliance rules. In: ICSOC'09. Volume 5900 of LNCS., Springer (2009) 500–515
42. Knuplesch, D., Ly, L.T., Rinderle-Ma, S., Pfeifer, H., Dadam, P.: On enabling data-aware compliance checking of business process models. In: ER'2010. Volume 6412 of LNCS., Springer (2010) 332–346
43. Kokash, N., Krause, C., de Vink, E.: Time and data aware analysis of graphical service models. In: SEFM'10, IEEE (2010) 125–134
44. Ly, L.T., Knuplesch, D., Rinderle-Ma, S., Göser, K., Pfeifer, H., Reichert, M., Dadam, P.: Seaflows toolset - compliance verification made easy for process-aware information systems. In: Information Systems Evolution. Volume 72 of LNBIP. Springer (2011) 76–91
45. Knuplesch, D., Reichert, M., Fdhila, W., Rinderle-Ma, S.: On enabling compliance of cross-organizational business processes. In: BPM'13. Volume 8094 of LNCS. Springer (2013) 146–154
46. Knuplesch, D., Reichert, M., Pryss, R., Fdhila, W., Rinderle-Ma, S.: Ensuring compliance of distributed and collaborative workflows. In: CollaborateCom'13, IEEE (2013) 133–142
47. Höhn, S.: Model-based reasoning on the achievement of business goals. In: SAC'09, ACM (2009) 1589–1593
48. Accorsi, R., Lowis, L., Sato, Y.: Automated certification for compliant cloud-based business processes. Business & Information Systems Engineering **3**(3) (2011) 145–154
49. Awad, A., Weidlich, M., Weske, M.: Visually specifying compliance rules and exp-laining their violations for business processes. Journal of Visual Languages & Computing **22**(1) (2011) 30–55
50. Feja, S., Speck, A., Witt, S., Schulz, M.: Checkable graphical business process representation. In: ADBIS'11. Volume 6295 of LNCS., Springer (2011) 176–189
51. Rinderle, S., Wombacher, A., Reichert, M.: Evolution of process choreographies in DYCHOR. In: OTM'06. Volume 4275 of LNCS., Springer (2006) 273–290
52. Cabanillas, C., Resinas, M., Cortés, A.R.: Defining and analysing resource assignments in business processes with RAL. In: ICSOC'11. Volume 7084 of LNCS., Springer (2011) 477–486
53. Reichert, M.: Process and data: Two sides of the same coin? In: OTM'12. Volume 7565 of LNCS., Springer (2012) 2–19
54. Dwyer, M.B., Avrunin, G.S., Corbett, J.C.: Property specification patterns for finite-state verification. In: FMSP'98, ACM (1998) 7–15