Software & Systems Modeling manuscript No.
(will be inserted by the editor)

A Visual Language for Modeling Multiple Perspectives of Business Process Compliance Rules

David Knuplesch and Manfred Reichert

Received: 8 October 2014 / Revised: 21 December 2015 / Accepted: 3 March 2016

Abstract A fundamental challenge for enterprises is to ensure compliance of their business processes with imposed compliance rules stemming from various sources, e.g., corporate guidelines, best practices, standards, and laws. In general, a compliance rule may refer to multiple process perspectives including control flow, time, data, resources, and interactions with business partners. On one hand, compliance rules should be comprehensible for domain experts who must define, verify and apply them. On the other, these rules should have a precise semantics to avoid ambiguities and enable their automated processing. Providing a visual language is advantageous in this context as it allows hiding formal details and offering an intuitive way of modeling the compliance rules. However, existing visual languages for compliance rule modeling have focused on the control flow perspective so far, but lack proper support for the other process perspectives. To remedy this drawback, this paper introduces the extended Compliance Rule Graph language, which enables the visual modeling of compliance rules with the support of multiple perspectives. Overall, this language will foster the modeling and verification of compliance rules in practice.

Keywords business process compliance, extended compliance rule graphs, business process modeling, smart processes

1 Introduction

During the last decades a variety of techniques for verifying the correctness of business process models were proposed. While early approaches focused on

David Knuplesch and Manfred Reichert
Ulm University,
Institute of Database and Information Systems
James-Franck-Ring
89081 Ulm, Germany
E-mail: firstname.familyname@uni-ulm.de

issues related to structural and behavioral model correctness (e.g., absence of deadlocks and livelocks) [1,84], the semantic correctness of process models with respect to imposed compliance rules (i.e., business process compliance) has been subject to recent works [34,65,10,45]. Compliance rules constrain the execution order (i.e. control flow) of tasks and may originate, for example, from security constraints, domain-specific guidelines, corporate standards, and legal regulations. Besides the control flow perspective, other fundamental perspectives[1] relevant in the context of business process compliance refer to time, data, and resources as well as the interactions a business process has with partner processes [20,81,51].

1.1 Problem Statement

In practice, compliance rules are represented in a rather verbose and ambiguous way. To enable the computer-based verification of business process compliance, i.e., to verify that a particular business process meets imposed compliance rules, subject matter experts and business analysts should provide unambiguous descriptions of compliance rules, which then can be translated into a machine-readable representation by IT experts. For the latter purpose, several approaches for the formal specification of compliance rules exist, e.g. applying linear temporal logics (LTL) [30] or using the formal contract language (FCL) [33]. As formal rule languages would be too intricate for subject matter experts and business analysts, rule patterns hiding formal details and providing informal explanations were suggested [24,97,80,82]. Although few approaches exist that not only consider the control flow perspective, but also the data, time and resource perspectives, these approaches only support a pre-specified set of rule patterns.

As shown by empirical studies, business process modeling as well as compliance rule description languages, which both employ visual notations, offer advantages compared to purely text-based specifications [77,38]: First, visual notations significantly increase model and rule comprehensibility after providing some training to users. Second, they foster the communication among business analysts and subject matter experts on one hand and process engineers on the other. As a prequisite for the computerized support of visual specifications, the latter should be machine-readable, relying on a precise formal semantics.

Examples of visual notations for compliance rules include Compliance Rule Graphs [66], BPMN-Q [9], and BPSL [62]. Like visual process modeling languages (e.g., YAWL [3], ADEPT [85] and BPMN [75])[2], theses approaches

[1] In this paper, the notion of *process perspective* corresponds to a specific business process modeling dimension according to [17].

[2] Note that imperative process modeling approaches tend to (over-)specify business processes. Hence, they are not well suited for specifying declarative constraints and process compliance rules [79,35,87].

combine an intuitive notation with the advantages of a formal language. Existing visual compliance rule languages, however, lack a comprehensive support of the time, data, resource, and interaction perspectives of a business processes, which hinders their use in more sophisticated scenarios.

1.2 Contribution

Although there exist pattern-based approaches for modeling compliance rules also covering the time, data, and resource perspectives, a respective visual modeling support has not been provided so far [80,97]. To remedy this drawback, this paper provides an approach for the visual modeling of compliance rules that may refer to these perspectives as well as to the interactions a business process may have with partner processes. In particular, the paper shows how the various perspectives can be visually represented with the *extended Compliance Rule Graph* (eCRG) language. We evaluate the expressiveness of the latter based on well-known patterns and apply the eCRG language to a real-world healthcare scenario. Furthermore, understandability issues are considered in an empirical study. Finally, we present a proof-of-concept prototype that supports the modeling of eCRGs as well as their verification. The latter is based on a profound formal semantics of the eCRGs, which is provided in a technical report [49]. Altogether, the eCRG language allows domain experts to capture compliance requirements at both an abstract and a visual level, while enabling the specification of verifiable compliance rules that consider the various perspectives.

This paper significantly extends our previous work, which introduced fundamentals of the eCRG language [50,94]: In [50], we provided a very brief overview of the eCRG language, whereas in [94] we focused on the resource perspective solely. In addition to these preliminary works, this paper provides

- the first detailed presentation of the eCRG elements covering the interaction, time, data, and resource perspectives,
- an empirical study on the comprehensibility of the eCRG language
- a proof-of-concept prototype, which comprises a modeling environment as well as an eCRG compliance checker verifying the compliance of given process execution logs with a set of eCRGs, and
- an extended and more profound discussion of related work.

The remainder of the paper is organized as follows: Section 2 provides an application scenario from the healthcare domain, which we will use as illustrating example throughout the paper. In Section 3, we introduce the eCRG language step-by-step. Specifically, this language supports the visual modeling of compliance rules that may refer to the control flow, interaction, time, data, and resource perspectives. To assess the approach, Section 4 provides a pattern-based evaluation and demonstrates the modeling of real-world compliance rules with the eCRG language. Furthermore, it presents an empirical study and the proof-of-concept prototype we developed. Related Work is dis-

cussed in Section 5. Section 6 concludes the paper and provides an outlook on future work.

2 Application Scenario

This section introduces a healthcare scenario dealing with compliance rules and processes captured in the context of a large process engineering project in a university hospital [53,56]. We will refer to this scenario throughout the paper to illustrate how the elements of the eCRG language can be applied.

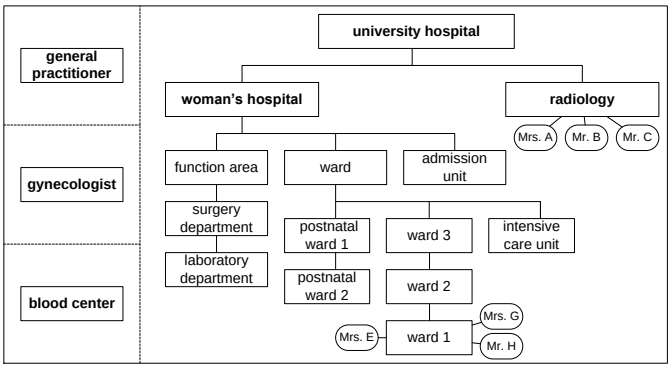


Fig. 1: Organizational units and partners

Selected organizational units and related subunits are depicted in Fig. 1 [53,56]. On the right, two subunits of the *university hospital* are shown; i.e., the *woman's hospital*, together with its wards (e.g. *postnatal ward 1*) and service units (e.g., *admission unit*), and the *radiology department*. On the left, external partners of the hospital are depicted, e.g., a *gynecologist*, a *general practitioner*, and a *blood center*.

The example refers to six actors (i.e., *Mrs. A*, *Mr. B*, *Mr. C*, *Mrs. E*, *Mrs. G*, and *Mr. H*) having different roles and being assigned to different organizational units. *Mrs. A*, *Mr. B* and *Mr. C* are members of the *radiology department*, whereas *Mrs. E*, *Mrs. G* and *Mr. H* are assigned to *ward 1*. *Mr. B* and *Mrs. E* both have role *physician*, whereas *Mrs. A* has role *MTA* (i.e., medical technical assistant); *Mrs. G* and *Mr. H* are nurses and *Mr. C* is the secretary of the *radiology department*.

Table 1 depicts compliance rules C1 – C10, which emphasize the need for covering the control flow, time, data, and resource perspective of business processes as well as the interactions a business process has with partner processes.

All rules from Table 1 consider the control flow perspective. For instance, C1 requires the execution of tasks *order X-ray* and *fill order form* prior to the one of task *X-ray examination*. C2 states that the document *informed consent* must be received and checked prior to the execution of task *X-ray examination*.

Table 1: Examples of healthcare compliance rules based on [56]

C1	For an inpatient, an X-ray examination must be ordered by a *ward physician*. The *same physician* must fill in the respective order form.
C2	An X-ray examination in the *radiology department* must be performed by a *radiologist*. Before, the informed consent of the patient must be received and be checked by a *medical technical assistant (MTA)* of the *radiology department*.
C3	Diagnoses shall be provided by *physicians* only after receiving all X-ray images from the *radiology department*; i.e., *no X-ray image may be received afterwards*.
C4	A patient shall be formally admitted within one week after her referral to the hospital.
C5	At least one day before a surgery takes place, blood bags must be ordered.
C6	Before a *physician* requests an informed consent, she or he (i.e., the *same physician*) must inform the patient about risks.
C7	A period of at least 5 days shall elapse between the administrations of the drugs Aspirin and Marcumar.
C8	For patients older than 75, an additional tolerance test is required prior to the medical examination.
C9	If an additional X-ray examination is ordered to prepare a scheduled surgery, the X-ray must be completed before the surgery.
C10	After 26 October 2013, the duration of a tolerance test must not exceed 30 minutes.

C4 requires that task *refer patient* shall be succeeded by task *admit patient*. Rules C2 and C3 additionally refer to the interaction perspective of a business process (i.e., the sending and receipt of messages). More precisely, C2 requires the receipt of a message including document *informed consent*, whereas rule C3 requires waiting for the receipt of a message containing the X-ray images.

The time perspective is considered by rules C4, C5, C7 and C10. Rule C4 defines a maximum time distance between tasks *refer patient* and *admit patient* (i.e., one week), whereas C5 and C7 specify minimal time distances (see [61, 60] for relevant process time patterns). More precisely, C5 constrains the time minimum distance between tasks *order blood bags* and *surgery*, and C7 the minimum time distance between two different occurrences of task *administer*, i.e., one occurrence with drug *Aspirin* and another one with drug *Marcumar*. In addition, C10 refers to a fixed point in time; i.e., *26 October 2013*.

Rules C2, C7 and C8 constitute examples of compliance rules refering to the data perspective. According to C2, document *informed consent* needs to be first transmitted through a message and then be checked. Rule C7 expresses that drugs *Aspirin* and *Marcumar* are used by task *administer*. C8 requires an additional tolerance test if the value of data object *age* is greater than 75.

Compliance rules C1, C2, C3, and C6 refer to the resource perspective. On one hand, C1 reflects the resource perspective by requiring the assignment of a performer with role *physician* to the respective ward. On the other, C1 requires that both tasks (i.e., *order X-ray* and *fill order form*) are performed by the same person (i.e., binding of duties). Rule C2 requires performers with different roles being assigned to the same organizational unit. By contrast, C3 solely refers to role *physician*. Finally, compliance rule C6 provides another example of a binding of duties constraint, i.e., one and the same *physician* should perform tasks *request informed consent* and *inform patient*.

Fig. 2 provides an example of a possible execution log of a healthcare process that shall comply with the rules from Table 1.

step	date	start	end	activity / message	performer	org unit	data access			
							parameter		data object	container
							type	name		
1	5.2.09	09:20	09:36	examine patient	Mrs. E	wrd	in	patient record	(id=1452)	patient record
2	5.2.09	09:40	09:42	order X-ray	Mrs. E	wrd				
3	5.2.09	09:45	09:46	fill request form	Mrs. E	wrd	ex	examination request tpl.	(id=382)	
							out	X-ray request	(id=12305)	examination request (wrd)
4	5.2.09	09:50	09:55	inform patient	Mrs. E	wrd				
5	5.2.09	09:58	10:03	request informed consent	Mrs. E	wrd	ex	informed consent tpl.	(id=141)	
							out	informed consent	(id=12301)	patient consent (wrd)
6	5.2.09	10:10	10:19	transfer patient	Mrs. G	wrd				
7	5.2.09	10:12		*transmit* request with informed consent	Mr. H	wrd ↓ rad	in	X-ray request	(id=12305)	examination request (wrd)
							in	informed consent	(id=12301)	patient consent (wrd)
							out	X-ray request	(id=12305)	current request (rad)
							out	informed consent	(id=12301)	patient consent (rad)
8	5.2.09	10:20	10:22	admit patient	Mr. C	rad	in	X-ray request	(id=12305)	examination request (rad)
9	5.2.09	10:45	10:46	check informed consent	Mrs. A	rad	in	informed consent	(id=12301)	patient consent (rad)
10	5.2.09	10:50	10:52	prepare patient	Mrs. A	rad	in	X-ray request	(id=12305)	X-ray request (rad)
10	5.2.09	11:05	11:12	perfom X-ray	Mr. B	rad	in	request	(id=12305)	X-ray request (rad)
							out	X-ray image	(id=12346)	X-ray image (rad)
12	5.2.09	11:15	11:15	*transmit* progress message	Mr. C	rad ↓ wrd				
13	5.2.09	11:20	11:33	transfer patient	Mrs. G	wrd				
13	5.2.09	11:33	11:35	admit patient	Mr. G	wrd				
14	5.2.09	11:34	11:39	archive X-ray image	Mr. C	rad	in	X-ray image	(id=12346)	X-ray image (rad)
15	5.2.09	11:45		*transmit* X-ray image	Mr. C	rad ↓ wrd	in	image	(id=12346)	X-ray image (rad)
							out	image	(id=12346)	examination results (wrd)
16	5.2.09	14:10	14:15	make diagnosis	Mrs. E	wrd	in	X-ray image	(id=12346)	examination results (wrd)
							out	diagnosis	(id=12352)	current diagnosis (wrd)
17	5.2.09	14:15	14:20	prescribe therapy	Mrs. E	wrd	in	diagnosis	(id=12352)	current diagnosis (wrd)
							out	therapy	(id=12358)	current therapy (wrd)
18	5.2.09	14:40	14:40	document therapy	Mr H	wrd	in	diagnosis	(id=12352)	current diagnosis (wrd)
							in	therapy	(id=12358)	current therapy (wrd)

Fig. 2: Execution log of a healthcare process

3 The Extended Compliance Rule Graph Language

This section introduces the *extended Compliance Rule Graph* (eCRG) language, which enables the visual modeling of compliance rules considering multiple perspectives, i.e., control flow, interaction, time, data, and resources. We discuss the fundamental language design in Section 3.1, whereas Section 3.2 presents the elements of the eCRG language in detail.

3.1 Language Design

The eCRG language enables the visual modeling of compliance rules in terms of rule graphs. It extends the CRG language introduced in [66,67] by adding elements to explicitly cover the interaction, time, data, and resource perspectives as well. The overall purpose of the eCRG language is not only to provide

a well-defined visual language, but to provide a tool fostering the communication between IT experts on one side and domain experts on the other.

Basically, an extended Compliance Rule Graph (eCRG) comprises *nodes* and *edges* that may be further enriched with *attachments*. Nodes may be used to either express *events* (e.g., the start or completion of a task, the receipt of a message, or the occurrence of a particular point in time) or *entities* (e.g., a data object, staff member, or role). In turn, the edges of an eCRG describe the relations between the nodes. Examples of such relations include the order of events or hierarchical relations between the staff members. Moreover, conditions may be attached to refine event and entity nodes as well as the relations (i.e., edges between them). Examples of respective *attachments* include restrictions on the data flow, the time distance between tasks, and the properties of staff members (e.g., a threshold for data flow values, maximum time distances, or required capabilities). Fig. 3 depicts the various kinds of nodes, edges and attachments supported by the eCRG language.

The specification of an eCRG comprises a pre- and a postcondition. The precondition specifies when the compliance rule shall be applied, whereas the postcondition needs to be met to satisfy the compliance rule. Accordingly, the elements of an eCRG (i.e., its nodes, edges and attachments) are partitioned into an *antecedence pattern* (A), which specifies the precondition, and at least one *consequence pattern* (C) specifying a postcondition. An eCRG may further contain references to *instances* of entities like a particular staff member (e.g. *Mr. Smith*), data container (e.g. *credit points*), or point in time (e.g. *26 October 2013*). Note that certain instance nodes are neither part of the antecedence nor the consequence patterns as they are independent from both the pre- and postcondition of the eCRG.

Since the pre- and postconditions (i.e., the antecedence and consequence patterns) of an eCRG may require both the occurrence and the absence of certain events, the two patterns are further sub-divided into an *occurrence sub-pattern* on the one hand and an *absence sub-pattern* on the other. Note that elements of the *antecedence occurrence* (AO) and the *consequence occurrence* (CO) patterns require the occurrence of events, whereas elements of the *antecedence absence* (AA) and the *consequence absence* (CA) patterns require certain events to not occur. To visually distinguish between the patterns, the eCRG language uses dashed lines and round shapes for the elements of a consequence pattern and, by contrast, solid lines and square shapes for the ones of an antecedence pattern. Thick lines and square shapes are used to represent elements referring to particular instances of entities. Absence nodes are crossed out by an oblique cross in order to to differentiate them from occurrence nodes.

Fig. 4 illustrates the partitioning of an eCRG into its antecedence and consequence patterns as well as corresponding sub-patterns.

Considering this partitioning of an eCRG, an execution log is *compliant* with an eCRG, *iff* for each match of the eCRG antecedence pattern (i.e., satisfaction of the precondition), at least one corresponding match of a consequence pattern of the eCRG can be found (i.e., satisfaction of the postcondition). If the log does not include any match of the antecedence pattern (i.e., violation

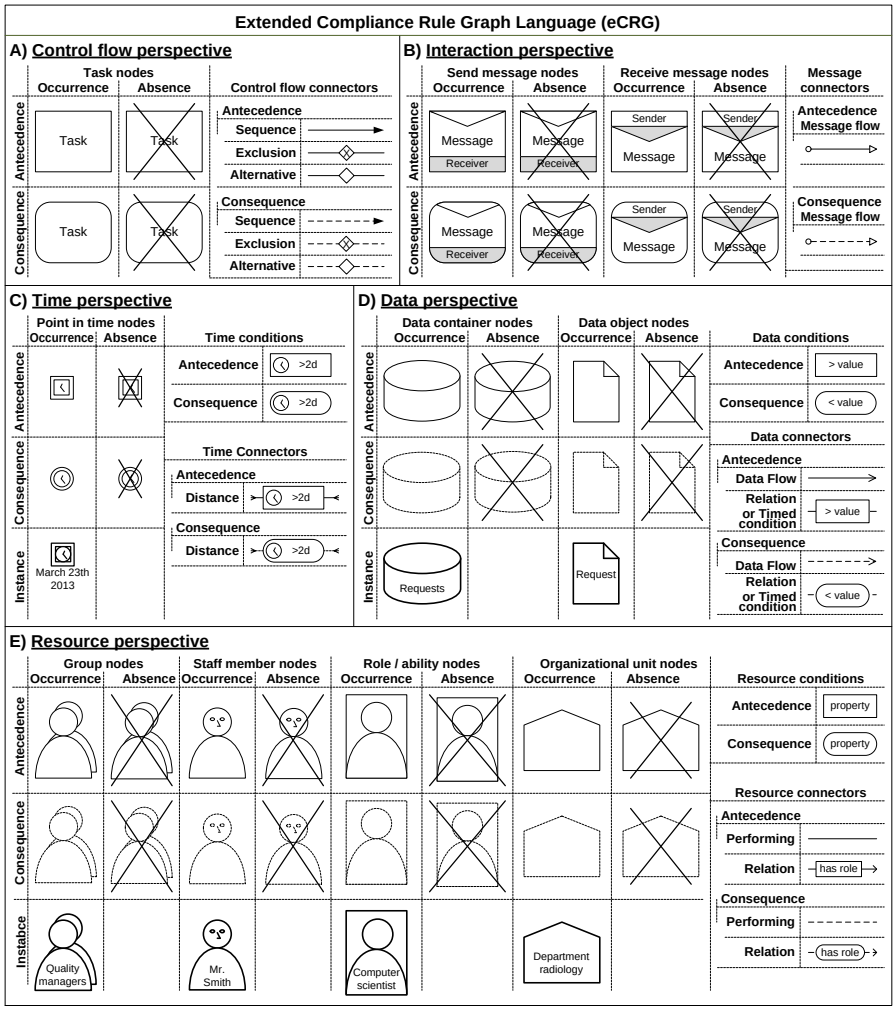


Fig. 3: Nodes, edges and attachments of an eCRG

of the precondition), the process log is *trivial compliant*. If there is a match of the antecedence pattern (i.e., satisfaction of the precondition), for which no corresponding match of a consequence pattern exists (i.e., violation of the postcondition), the execution log *violates* the eCRG. Once compliance violations are detected, proper actions can be triggered in order to compensate the rule violation or to at least ensure its reporting.

A match of a particular antecedence or consequence pattern requires suitable events regarding the nodes of the occurrence sub-pattern. In addition, these events must satisfy all conditions imposed by the other elements of the pattern (i.e., edges, attachments, and entity nodes). For each absence node,

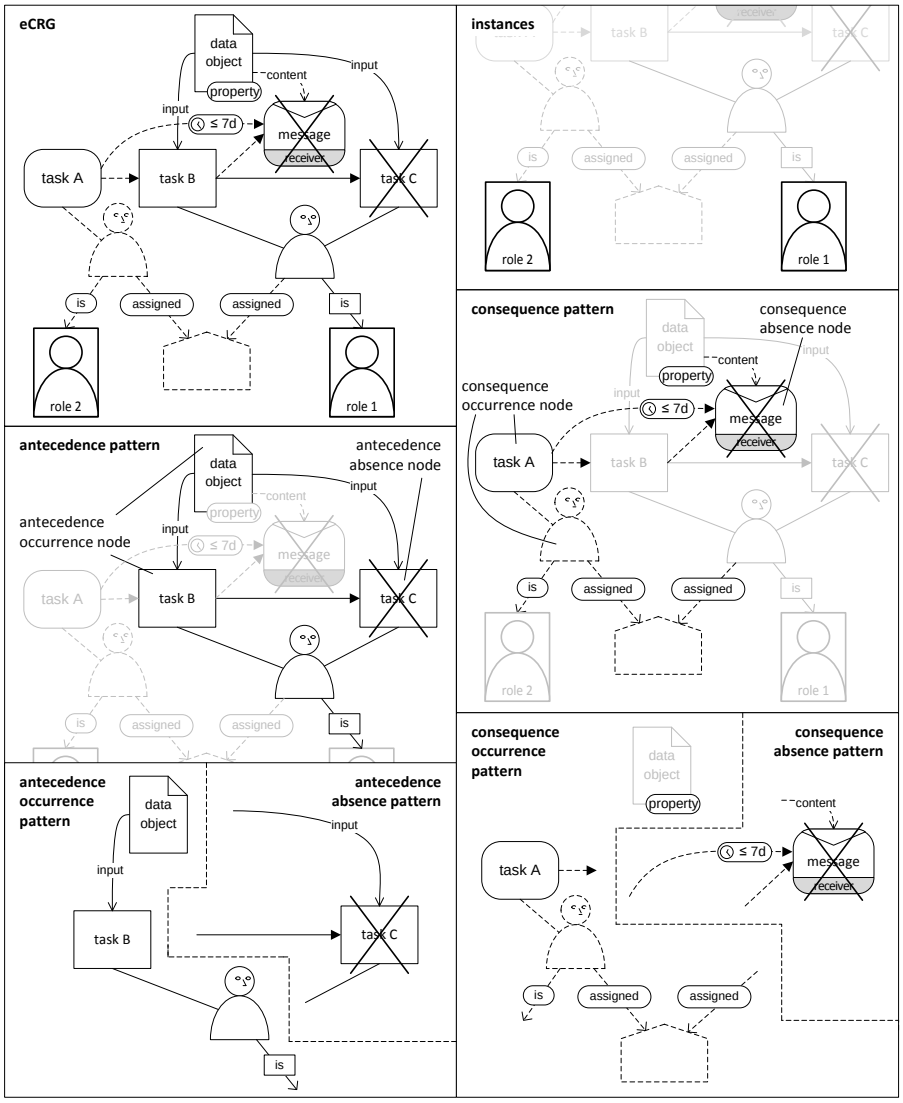


Fig. 4: Components of an eCRG

in turn, the corresponding events must be missing or not meet the conditions imposed by connected edges and attachments.

Note that the design of the eCRG language partially considers the *principles for designing effective visual notations* [73]. Especially, the concepts of semiotic clarity, perceptual discriminability, semantic transparency, graphic economy, and cognitive fit were taken into account.

3.2 eCRG Modeling

This section shows how the different process perspectives can be modeled with the eCRG language. For each perspective, we first introduce the corresponding eCRG elements before illustrating their semantics along simple examples.

3.2.1 Control Flow Perspective

A fundamental process perspective of compliance rules concerns the control flow, which constrains the execution sequence as well as the occurrences of tasks. The elements of the eCRG language referring to the control flow perspective are shown in Fig. 5.

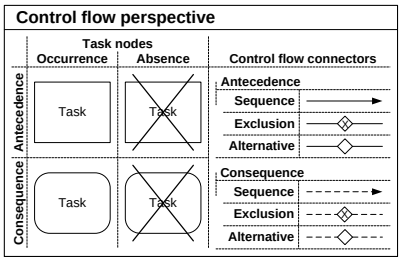


Fig. 5: eCRG elements covering the control flow perspective

Task nodes express whether or not tasks of the given task type shall be executed. In detail, four kinds of *task nodes* are provided, which refer to the *antecedence occurrence*, *antecedence absence*, *consequence occurrence*, and *consequence absence pattern*. Note that the specification of the task type may be omitted, if the latter is not relevant.

In addition to task nodes, *antecedence* and *consequence sequence flow connectors* are provided. These connectors allow constraining the execution sequence of tasks. The absence of sequence flow indicates a parallel flow; i.e., then any possible execution sequence is allowed for the respective tasks. To clearly distinguish between *start-start*, *start-end*, *end-start*, and *end-end* constraints in respect to the execution sequence of tasks, sequence flow connectors are either connected to the right or left border of a task node. *Exclusive connections* express that exactly one of the connected tasks must be executed. In turn, *alternative connections* express that at least one of the connected tasks shall occur. Exclusive as well as alternative connections may be part of both the antecedence and the consequence pattern, but they must solely connect nodes within a particular pattern. Exclusive as well as alternative connections may involve more than two nodes.

Fig. 6 provides basic examples of compliance rules referring to the control flow perspective. The eCRG from Fig. 6a refers to compliance rule C6, whereas the eCRG from Fig. 6b models a variation of C1. Figs. 6c and 6d refer to the control flow perspective of C8 and C9 respectively. All eCRGs from Figs. 6a – 6c use one *antecedence* and one *consequence occurrence task node* as well as a *consequence sequence flow connector* that connects both task nodes restricting their execution order. Fig. 6d shows an *antecedence sequence flow connector* specifying the order of two *antecedence occurrence task nodes*, whereas two *consequence sequence flow connectors* link the *consequence occur-*

rence task node placed between them.

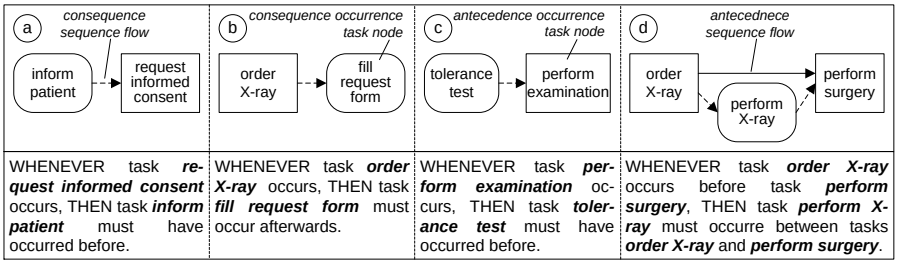


Fig. 6: Basic examples referring to the control flow perspective

More sophisticated examples are depicted in Fig. 7. Fig. 7a uses two *consequence occurrence task nodes* as well as a *consequence sequence flow connector* to provide a more precise specification of compliance rule C1 (cf. Table 1). In order to capture the control flow perspective of C7, Fig. 7b provides two *consequence absence task nodes* linked by a *consequence alternative connection*. In turn, Fig. 7c solely uses an *antecedence occurrence task node* on the left and a *consequence absence task node* on the right side (i.e., no connector is used). Fig. 7d refers to compliance rule C4; note that its *consequence sequence flow connector* does not express an *end-start* constraint (as, for example, the rule in Fig. 7a), but an *end-end* constraint on the execution sequence of tasks.

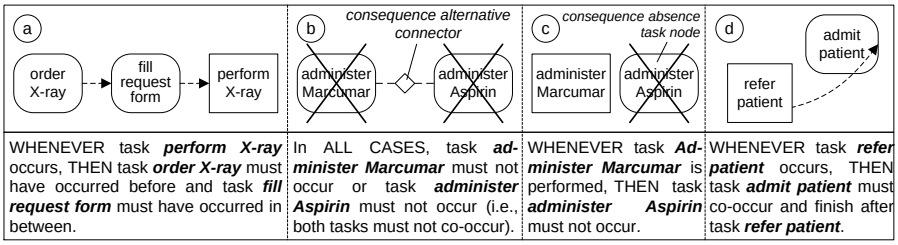


Fig. 7: Advanced examples referring to the control flow perspective

3.2.2 Interaction Perspective

The interaction perspective constrains the interactions a process may have with external partner processes, i.e., the exchange of messages with other organizations and information systems. The eCRG language offers specific nodes for representing the events of *sending* and *receiving* messages. *Sending* and *receiving message nodes* may be part of the *antecedence occurrence*, *antecedence*

absence, *consequence occurrence*, or *consequence absence* patterns. As indicated in Fig. 8, labels on the message node not only allow specifying the message type, but also the message receiver and message sender respectively. Similar to task nodes, the sender, receiver and message type label may be omitted. *Antecedence* and *consequence message flows* connect message nodes referring to the sending and receipt of the same message.

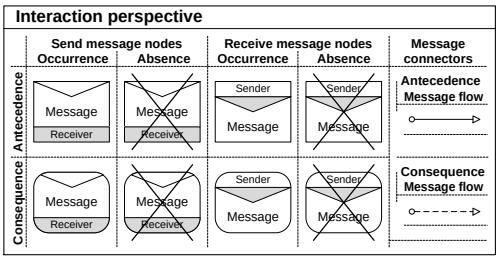
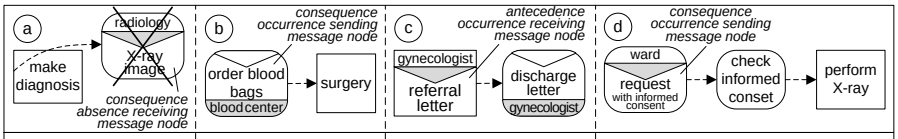


Fig. 8: eCRG elements covering the interaction perspective

Fig. 9 uses the elements from Fig. 8 to model compliance rules referring to the interaction perspective. In particular, Fig. 9a shows a *start-start* constraint that uses the *consequence absence* variant of the *receiving message node* to model the control flow and interaction perspectives of compliance rule C3 (cf. Table 1). Fig. 9b reflects C5 and includes a *consequence occurrence sending message*. The combination of an *antecedence occurrence receiving message* as precondition and a *consequence occurrence sending message* as postcondition is shown in Fig. 9c, whereas Fig. 9d depicts a *consequence occurrence receiving message*.



WHENEVER task ***make diagnosis*** *occurs, THEN* message ***X-ray image*** must not be received after the start of ***make diagnosis***.	WHENEVER task ***surgery*** occurs, THEN message ***order blood bags*** must have been sent to the ***blood center*** before.	WHENEVER a ***referral letter*** is received from the ***gynecologist***, THEN a ***discharge letter*** must be sent to him later.	WHENEVER task ***perform X-ray*** occurs, THEN message ***request*** must have been received from a ***ward*** before. Further, task ***check informed consent*** must have occurred in between.

Fig. 9: Examples referring to the interaction perspective

3.2.3 Time Perspective

When having a closer look on the informal definition of compliance rules C4 and C7 from Table 1, it becomes clear that Figs. 7b – 7d do not fully cover them yet. The time distance between the interactions and tasks of C4 is not considered by the eCRGs from Fig. 7d. The eCRGs from Figs. 7b+c completely disallow for the co-occurrence of the two tasks instead of solely defining the required minimum time distance between them.

The time perspective of the eCRG language provides elements for covering such kinds of constraints (cf. Fig. 10), i.e., for modeling *points in time* and *time conditions*. Similar to task and message nodes, *antecedence occurrence*, *antecedence absence*, *consequence occurrence*, and *consequence absence point-in-time nodes* are supported. Additionally, *instance point-in-time nodes* enable the specification of a concrete point in time (e.g., 26 October 2013). *Antecedence* and *consequence time conditions* may be attached to task nodes as well as sequence flow connectors to either constrain the duration of a task or the time distance between tasks/messages. Finally, *antecedence* and *consequence time distance connectors* may constrain the time distance between tasks/messages without implying a particular order between them.

Fig. 11 depicts examples of eCRGs considering the time perspective as well. Fig. 11a connects two antecedence occurrence task nodes using a *consequence time distance connector*. This eCRG exactly corresponds to compliance rule C7 from Table 1. In turn, Fig. 11b enriches a consequence sequence flow connector with a *time condition* to model compliance rule C4. To visually specify compliance rule C10, Fig. 11c uses an *instance point-in-time node* and attaches a *time duration condition attachment* to an antecedence occurrence task node. The eCRG from Fig. 11d models compliance rule C5.

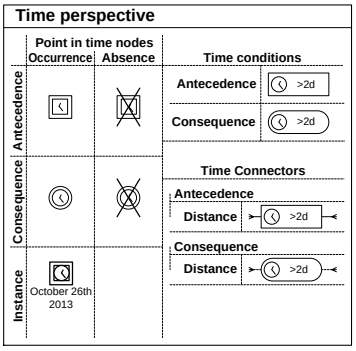


Fig. 10: eCRG elements covering the time perspective

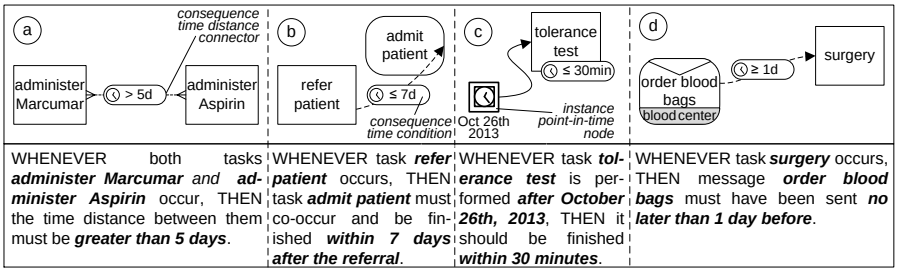


Fig. 11: Examples referring to the time perspective

3.2.4 Data Perspective

Fig. 12 depicts elements for modeling the data perspective of eCRGs: *Data container nodes*, *data object nodes*, *data flow connectors*, and *data conditions* are provided. *Data containers* refer to process data elements or global data

stores, whereas *data objects* refer to specific data values and object instances respectively. Similar to point-in-time nodes, *data container* and *data object nodes* may be part of the *antecedence occurrence*, *antecedence absence*, *consequence occurrence*, or *consequence absence patterns*, whereas *instance data nodes* refer to specific data containers or data objects (e.g., data container *current diagnosis* or data object *X-ray image of Mr. Smith*).

Antecedence and *consequence data flow connectors* define which process tasks read from (write to) which data objects or data containers. To enable the distinction of different data flows, labels may refer to input/output parameters. The caption of a data object/container may specify the parameter name as well. Finally, *antecedence* and *consequence data relation connectors* express relations among data objects.

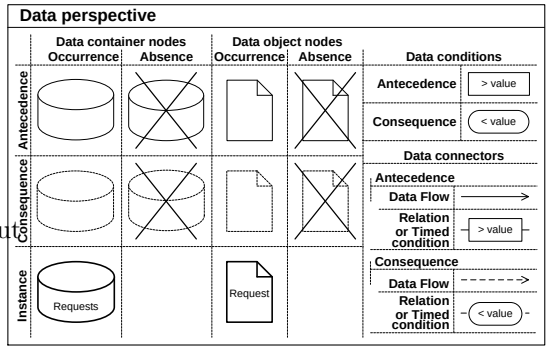


Fig. 12: eCRG elements covering the data perspective

To constrain values of data flow or parameters of tasks/messages, *antecedence* or *consequence data conditions* may be attached to data flow connectors and task/message nodes. *Timed condition connectors* may be added to the *antecedence* or *consequence patterns* to constrain the values of data containers at particular points in time.

Fig. 13 provides examples of eCRGs with *data object nodes*. Fig. 13a depicts an eCRG with a *consequence occurrence data object* and a *consequence data flow connector*. Figs. 13b+c depict a variant of compliance rule C1 (cf. Table 1) and highlight the specification of input/output parameters based on labeled

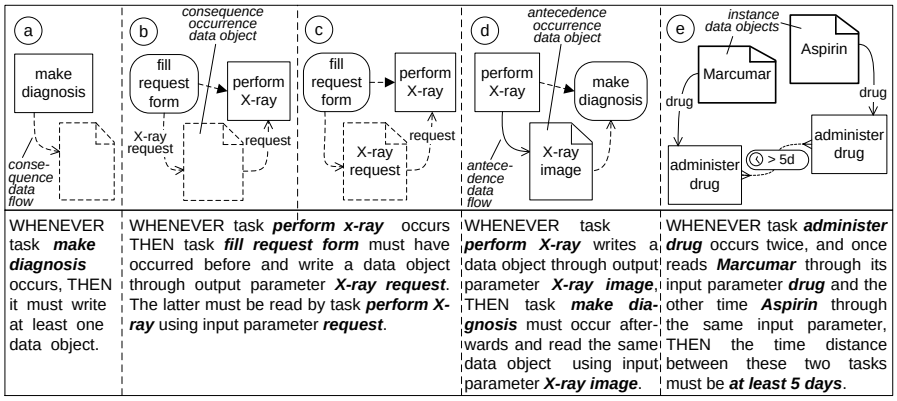
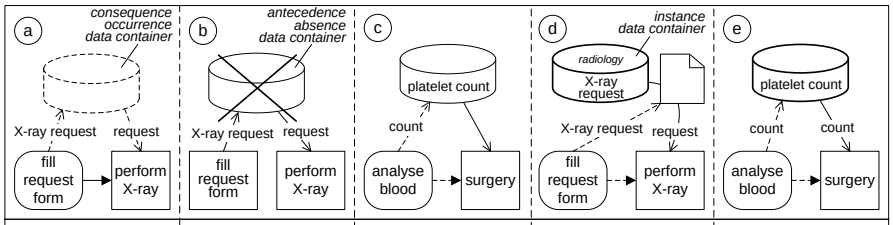


Fig. 13: Examples referring to the data perspective using data objects



WHENEVER task ***perform X-ray*** occurs, THEN task ***fill request form*** must occur before and use output parameter ***X-ray request*** to write into a data container. Further, ***perform X-ray*** must read the same data container using input parameter ***request***.	WHENEVER tasks ***perform X-ray*** and task ***fill request form*** occur and the output parameter ***X-ray request*** of the latter task is ***not*** written to ***the same data container*** from which task ***perform X-ray*** reads through input parameter ***request***, THEN...	WHENEVER task ***surgery*** occurs and reads the ***platelet count*** from a data container, THEN task ***analyse blood*** must have occurred before and stored a value through its output parameter ***count*** in the same data container.	WHENEVER task ***perform X-ray*** occurs and reads from data container ***X-ray request*** through its input parameter ***request***, THEN the data object read must have been written before by task ***fill request form*** via output parameter ***X-ray request***	WHENEVER task ***surgery*** occurs and reads from data container ***platelet count*** through input parameter ***count***, THEN task ***analyse blood*** must have occurred before and written to data container ***platelet count*** via its output parameter ***count.***

Fig. 14: Examples referring to the data perspective using data containers

consequence occurrence data objects and *consequence data flow connectors*. An *antecedence occurrence data object* and the corresponding *antecedence data flow* are shown in Fig. 13d, whereas Fig. 13e uses two *particular instances* of data objects (i.e., Marcumar and Aspirin) to model compliance rule C7.

Fig. 14 illustrates the use of *data container nodes*. The eCRG from Fig. 14a uses a *consequence occurrence data container* and *consequence data flow*. The use of an *antecedence absence data container* is illustrated by Fig. 14b, whereas the eCRG from Fig. 14c comprises an *antecedence occurrence data container*. Figs. 14d+e refer to *instance data container nodes*.

Fig. 15 provides examples illustrating the use of *data condition attachments*. In Fig. 15a, a *consequence data condition* constrains the value of a data flow, whereas the *consequence data condition* in Fig. 15b refers to a data object. An *antecedence timed data condition* of a data container is depicted in Fig. 15c. Fig. 15d provides an *antecedence data condition* constraining the execution parameter of an *antecedence occurrence task node*. Finally, Fig. 15e attaches an *antecedence data condition* to a data object.

3.2.5 Resource Perspective

The resource perspective of the eCRG language covers various kinds of human resources as well as their relations. Furthermore, it allows constraining the assignment of resources to tasks (cf. Fig. 16).

The eCRG language covers various resources, including *staff member*, *role*, *group*, and *organizational unit*, as well as their relations with tasks. Similar to data nodes, *resource nodes* may be part of the *antecedence occurrence*, *antecedence absence*, *consequence occurrence*, and *consequence absence* pattern of an eCRG or refer to *particular instances* of resource entities (e.g., staff member *Mrs. A* or role *physician*). *Performing relation connectors* indicate the performers of tasks. *Resource relation connectors* express relations among

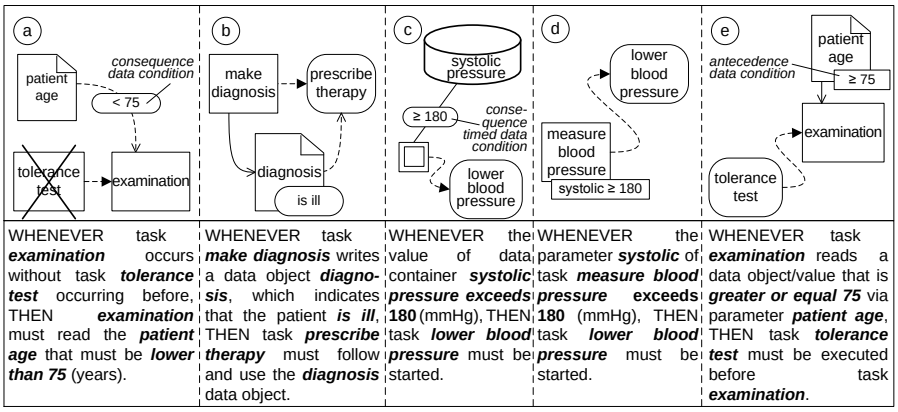


Fig. 15: Examples referring to the data perspective using data conditions

resources. Both connectors are either part of the antecedence or the consequence pattern. *Resource condition attachments* constrain the corresponding resource nodes. – Note that the resource perspective can be easily extended with other kinds of resources if required.

Fig. 17 illustrates the application and semantics of the *performing relation connector*. In Fig. 17a, *antecedence performing relations* are used to connect two antecedence tasks with the same *antecedence staff member*. Fig. 17b depicts a *consequence performing relation* and an *antecedence performing connector*. In Fig. 17c, two *consequence performing relations* connect both antecedence tasks with the same *consequence staff member*. Note that the eCRGs from Figs. 17b+c express the same, as we assume that each task always has exactly one performer. Fig. 17d shows how a consequence task can be connected to an antecedence task by using a *consequence performing relation*, whereas Fig. 17e uses a *consequence absence staff member* that is connected to two antecedence tasks through two *consequence performing relations*.

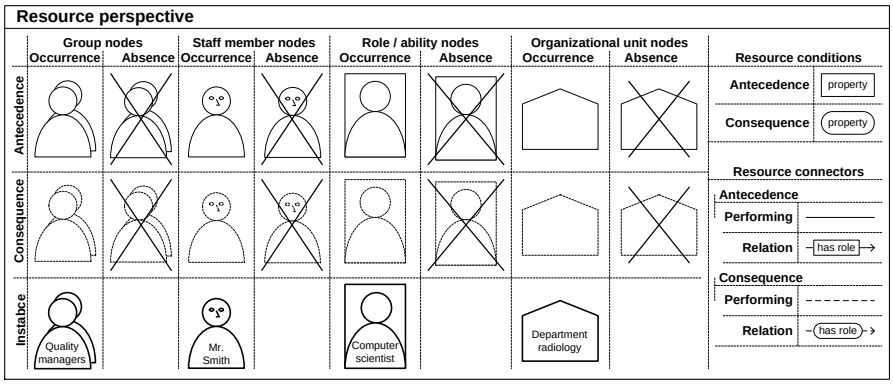


Fig. 16: eCRG elements covering the resource perspective

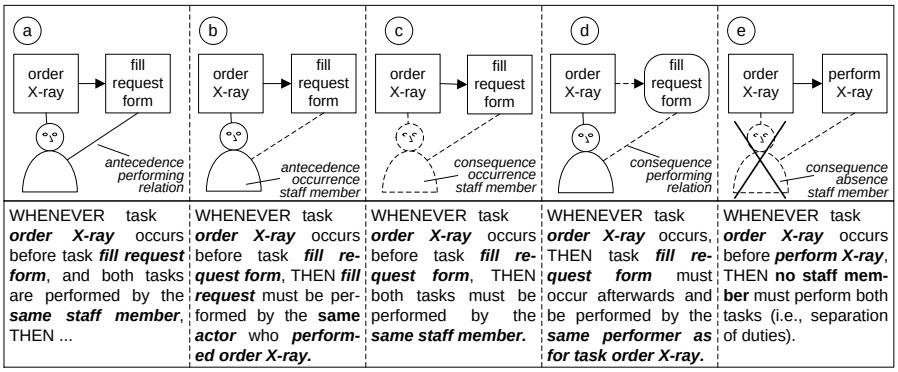


Fig. 17: Examples referring to the resource perspective using performing relations

Fig. 18 illustrates the use and semantics of the *resource relation connector*. Fig. 18a uses *antecedence resource relations* to connect two *antecedence staff members* with an *antecedence absence organizational unit*. Fig. 18b depicts an *antecedence* as well as a *consequence resource relation* connecting two different *antecedence staff members* with the same *antecedence organizational unit*. Fig. 18c comprises a *consequence resource relation* that connects *antecedence* and *consequence staff members*, whereas an *antecedence resource relation* connects the same *antecedence staff member* with the particular *instance role* physician. Fig. 18d applies a *consequence resource relation connector* to refer from the *antecedence staff member* to *instance role* physician. Finally, Figs. 18e+f show how the *performing relation connector* can directly refer to organizational unit and role nodes in order to specify the performers of tasks.

Resource conditions and their semantics are illustrated in Fig. 19. Fig. 19a illustrates the use of an *antecedence resource condition* constraining an *antecedence organizational unit*. Fig. 19b applies a *consequence resource condition* to the same antecedence organizational unit, whereas a *consequence organizational unit* is subject to this condition in Fig. 19c. Despite this difference, Figs. 19b+c express the same. Note that the meaning of Fig. 19d would change when turning the constrained *consequence organizational unit* into an *antecedence consequence organizational unit*.

To enable a formal analysis and verification of eCRGs, we provide formal semantics for the eCRG language as well (for details we refer to a technical report [49]). This semantics provides a transformation of eCRGs to first-order logic (FOL) defining how to interpret and evaluate an eCRG over execution logs.

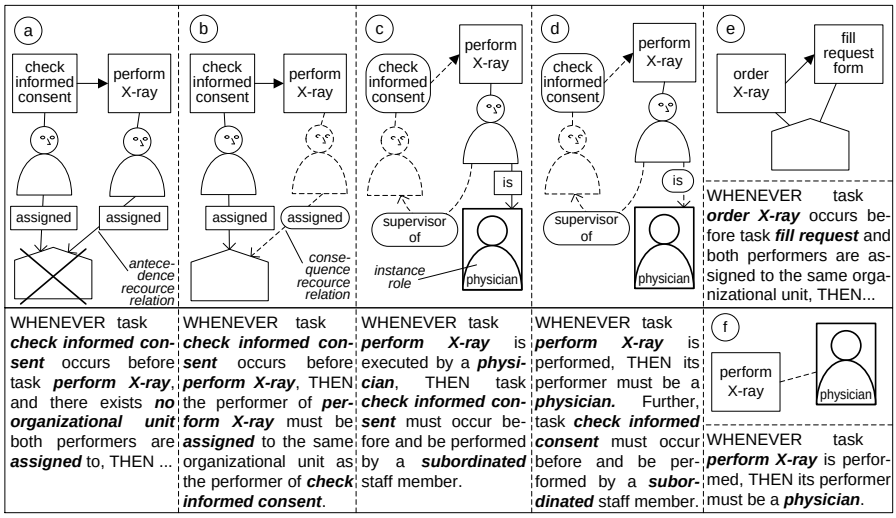


Fig. 18: Examples referring to the resource perspective using resource relations

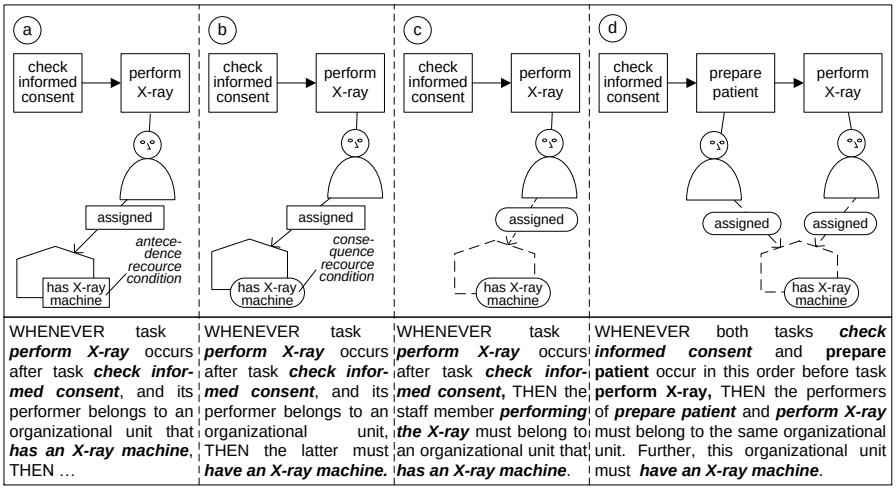


Fig. 19: Examples referring to the resource perspective using resource conditions

4 Evaluation

This section evaluates the eCRG language and illustrates its use. First, Section 4.1 evaluates the expressiveness of the eCRG language based on well-known compliance patterns. Second, Section 4.2 assesses the suitability of the eCRG language with respect to the modeling of compliance rules from the real-world. Third, Section 4.3 evaluates the comprehensibility of the eCRG

language through an empirical study. Finally, Section 4.4 presents a proof-of-concept prototype demonstrating the feasibility of an eCRG-based tool enabling compliance modeling and automated compliance checking.

4.1 Pattern-based Evaluation

This section evaluates the expressiveness of the eCRG language taking existing compliance pattern sets as benchmark. Respective patterns [97, 80, 61] resulted from extensive studies of regulations, standards, frameworks, process collections, and literature. Hence, they constitute suitable references for assessing the expressiveness of compliance rule modeling languages. More precisely, we tried to model the patterns covered by the *business process control patterns* [97], *compliance rule patterns* [80], and *time patterns* [61, 60] using the eCRG language. In addition, we checked for the semantic correspondence between eCRGs and related compliance patterns.

In [97], 27 *business process control patterns* (BPCP) were presented. These also include the property specification patterns [24]. Out of the 27 patterns, 15 focus on the control flow perspective, 7 on the resource perspective, and 5 on the time perspective. The data perspective is covered implicitly by each of the patterns, which do not distinguish between tasks and data conditions. Fig. 20 illustrates how the 15 control flow patterns can be modeled using the eCRG language, and Figs. 21 and 22 show eCRGs covering the 7 resource patterns and the 5 time patterns respectively.

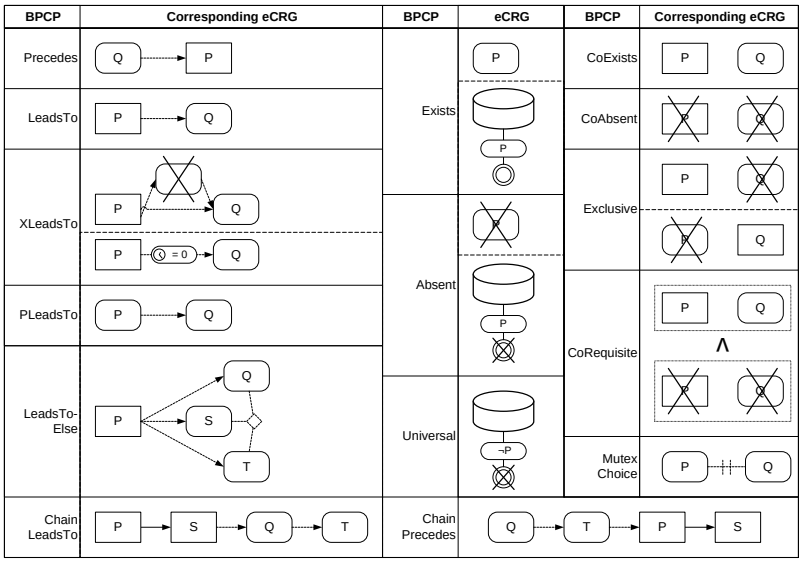


Fig. 20: Modeling control flow BPCPs as eCRGs

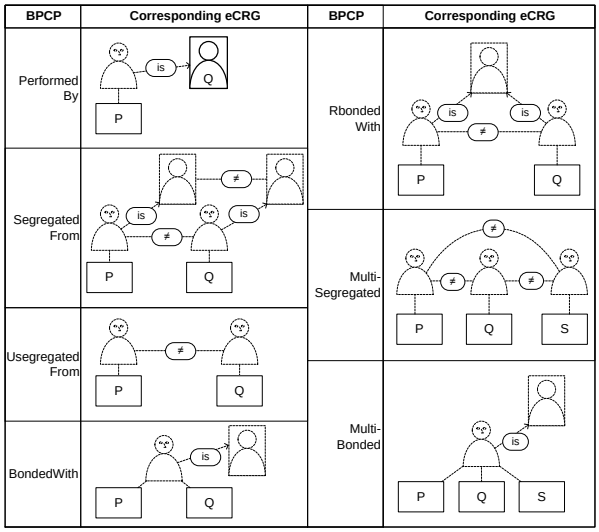


Fig. 21: Modeling resource BPCPs as eCRGs

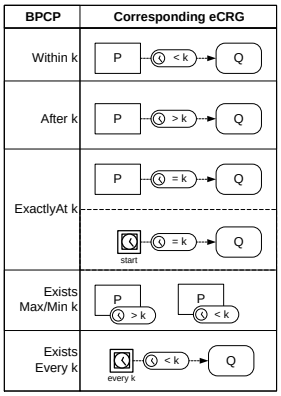


Fig. 22: Modeling time BPCP as eCRGs

Altogether, we are able to model 26 out of the 27 business process control patterns (BPCP) with the eCRG language, including the 5 time patterns and 6 out of the 7 resource patterns. The only pattern for which we cannot provide a generic eCRG is the *Multi-Segregated* BPCP. Note that Fig. 21 only shows the eCRG for a special case of this BPCP, which requires the numbers of performers and tasks to be equal. Other variants of the Multi-Segregated BPCP can not be expressed when solely using one consequence pattern. For example, if 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 needed.

In [80], 55 control flow *compliance rule patterns* (CRP) were introduced. Additionally, [80] provides examples of the data and resource perspectives. The control flow compliance rule patterns are partitioned into 15 categories. We are able to show that the eCRG language supports all categories as well as their corresponding rules. Furthermore, the mentioned examples of the data and resource perspectives can be mapped to eCRGs as well. The respective eCRGs are described in a technical report [49], which also shows that the eCRG language is able to cover well-known *time patterns* [61].

4.2 Modeling Real-World Compliance Rules as eCRGs

To evaluate the practical suitability of the eCRG language, we modeled real-world compliance rules from the healthcare domain [93]. Six process model collections were analyzed by a Master student from Management Science; i.e.,

someone with competencies comparable to a business analyst rather than to an IT expert. The student checked whether the process model collections are related to compliance rules and, if yes, whether these can be modeled with the eCRG language. The student performed these analyzes within a period of six month.

The process model collections and related artifacts had been created in the context of a large process reengineering project at a university hospital (cf. Section 2). The collections contain process models related to laboratory examinations [55], radiological services [56], therapeutic treatments [92,90], and surgeries [54,91]. The corresponding process models are described both visually and as text, providing information about the data and resource perspectives (e.g., organizational units, human resources) as well. Finally, the organizational structure of the hospital is described in terms of organization charts [53].

Before the study, the Master student received specific training. She attended a course on business process management, which included a lecture on business process compliance [86], and received a 30 minutes introduction into the eCRG language. Furthermore, relevant eCRG documentations (i.e., [50, 49]) and Microsoft Visio shapes representing the eCRG elements were handed over to her.

Altogether, the Master student identified 30 compliance rules in the context of the six process model collections. Out of these 30 compliance rules, 5 refer to the interaction perspective, 8 to the time perspective, 14 to the data perspective, and 17 to the resource perspective. Note that a particular compliance rule may refer to multiple perspectives as well. The Master student was able to model each of the 30 compliance rules using the eCRG language and Microsoft Visio as modeling tool. The correctness of the eCRGs was reviewed twice. An IT expert, being familiar with the eCRG language, checked the syntax. The semantics (i.e., meaning) of the eCRGs was validated by a subject matter expert involved in the aforementioned process reengineering project.

Modeling real-world compliance rules revealed a few drawbacks regarding the modeling of the control flow, interaction and time perspectives. In particular, it emphasizes the missing ability to refine tasks or to constrain business partners sending or receiving messages. Finally, a specific symbol for explaining periodic time events was missing. Such a symbol might ease the distinction between point-in-time nodes that refer to periodic and one-time events.

4.3 Experimental Evaluation

To evaluate the comprehensibility of the eCRG language, we conducted a controlled experiment. The latter investigated the use of the eCRG language by students from Management Science (i.e. prospective business analysts) and Computer Science. All subjects had been provided with a short training before.

First, we tested whether the reading of eCRGs supports subjects in understanding the meaning of the respective rules (cf. Hypothesis H1+H2). Second,

we compared the understanding of the eCRG language for the two subject groups (i.e. Management Scientists and Computer Scientists – cf. Hypothesis H3).

H1 Trained Management Scientists are able to understand eCRGs, i.e. reading eCRGs increases their domain understanding.
H2 Trained Computer Scientists are able to understand eCRGs, i.e. reading eCRGs increases their domain understanding.
H3 There is no large difference between Management Scientists and Computer Scientists regarding the understanding of eCRGs.

As subjects 80 students from Ulm University were chosen; 59 of them were male and 21 female. 55 subjects studied Management Science [MS] and 25 Computer Science [CS]. All subjects attended a course on business process management at the time the experiment took place[3] and received course credit points for participating in the study. Furthermore, the course included a lecture on business process compliance [86], which covered the eCRG language as well. As additional material, a course book was distributed to the students.

The study is based on a questionnaire that, first of all, requests general information from the subjects (e.g., age, gender, and study program of the subject). Then, it asks for self-ratings regarding the familiarity of the respective subject with the eCRG language and related notations (e.g. BPMN). The main part of the questionnaire comprises 10 eCRGs. For each eCRG, 3 questions related to domain understandability (i.e. the expressed compliance rule) were asked. A correct answer scores with one point. Accordingly, the maximum total score is 30, expressing that the respective subject was able to answer all questions correctly. Each time after answering 6 questions (i.e., after processing two eCRGs), the subject is asked additional questions.

To rule out learning effects due to the labeling of the eCRG elements, we provided two versions of the questionnaire that alternately abstract the elements of eCRGs; i.e. concrete labels are replaced by abstract identifiers (e.g., using label *Task A* instead of label *check job application*). This ensures that any measured increase in understandability can be directly related to the eCRG language, i.e., it is not biased due to existing domain knowledge of the subjects. Finally, we recorded the time needed by the respective subjects to process the questionnaire.

Table 2 provides the descriptive statistics of the experiment regarding the scores of the subjects. Fig. 23 shows the corresponding distribution. As we cannot assume a normal distribution[4], we apply non-parametric tests in addition to parametric t-tests. While parametric tests are only considered as robust in the context of non-normal distributions [83,63], non-parametric tests are independent from this assumption.

[3] Note that we re-conducted the experiment with the same environment in order to involve a larger number of subjects (42 subjects in the first run, 38 subjects in the second run)

[4] A Kolmogorov-Smirnov test (p-value 0.006) suggested not accepting the assumption of a normal distribution.

To test H1 and H2 we check whether the scores of groups MS and CS significantly differ from random guessing. In this context we consider solely score points related to abstracted eCRGs in order to ensure that results are not biased due to the understanding of labels or domain knowledge. We perform a *one-sample t-test* and a *one-sample Wilcoxon signed-rank test*. The p-values of the tests are provided in Table 3. Both tests support H1 and H2; i.e., both tests show a very significant difference based on a 0.05 significance level.

In order to test H3, we compare the complete score of group MS with the one of group CS. For this purpose, we performed an *unpaired two-sample t-test* and a *Mann-Whitney U signed-rank test* (cf. Table 4 for p-values). Both tests do not reveal a significant difference between the scores of the two groups when presuming a 0.05 significance level. Note that this does not mean that there is no difference between the two groups, but that this difference is too small to be detected by the experiment and corresponds to a *trivial effect* (effect size according to *Cohen* is $d = 0.16$). As shown in Table 4, the power of the tests is high enough to detect medium and large effects based on the recommended

Table 2: Descriptive statistics: Score

group	N	min	max	avg	sd	sem
MS	55	10.0	30.0	20.182	4.583	0.618
CS	25	11.0	29.0	20.920	4.743	0.949
all	80	10.0	30.0	20.413	4.616	0.516

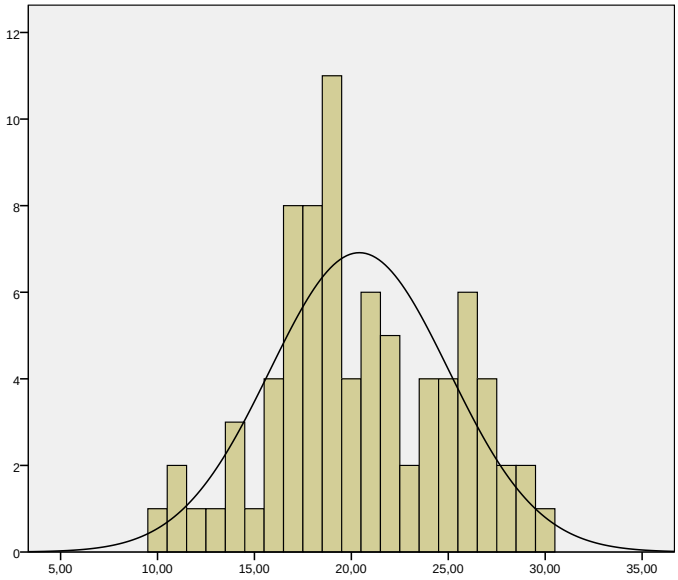


Fig. 23: Distribution of the score

Table 3: Increase in domain understanding (H1 and H2)

	one-sample t-test	one-sample Wilcoxon signed-rank
H1	<0.001	<0.001
H2	<0.001	<0.001

Table 4: Comparing group MS with group CS (H3)

	unpaired two-sample t-test	Mann-Whitney U
H3	0.511	0.670
Power ($d = 0.7 \parallel \geq \pm 3.27$)	0.819	0.801
Power ($d = 0.8 \parallel \geq \pm 3.73$)	0.906	0.892

0.80 level. Such an effect corresponds to a difference of at least 3.27 points in the context of our experiment [28,21].[5]

Altogether, the experiment confirms that Management Scientists (i.e. no IT experts) are able to understand eCRGs and their eCRG understanding can reach a level not largely differing from the one of Computer Scientists (i.e., IT experts).

Limitations. Apparently, the experiment faces several limitations. First, we did not involve and compare professional business analysts and IT experts from industry, but prospective ones (i.e. students). Although various investigations have shown that students are proper substitutes for professionals in empirical studies (e.g. [42,96]) the results for professionals may differ. Second, the experiment uses 10 prespecified eCRGs that address all process perspectives supported by the eCRG language as well as its core elements. However, the questionnaire neither included all elements of the eCRG language nor did it consider all possible combinations of language elements. Accordingly, we cannot ensure that we always obtain similar results for the various eCRGs. Finally, the sample size of the experiments only allows detecting medium and large differences.

4.4 Proof-of-Concept Prototype

To demonstrate the feasibility of *a-posteriori* compliance checking [47] based on the eCRG language, we implemented a proof-of-concept prototype.

The modeling component of this prototype is depicted in Fig. 24. On the left, the elements of the eCRG language (i.e., nodes, edges and attachments) are displayed, which may then be dragged and dropped on the drawing panel in the center. The latter allows layouting the elements and connecting them with each other. The map on the bottom left highlights the active region of the drawing panel. Furthermore, the modeling environment supports different export formats including svg, jpg, pdf, and xml.

[5] Interpretation of *Cohen*'s *d*: $d \geq 0.8$ large effect, $0.8 > d \geq 0.5$ medium effect, $0.5 > d \geq 0.2$ small effect, $d < 0.2$ trivial/no effect

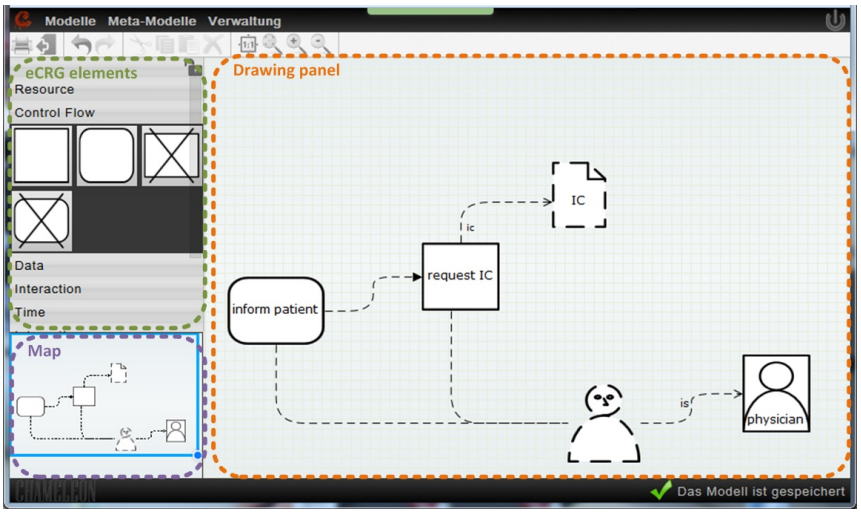


Fig. 24: Modeling Component

Another component of the prototype (cf. Fig. 25) enables *a posteriori* compliance checking of execution logs; i.e., it allows determining whether logs comply with a given eCRG. For this purpose, eCRGs created with the modeling tool can be imported.

Based on the formal semantics of the eCRG language (see [49] for details), the tool enables *a-posteriori* analyses of process execution logs to determine whether completed or running process instances comply with a particular eCRG. Users may load eCRGs and execution logs dynamically. Compliance verification starts when pressing the *verify* button on the bottom left. The corresponding result, in turn, is shown below the button. Finally, users may visualize eCRGs and execution logs.

The described prototype was applied to various scenarios and compliance rules, respectively, including the ones from the presented healthcare example. Fig. 25 provides a screenshot of the eCRG checker.[6]

5 Related Work

The eCRG language enables the visual modeling and verification of compliance rules referring to multiple perspectives of business processes. Beyond the control flow perspective, the interaction, time, resource, and data perspectives of business processes are considered. Accordingly, we structure related work into three categories: Section 5.1 presents approaches addressing the interaction, time, resource, and data perspectives in the context of process modeling.

[6] Note that the compliance checker visualizes eCRGs based on a layouting algorithm; i.e., positioning information from the modeling environment is not used.

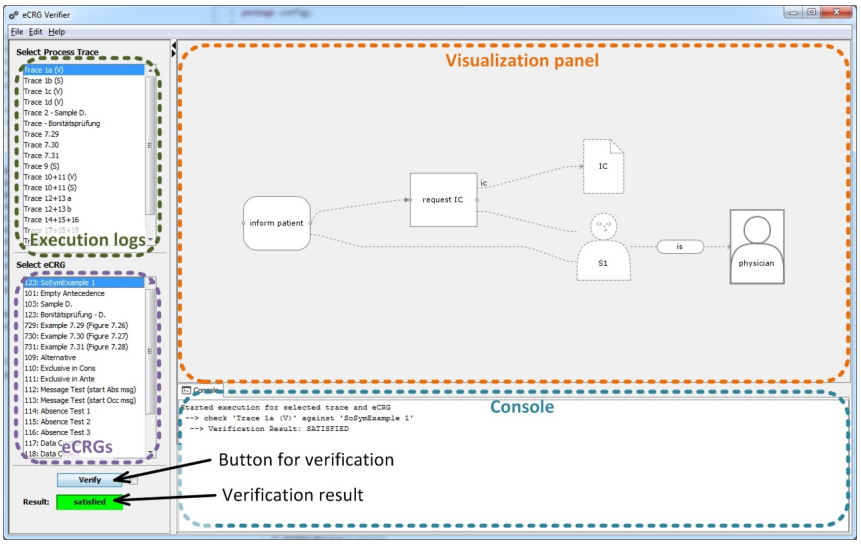


Fig. 25: Compliance Checking Component

Business process compliance and compliance verification are addressed in Section 5.2. Finally, Section 5.3 discusses other notations for modeling process compliance rules.

5.1 Perspectives of Business Processes Beyond Control Flow

Modeling issues related to the interaction, time, resource, and data perspectives of business processes have been addressed by a plethora of approaches and languages. For example, [4,22,13] deal with the interaction perspective, i.e. the exchange of messages between partners involved in a cross-organizational process. The integration of temporal constraints into business process models (i.e., time perspective) is addressed in [25,61], whereas [59,58,71] focus on the data perspective. In turn, [88,19,18] deal with the assignment of resources to business process activities (i.e., resource perspective). However, there is only little work dealing with the interplay of multiple perspectives. For example, [46] and [70] deal with the data perspective of cross-organizational business processes (i.e., the data and interaction perspective). The modeling of process-aware enterprises with respect to multiple perspectives is addressed in [17,29].

5.2 Business Process Compliance

Business process compliance has gained increasing attention over the last years and several surveys have emerged [44,5,15,51,27]. On the one hand, there exist frameworks that address the integration of business process compliance

throughout the entire process lifecycle [47,65,81]. On the other, there are approaches dealing with business process compliance in a particular stage of the process lifecyle. In particular, [2,80,78] provide techniques for *a-posteriori* verifying the compliance of execution logs with a set of constraints. Certain approaches not only focus on the control flow perspective, but take the time perspective [82] or resource perspective [14] into account as well. To be able to quickly react to compliance violations or to prevent them, *compliance monitoring* [16,31,67–69] and *continous auditing* [8] allow detecting process compliance violations during runtime. A framework for comparing respective approaches is proposed by [64]. To verify whether compliance rules are met by a particular process model at design time, a multitude of approaches exist applying model checking techniques (e.g., [30,62]). In this context, [10,45] consider the data perspective and [48,52] the interaction perspective. An approach checking the compliance of process models with respect to given semantic constraints and ensuring the validity of process change operations based on Mixed-Integer Programming formulation is proposed in [57]. The latter further introduces notions like *degree of compliance*, *validity of change operations*, and *compliance by compensation*. Other approaches for verifying compliance at design time apply the notion of *semantic congruence* [41] or use *Petri Nets* [6], considering the data and time perspectives as well. Finally, declarative approaches [32,79,7,37,99] ensure compliance in an elegant way. Since processes are defined by means of a set of constraints, imposed compliance rules only have to be added to the process definition to ensure business process compliance.

5.3 Compliance Rule Notations

To enable the verification of business process compliance rules, the latter must be specified unambiguously in a machine-readable way. For this purpose, [35] developed a logic-based formalism for describing both the semantics of normative specifications and compliance checking procedures. This approach allows modeling business obligations and regulating the execution of business processes. In turn, [24,74,97,80] apply patterns to specify compliance rules. Furthermore, there are approaches using semantic annotations to ensure compliance [33]. Other approaches rely on temporal logics (e.g., [30,45,26]), like the linear temporal logic (LTL), with which the control flow perspective can be modeled based on operators like *next*, *eventually*, *always*, and *until*.

An approach for visually modeling compliance is described in [10,11]. It considers the control flow and data perspectives; [89] additionally includes security constraints. There exist other visual approaches for compliance rule modeling [62,66,26], which focus on the control flow and partially on the data perspective, but ignore the other perspectives.

A generic querying language, which can be applied to a wide range of conceptual models is presented in [23]. In particular, this language can be used to specify compliance rules as well.

An approach connecting visual compliance rule notations with informal and textual specifications is presented in [95], whereas [98] deals with the transformation of natural text into formal compliance rules.

Visual notations for declarative business processes (e.g., DECLARE [79], DCR Graphs [40]) could be used to model compliance rules as well. Note that some of these approaches address the data and/or time perspectives [72,12].

Artifact-centric approaches [43] and business rules [39] emerged recently. Furthermore, there exist commercial as well as open source business rule engines (e.g., IBM ILOG and JBoss Drools) and related standards (e.g., BRML [36], SBVR [76]). Note that respective approaches are expressive and allow addressing the different perspectives of business processes and compliance rules. As a drawback, however, these approaches are mainly text-based, i.e., they do not provide a visual notation with an explicit support of the control flow, interaction, time, data, and resource perspectives.

6 Summary and Outlook

While compliance rule modeling has been addressed by a plethora of approaches, the integrated visual modeling of the control flow, interaction, time, data, and resource perspectives has not been sufficiently addressed yet [20,80, 97]. To remedy this drawback, this paper proposes the extended Compliance Rule Graph (eCRG) language. This language not only considers the control flow perspective, but enables the visual modeling of compliance rules with the support of the other mentioned perspectives as well.

Taking our previous work on the eCRG language [50,94] into account, we introduced the elements of the eCRG language in detail and illustrated them along examples. We showed that the eCRG language covers the various perspectives one faces when modeling compliance rules from real-world scenarios. The provided pattern-based evaluation further confirmed the expressiveness of the eCRG language. To enable tool support for both the modeling and the verification of compliance rules, a formal semantics of the presented visual compliance rule language has been provided in a technical report [49]. Based on this semantics, we implemented a proof-of-concept prototype that *a-posteriori* analyzes whether execution logs comply with given eCRGs, taking the control flow, interaction, time, data, and resource perspectives into account. To support the visual modeling of eCRGs another component of this proof-of-concept prototype is provided by the modeling environment for the eCRG language.

In future work, we will consider the feedback we gathered during the modeling of real-world compliance rules to enhance the visual compliance rule language. Furthermore, we will compare the eCRG language with pattern- and logic-based approaches in another empirical study. Our overall aim is to ensure multi-perspective compliance for all phases of the process life cycle, including runtime compliance monitoring as well as *a priori* compliance checking

at design time. Finally, we will consider compliance checking in the context of process changes.

Acknowledgements This work was accomplished in the C^{3}Pro research project, which is funded by the German Research Foundation (DFG), under Project number RE 1402/2-1, as well as the Austrian Science Fund (FWF) under project number I743.

References

1. van der Aalst, W.M.P.: Verification of Workflow Nets. In: ICATPN'97, *LNCS*, vol. 1248, pp. 407–426. Springer (1997)
2. van der Aalst, W.M.P., Beer, H.D., van Dongen, B.: Process mining and verification of properties: An approach based on temporal logic. In: OTM'05, *LNCS*, vol. 3760, pp. 130–147 (2005)
3. van der Aalst, W.M.P., ter Hofstede, A.H.: YAWL: yet another workflow language. Information Systems **30**(4), 245–275 (2005)
4. van der Aalst, W.M.P., Lohmann, N., Massuthe, P., Stahl, C., Wolf, K.: Multiparty contracts: Agreeing and implementing interorganizational processes. The Comp Journal **53**(1), 90–106 (2010)
5. Abdullah, N.S., Sadiq, S.W., Indulska, M.: Emerging challenges in information systems research for regulatory compliance management. In: CAiSE'10, *LNCS*, vol. 6051, pp. 251–265. Springer (2010)
6. Accorsi, R., Lowis, L., Sato, Y.: Automated certification for compliant cloud-based business processes. Business & Information Systems Engineering **3**(3), 145–154 (2011)
7. Alberti, M., Chesani, F., Gavanelli, M., Lamma, E., Mello, P., Montali, M., Torroni, P.: Expressing and verifying business contracts with abductive logic programming. International Journal of Electronic Commerce **12**(4), 9–38 (2008)
8. Alles, M., Kogan, A., Vasarhelyi, M.: Putting continuous auditing theory into practice: Lessons from two pilot implementations. Inf Sys **22**(2), 195–214 (2008)
9. Awad, A., Decker, G., Weske, M.: Efficient compliance checking using BPMN-Q and temporal logic. In: BPM'08, *LNCS*, vol. 5240, pp. 326–341. Springer (2008)
10. Awad, A., Weidlich, M., Weske, M.: Specification, verification and explanation of violation for data aware compliance rules. In: ICSOC'09, *LNCS*, vol. 5900, pp. 500–515. Springer (2009)
11. 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), 30–55 (2011)
12. Barba, I., Lanz, A., Weber, B., Reichert, M., Valle, C.D.: Optimized time management for declarative workflows. In: BPMDS'12, *LNCS*, vol. 113, pp. 195–210. Springer (2012)
13. Barros, A., Dumas, M., ter Hofstede, A.: Service interaction patterns. In: BPM'05, *LNCS*, vol. 3649, pp. 302–318. Springer (2005)
14. Baumgrass, A., Baier, T., Mendling, J., Strembeck, M.: Conformance checking of rbac policies in process-aware information systems. In: BPM'12 Workshops, *LNBIP*, vol. 100, pp. 435–446. Springer (2012)
15. Becker, J., Delfmann, P., Eggert, M., Schwittay, S.: Generalizability and applicability of model-based business process compliance-checking approaches — a state-of-the-art analysis and research roadmap. BuR - Business Research **5**(2), 221–247 (2012)
16. Berry, A., Milosevic, Z.: Extending choreography with business contract constraints. Coop Inf Sys **14**(2-3), 131–179 (2005)
17. Businska, L.: Multidimensional business process modeling approach. In: ADBIS'10, *LNCS*, vol. 5968, pp. 247–256. Springer (2010)
18. Cabanillas, C., Knuplesch, D., Resinas, M., Reichert, M., Mendling, J., Ruiz-Cortes, A.: RALph: A graphical notation for resource assignments in business processes. In: CAiSE'15, *LNCS*, vol. 9097, pp. 53–68. Springer (2015)

19. Cabanillas, C., Resinas, M., Cortés, A.R.: Defining and analysing resource assignments in business processes with RAL. In: ICSOC'11, *LNCS*, vol. 7084, pp. 477–486. Springer (2011)
20. Cabanillas, C., Resinas, M., Ruiz-Cortés, A.: Hints on how to face business process compliance. In: JISBD'10, pp. 26–32 (2010)
21. Cohen, J.: Statistical Power Analysis for the Behavioral Sciences. Hillsdale (1998)
22. Decker, G., Weske, M.: Interaction-centric modeling of process choreographies. Information Systems **36**(2), 292–312 (2011)
23. Delfmann, P., Steinhorst, M., Dietrich, H.A., Becker, J.: The generic model query language {GMQL} conceptual specification, implementation, and runtime evaluation. Information Systems **47**(0), 129 – 177 (2015)
24. Dwyer, M.B., Avrunin, G.S., Corbett, J.C.: Property specification patterns for finite-state verification. In: FMSP'98, pp. 7–15. ACM (1998)
25. Eder, J., Tahamtan, A.: Temporal conformance of federated choreographies. In: DEXA'08, *LNCS*, vol. 5181, pp. 668–675. Springer (2008)
26. Feja, S., Speck, A., Witt, S., Schulz, M.: Checkable graphical business process representation. In: ADBIS'11, *LNCS*, vol. 6295, pp. 176–189. Springer (2011)
27. Fellmann, M., Zasada, A.: State-of-the-art of business process compliance approaches: A survey. In: ECIS'14, pp. 1–17 (2014)
28. Field, A., Hole, G.: How to Design and Report Experiments. SAGE (2003)
29. Frank, U.: Multi-perspective enterprise modeling: foundational concepts, prospects and future research challenges. Software & Systems Modeling **13**(3), 941–962 (2014)
30. Ghose, A.K., Koliadis, G.: Auditing business process compliance. In: ICSOC'07, *LNCS*, vol. 4749, pp. 169–180. Springer (2007)
31. Giblin, C., Müller, S., Pfitzmann, B.: From regulatory policies to event monitoring rules: Towards model-driven compliance automation. Tech. Rep. RZ-3662, IBM (2006)
32. Goedertier, S., Vanthienen, J.: Designing compliant business processes with obligations and permissions. In: BPM'06 Workshops, *LNCS*, vol. 4103, pp. 5–14. Springer (2006)
33. Governatori, G., Hoffmann, J., Sadiq, S., Weber, I.: Detecting regulatory compliance for business process models through semantic annotations. In: BPM'08 Workshops, *LNBIP*, vol. 17, pp. 5–17. Springer (2009)
34. Governatori, G., Milosevic, Z., Sadiq, S.: Compliance checking between business processes and business contracts. In: EDOC'06, pp. 221–232. IEEE (2006)
35. Governatori, G., Sadiq, S.: The journey to business process compliance. In: Handbook of Research on BPM, pp. 426–454. IGI Global (2009)
36. Grosof, B.N., Labrou, Y.: An approach to using XML and a rule-based content language with an agent communication language. In: Issues in Agent Communication, *LNCS*, vol. 1916, pp. 96–117. Springer (2000)
37. Haisjackl, C., Barba, I., Zugal, S., Soffer, P., Hadar, I., Reichert, M., Pinggera, J., Weber, B.: Understanding declare models: strategies, pitfalls, empirical results. Software and System Modeling pp. 1–28 (2014)
38. Haisjackl, C., Zugal, S.: Investigating differences between graphical and textual declarative process models. In: CAiSE'14 Workshops, *LNBIP*, vol. 178, pp. 194–206. Springer (2014)
39. Herbst, H.: Business rules in systems analysis: A meta-model and repository system. Information Systems **21**(2), 147 – 166 (1996)
40. Hildebrandt, T., Mukkamala, R., Slaats, T.: Nested dynamic condition response graphs. In: FSEN'12, *LNCS*, vol. 7141, pp. 343–350. Springer (2012)
41. Höhn, S.: Model-based reasoning on the achievement of business goals. In: SAC'09, pp. 1589–1593. ACM (2009)
42. Höst, M., Regnell, B., Wohlin, C.: Using students as subjects a comparative study of students and professionals in lead-time impact assessment. Empirical Software Engineering **5**(3), 201–214 (2000)
43. Hull, R.: Artifact-centric business process models: Brief survey of research results and challenges. In: OTM'08, *LNCS*, vol. 5332, pp. 1152–1163. Springer (2008)
44. Kharbili, M.E., de Medeiros, A., Stein, S., van der Aalst, W.M.P.: Business process compliance checking: Current state and future challenges. In: MobIS'08, pp. 107–113 (2008)

45. 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, *LNCS*, vol. 6412, pp. 332–346. Springer (2010)
46. Knuplesch, D., Pryss, R., Reichert, M.: Data-aware interaction in distributed and collaborative workflows: Modeling, semantics, correctness. In: CollaborateCom'12, pp. 223–232. IEEE (2012)
47. Knuplesch, D., Reichert, M.: Ensuring business process compliance along the process life cycle. Tech. Rep. 2011-06, Ulm University (2011)
48. Knuplesch, D., Reichert, M., Fdhila, W., Rinderle-Ma, S.: On enabling compliance of cross-organizational business processes. In: BPM'13, *LNCS*, vol. 8094, pp. 146–154. Springer (2013)
49. Knuplesch, D., Reichert, M., Ly, L.T., Kumar, A., Rinderle-Ma, S.: On the formal semantics of the extended compliance rule graph. Tech. Rep. 2013-05, Ulm University (2013)
50. 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, *LNCS*, vol. 8217, pp. 106–120. Springer (2013)
51. Knuplesch, D., Reichert, M., Mangler, J., Rinderle-Ma, S., Fdhila, W.: Towards compliance of cross-organizational processes and their changes. In: BPM'12 Workshops, *LNBIP*, vol. 132, pp. 649–661. Springer (2013)
52. Knuplesch, D., Reichert, M., Pryss, R., Fdhila, W., Rinderle-Ma, S.: Ensuring compliance of distributed and collaborative workflows. In: CollaborateCom'13, pp. 133–142. IEEE (2013)
53. Konyen, I., Reichert, M., Schultheiß, B.: Organisationsstrukturen einer Universitätsklinik am Beispiel der Uni-Frauenklinik Ulm. Tech. Rep. DBIS-18, University of Ulm (1996)
54. Konyen, I., Reichert, M., Schultheiß, B., Frank, S., Mangold, R.: Ein Prozessentwurf für den Bereich der minimal invasiven Chirugie. Tech. Rep. DBIS-14, University of Ulm (1996)
55. Konyen, I., Schultheiß, B., Reichert, M.: Prozessentwurf eines Ablaufs im Labor. Tech. Rep. DBIS-16, University of Ulm (1996)
56. Konyen, I., Schultheiß, B., Reichert, M.: Prozessentwurf für den Ablauf einer radiologischen Untersuchung. Tech. Rep. DBIS-15, University of Ulm (1996)
57. Kumar, A., Yao, W., Chu, C.: Flexible process compliance with semantic constraints using mixed-integer programming. INFORMS Journal on Computing **25**(3), 543–559 (2013)
58. Künzle, V., Reichert, M.: PHILharmonicFlows: towards a framework for object-aware process management. Journal of Software Maintenance and Evolution: Research and Practice **23**(4), 205–244 (2011)
59. Küster, J., Ryndina, K., Gall, H.: Generation of business process models for object life cycle compliance. In: BPM'07, *LNCS*, vol. 4714. Springer (2007)
60. Lanz, A., Reichert, M., Weber, B.: Process time patterns: A formal foundation. Information Systems **57**, 38–68 (2016)
61. Lanz, A., Weber, B., Reichert, M.: Time patterns for process-aware information systems. Requirements Engineering pp. 1–29 (2012)
62. Liu, Y., Müller, S., Xu, K.: A static compliance-checking framework for business process models. IBM Systems Journal **46**(2), 335–261 (2007)
63. Lumley, T., Diehr, P., Emerson, S., Chen, L.: The importance of the normality assumption in large public health data sets. Annual review of public health **23**(1), 151–169 (2002)
64. Ly, L.T., Maggi, F.M., Montali, M., Rinderle-Ma, S., van der Aalst, W.M.P.: A framework for the systematic comparison and evaluation of compliance monitoring approaches. In: EDOC'13, pp. 7–16. IEEE (2013)
65. Ly, L.T., Rinderle, S., Dadam, P.: Integration and verification of semantic constraints in adaptive process management systems. Data & Knowledge Engineering **64**(1), 3–23 (2008)
66. Ly, L.T., Rinderle-Ma, S., Dadam, P.: Design and verification of instantiable compliance rule graphs in process-aware information systems. In: CAiSE'10, *LNCS*, vol. 6051, pp. 9–23. Springer (2010)

67. Ly, L.T., Rinderle-Ma, S., Knuplesch, D., Dadam, P.: Monitoring business process compliance using compliance rule graphs. In: OTM'11, *LNCS*, vol. 7044, pp. 82–99. Springer (2011)
68. Maggi, F., Montali, M., Westergaard, M., van der Aalst, W.M.P.: Monitoring business constraints with linear temporal logic: an approach based on colored automata. In: BPM'11, *LNCS*, vol. 6896, pp. 132–147. Springer (2011)
69. Maggi, F.M., Francescomarino, C.D., Dumas, M., Ghidini, C.: Predictive monitoring of business processes. In: CAiSE'14, *LNCS*, vol. 8484, pp. 457–472. Springer (2014)
70. Meyer, A., Pufahl, L., Batoulis, K., Kruse, S., Lindhauer, T., Stoff, T., Fahland, D., Weske, M.: Automating data exchange in process choreographies. In: CAiSE'14, *LNCS*, vol. 8484. Springer (2014)
71. Meyer, A., Pufahl, L., Fahland, D., Weske, M.: Modeling and enacting complex data dependencies in business processes. In: BPM'13, *LNCS*, vol. 8094, pp. 171–186. Springer (2013)
72. Montali, M., Chesani, F., Mello, P., Maggi, F.M.: Towards data-aware constraints in declare. In: SAC'13, pp. 1391–1396. ACM (2013)
73. Moody, D.L.: The physics of notations: toward a scientific basis for constructing visual notations in software engineering. IEEE Transactions on Software Engineering **35**(6), 756–779 (2009)
74. Namiri, K., Stojanovic, N.: Pattern-Based design and validation of business process compliance. In: OTM'07, LNCS, pp. 59–76. Springer (2007)
75. OMG: BPMN 2.0. Recommendation, OMG (2011)
76. OMG: SBVR 1.3. Recommendation, OMG (2015)
77. Ottensooser, A., Fekete, A., Reijers, H.A., Mendling, J., Menictas, C.: Making sense of business process descriptions: An experimental comparison of graphical and textual notations. Journal of Systems and Software **85**(3), 596–606 (2012)
78. Outmazgin, N., Soffer, P.: A process mining-based analysis of business process workarounds. Software & Systems Modeling pp. 1–15 (2014)
79. Pesic, M., Schonenberg, H., van der Aalst, W.M.P.: DECLARE: full support for loosely-structured processes. In: EDOC'07, pp. 287–300. IEEE (2007)
80. Ramezani, E., Fahland, D., van der Aalst, W.M.P.: Where did i misbehave? diagnostic information in compliance checking. In: BPM'12, *LNCS*, vol. 7481, pp. 262–278. Springer (2012)
81. Ramezani, E., Fahland, D., van der Werf, J.M., Mattheis, P.: Separating compliance management and business process management. In: BPM'11 Workshops, *LNBIP*, vol. 100, pp. 459–464. Springer (2012)
82. Ramezani Taghiabadi, E., Fahland, D., van Dongen, B.F., van der Aalst, W.M.P.: Diagnostic information for compliance checking of temporal compliance requirements. In: CAiSE'13, *LNCS*, vol. 7908, pp. 304–320. Springer (2013)
83. Ratcliffe, J.F.: The effect on the t distribution of non-normality in the sampled population. Journal of the Royal Statistical Society. Series C (Applied Statistics) **17**(1), 42–48 (1968)
84. Reichert, M., Dadam, P.: ADEPT$_{flex}$ – supporting dynamic changes of workflows without losing control. Intelligent Inf Sys **10**(2), 93–129 (1998)
85. Reichert, M., Rinderle, S., Kreher, U., Dadam, P.: Adaptive process management with ADEPT2. In: ICDE'05, pp. 1113–1114. IEEE (2005)
86. Reichert, M., Weber, B.: Business Process Compliance, pp. 297–320. Springer (2012)
87. Reichert, M., Weber, B.: Enabling Flexibility in Process-Aware Information Systems - Challenges, Methods, Technologies. Springer (2012)
88. 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, *LNCS*, vol. 3520, pp. 216–232. Springer (2005)
89. Salnitri, M., Dalpiaz, F., Giorgini, P.: Modeling and verifying security policies in business processes. In: BPMDS'14, *LNBIP*, vol. 175, pp. 200–214. Springer (2014)
90. Schultheiß, B., Meyer, J., Mangold, R., Zemmler, T., Reichert, M.: Prozessentwurf für den Ablauf einer stationären Chemotherapie. Tech. Rep. DBIS-5, University of Ulm (1996)

91. 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. Tech. Rep. DBIS-6, University of Ulm (1996)
92. Schultheiß, B., Meyer, J., Mangold, R., Zemmler, T., Reichert, M., Dadam, P., Kreienberg, R.: Prozessentwurf für den Ablauf einer ambulanten Chemotherapie. Tech. Rep. DBIS-7, University of Ulm (1996)
93. Semmelrodt, F.: Modellierung klinischer Prozesse und Compliance Regeln mittels BPMN 2.0 und eCRG. Master Thesis, Ulm University, Germany (2013)
94. Semmelrodt, F., Knuplesch, D., Reichert, M.: Modeling the resource perspective of business process compliance rules with the extended compliance rule graph. In: BPMDS'14, *LNBIP*, vol. 175, pp. 48–63. Springer (2014)
95. Sunkle, S., Kholkar, D., Kulkarni, V.: Toward better mapping between regulations and operational details of enterprises using vocabularies and semantic similarity. In: CAiSE'15 Forum, pp. 229–236. CEUR-WS (2015)
96. Svahnberg, M., Aurum, A., Wohlin, C.: Using students as subjects - an empirical evaluation. In: ESEM'08, pp. 288–290. ACM (2008)
97. Turetken, O., Elgammal, A., van den Heuvel, W.J., Papazoglou, M.: Capturing compliance requirements: A pattern-based approach. IEEE Software pp. 29–36 (2012)
98. Zasada, A., Fellmann, M.: A pattern-based approach to transform natural text from laws into compliance controls in the food industry. In: LWA'15 Workshops, pp. 230–238. CEUR-WS (2015)
99. Zugal, S., Soffer, P., Haisjackl, C., Pinggera, J., Reichert, M., Weber, B.: Investigating expressiveness and understandability of hierarchy in declarative business process models. Software and System Modeling **14**(3), 1081–1103 (2015)