Integration of Process Constraints from Heterogeneous Sources in Process-Aware Information Systems

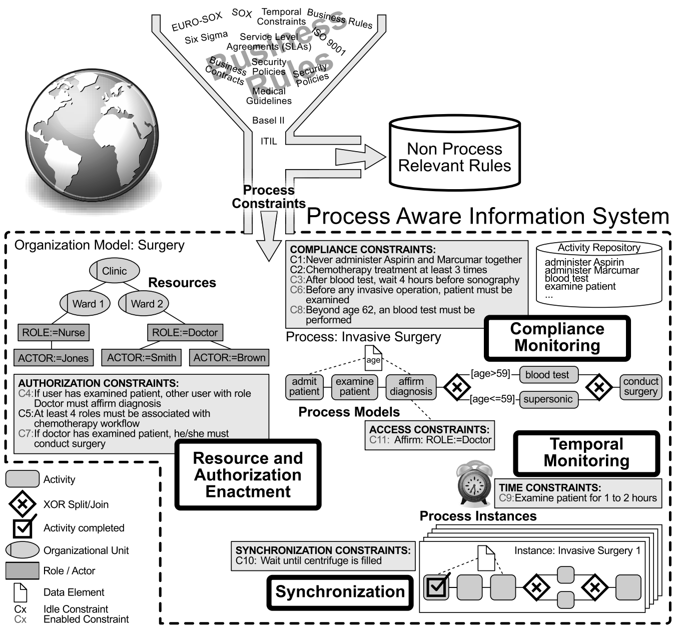
Stefanie Rinderle-Ma and Juergen Mangler
University of Vienna, Austria
Faculty of Computer Science
{stefanie.rinderle-ma,juergen.mangler}@univie.ac.at

Abstract: Business Process Compliance (BPC) has gained significant momentum in research and practice during the last years. Although many approaches address BPC, they mostly assume the existence of some kind of unified base of process constraints and focus on their verification over the business processes. However, it remains unclear how such an integrated process constraint base can be built up, particularly at the presence of process constraints stemming from heterogeneous sources such as internal controls, external contracts, or security and privacy policies. Hence in this paper, we propose a representation framework for the integration of process constraints stemming from different sources. The representation framework is generic such that existing formalisms to represent process constraints can be integrated. The framework is illustrated by means of real-world process constraints from different domains and heterogeneous sources. Altogether process constraint integration enables consistency checks and optimizations as well as maintenance and evolution of the constraint base.

1 Introduction

Business Process Compliance (BPC) has become a major challenge for enterprises nowadays. BPC requires that business processes comply with certain relevant rules, regulations, or norms. Although many approaches address BPC [NS07, LSG07, SGN07, LRD10, AWW11], they mostly assume the existence of some kind of integrated base of process constraints and focus on the verification of these constraints over the business processes. By *process constraints* we refer to those rules in the domain of interest, that are associated with processes. As literature and practice show, the co-existence of processes and process constraints is common and desired [HG09, LRD10]. However, it remains unclear how such an integrated process constraint base can be built up and used for compliance checks within the Process-Aware Information System (PAIS), even though this constitutes the essential prerequisite for all further compliance checks.

The particular challenge of providing an integrated process constraint base results from the heterogeneity of sources process constraints might stem from. First of all, the sources and purposes of process constraints might be different. Examples comprise internal quality directives such as Six Sigma or ITIL, external controls such as standards (e.g., ISO 001), regulations by authorizing bodies, guidelines [PT09], or regulations based on contracts (business contracts) [SGN07]. For a specific application scenario consider Fig. 1



EURO-SOX
SOX
Temporal Constraints
Business Rules
Six Sigma
Service Level Agreements (SLAs)
ISO 9001
Business Rules
Business Contracts
Security Policies
Security Policies
Medical Guidelines
Basel II
ITIL
Non Process Relevant Rules
Process Constraints
Process Aware Information System
Organization Model: Surgery
Clinic
Resources
Ward 1
Ward 2
ROLE:=Nurse
ROLE:=Doctor
ACTOR:=Jones
ACTOR:=Smith
ACTOR:=Brown
AUTHORIZATION CONSTRAINTS:
C4:If user has examined patient, other user with role Doctor must affirm diagnosis
C5:At least 4 roles must be associated with chemotherapy workflow
C7:If doctor has examined patient, he/she must conduct surgery
COMPLIANCE CONSTRAINTS:
C1:Never administer Aspirin and Marcumar together
C2:Chemotherapy treatment at least 3 times
C3:After blood test, wait 4 hours before sonography
C6:Before any invasive operation, patient must be examined
C8:Beyond age 62, an blood test must be performed
Activity Repository
administer Aspirin
administer Marcumar
blood test
examine patient
...
Compliance Monitoring
Process: Invasive Surgery
age
admit patient
examine patient
affirm diagnosis
[age>59]
[age<=59]
blood test
supersonic
conduct surgery
Process Models
ACCESS CONSTRAINTS:
C11: Affirm: ROLE:=Doctor
Resource and Authorization Enactment
Temporal Monitoring
TIME CONSTRAINTS:
C9:Examine patient for 1 to 2 hours
Process Instances
SYNCHRONIZATION CONSTRAINTS:
C10: Wait until centrifuge is filled
Instance: Invasive Surgery 1
Synchronization
Activity
XOR Split/Join
Activity completed
Organizational Unit
Role / Actor
Data Element
Cx Idle Constraint
Cx Enabled Constraint

Figure 1: Process Constraints Stemming from Heterogeneous Sources Imposed on Current PAIS

that displays example processes and rules from the medical domain: here, business processes might be subject to medical guidelines [PT09] on the one side and also subject to authorization and privacy constraints on the other side. Fig. 1 also covers simple resource synchronization as well as temporal rules. The basic challenge is to integrate all these different kinds of process constraints within one representation in order to enable consistency checks, integrated compliance verification, and a better support for constraint base maintenance. The latter becomes important since process constraints are often subject to change and evolution [LRGD09].

In this paper, we present a framework that enables the integrated representation for process constraints stemming from heterogeneous sources.[1] The framework covers process perspectives such as control flow, data flow, time, and resources that can be subject to process constraints. The integrated representation also enables the specification of behavior which is relevant for process constraints that require some sort of action (e.g., synchronization between process instances). The design of the representation framework also enables the integration of existing specification formalisms for process constraints, for example, SeaFlows [LRD10], logic-based rules [PSvdA07], or BPMN-Q constraints [AWW11]. The unified representation is validated against a selection of real-world process constraints. We further discuss how such a framework can be implemented within PAIS. Altogether the proposed approach provides a first step towards the integrated management, verification, and evolution of process constraints in PAIS, tackling the challenge

[1]An extended version is available as technical report [MRM11].

52

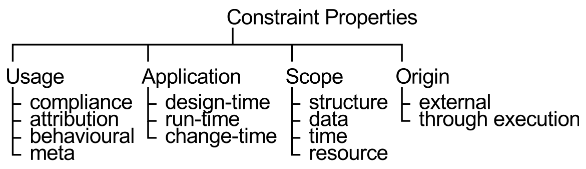


Figure 2: Process Constraint Properties

of heterogeneous sources for process constraints.

Sect. 2 discusses constraint properties in PAIS. The representation formalism for integrated process constraints is presented in Sect. 3 followed by an evaluation in Sect. 4. Sect. 5 compares related approaches and Sect. 6 concludes with summary and outlook.

2 Constraint Properties in Process-Aware Information Systems

In the previous sections, a unified representation for process constraints has been presented. In this section, we discuss its usage within Process-Aware Information Systems (PAIS) along different constraint properties as set out in Fig. 2.

Usage: Process constraints can be used to verify the compliance of business processes with relevant regulations and constraints. This is useful for process design and evolution. Constraints can also be used to alter or affect the behavior of processes (*behavioral*). Affecting the behavior of processes includes the attribution of process activities or, more generic, the attribution of structural patterns. Examples include resources allocation of process activities.

Additionally, constraints can be used to specify *meta* constraints. Meta constraints are intended to check the consistency of other constraints, for example, constraint C5 in Fig. 1. Another example is a meta constraint specifying that for each process activity referring to a resource `centrifuge` synchronization constraint C10 in Fig. 1 is assigned to.

Application: Application deals with the question at which phase of a process life cycle constraints may be enforced. Basically, at *design-time* constraints are checked to verify that a process complies to certain criteria (compliance) constraints. All constraints that are checked at design-time are *compliance* constraints or *meta* constraints.

All *behavioral* constraints, and some *compliance* constraints are checked at *run-time*. Examples include the checking for data value constraints, or the attribution of structural patterns (the attributes are used at run-time, though can be checked by meta constraints at design-time). Compliance constraints may affect run-time under certain circumstances. E.g. although a process may not conform to a specific constraint at design-time, it may do so at run-time, because only a certain execution path violates the constraint. Hence corresponding process instances are to be monitored at run-time [LRD10].

Scope: The most important scope perspective is *structure* as in all constraints structural

53

information has to be present (either explicit or implicit through connections of information to structure), as otherwise they would not be connected to processes and could thus not be enforced in PAIS. Note that if a constraint holds on only structural information, it is always a *compliance* constraint.

Data is also an integral part of a process tightly connected to structure. At design-time, it is possible to check if data types and data flow conform to a certain schema. Further it can be checked whether certain data values will lead to compliance violations are run-time. If, for example, a treatment process states that a lab test is to be performed for all patients beyond 65 years and the corresponding medical guideline states that the lab test is mandatory for patients beyond 62 year, it can be already checked at design-time, that for patients between 62 and 64 there will be a violation of the corresponding constraint at run-time.

Resource and *time* are attributes that are typically connected to structure (or data). Their purpose is to describe information that aids the execution of a process. *Resource-aware* and *time-aware* constraints in PAIS are typically checked or enforced at run-time including:

- Resource assignment to process activities, typically specified based on access constraints (e.g., constraint C11 in Fig. 1). Based on role assignment, process activities are offered to authorized actors in their work lists at run-time (e.g. by a RBAC component).
- On top of access constraints, authorization constraints can be specified such as dynamic separation of duties. Dynamic authorization constraints are verified during run-time.
- Assign temporal information to activities (e.g. the normal duration of a certain activity is 2 hours with a standard deviation of 10 minutes).

Origin: All constraints become available through not specified **external** resources, either at design time, run-time or change time. They are identified and structured by constraint designers and then made available through a constraint-base, and have to be enacted from this point on. One exception is constituted by SLAs for dynamic service selection. These constraints become available *through execution* of a process instance, and vanish after the instance finishes.

3 A Framework for Process Constraint Integration

In this section the framework for integrating process constraints in PAIS is introduced that should cover all constraint properties set out in Section 2. As fundamental property, process constraints are associated with one or several processes that are implemented and executed within the PAIS of interest. As process literature study shows, all formalisms on process constraints (implicitly) incorporate a structural pattern that contains at least one activity either executed within a process or stored within the process activity repository.

In the SeaFlows project [LRD10], for example, process constraints are tied to one or more process activities and consist of an *antecedent* and a *consequence* part. More precisely, based on a triggering antecedent pattern, a consequence pattern must hold to fulfill the constraint. Process constraints as defined in DECLARE also refer to process activities and the semantics for relations between activities are based on LTL (Linear Temporal Logic) [PSvdA07]. BPMN-Q offers compliance patterns containing structural patterns within the triggering part of the constraint [AWW11].

In the following section we will define an extended concept of *Linkage* for process constraints that subsumes the structural pattern of the constraint, but might further incorporate information on the *Context* of the process constraint, i.e., process types or process instances the constraints refer to, and the *Trigger position*. The latter is required for capturing process constraints that can trigger an action such as synchronization.

3.1 Expressing Process Context by Linkage

The following definition of *Linkage* integrates the $\texttt{Context}_c$ of a process constraint c together with its structural pattern $\texttt{SP}_c$ and a trigger position $\texttt{TP}_c$ (cf. Fig. 3).

Definition 1 (Linkage) *Let $\mathcal{P}$ be a set of all process types of consideration and $\mathcal{I}_P$ be the set of process instances running according to a process type $P \in \mathcal{P}$. A process type $P \in \mathcal{P}$ is described by a process schema S_P := (A_P, E_P, D_P)[2] where A_P denotes the set of activities, E_P the set of control/data edges, and D_P the set of data elements S_P consists of. Then the linkage $Linkage_c$ to a process type P is defined as follows:*

$$\begin{aligned}
&\mathit{Linkage}_c \subseteq \mathit{Context}_c \times \mathit{SP}_c \times \mathit{TP}_c && \textit{where}\\
&\mathit{Context}_c \subseteq \mathcal{P} \times \mathcal{I}_P && \wedge\\
&\quad \exists \mathit{SP}_c && \wedge\\
&\quad\quad \mathit{TP}_c \in \emptyset, \mathit{before}(a_{n,P}), \mathit{after}(a_{n,P})
\end{aligned}$$

In Def. 1, $\texttt{Context}_c$ describes in which processes or process instances a constraint may occur. Possibilities include single instances (e.g. SLAs for services that have been dynamically selected), all instances of a process (attribution of resources to tasks) or even instances in multiple processes (when a resource is used in multiple processes, and synchronization has to take place). Structural patterns $\texttt{SP}_c$ express the association between process constraint and process. In connection with *context*, it defines for which parts of process types and process instances the corresponding process constraint is to be enforced or verified. Structural patterns may not only spawn single activities but also several activities connected through complex control decisions. Finally, the trigger position $\texttt{TP}_c$ is relevant for synchronization constraints, since synchronization constraints are constraints that not only set out certain conditions on process execution, but enforce an action. In this

[2]In order to stay meta-model independent, we assume a simple generic representation for process schemas that can be adopted by any (imperative) process meta model.

case the trigger position specifies whether the action should take place before the affected activity is started or after. The Trigger Position is solely present for run-time (behavioral) constraints. As a structural pattern may not only spawn a single activity but also several activities connected through complex control decisions, it is necessary to determine when exactly a constraint has to fire. This is the equivalent of describing the condition under which an event occurs, in an Event Condition Action (ECA) rule. The trigger position itself is simple: before or after an activity occurs. Of course multiple before / after positions can be specified.

Both, $\texttt{SP}_C$ and $\texttt{TP}_c$ are grouped as *Connection* in Fig. 3 to express their tight integration. A $\texttt{TP}_c$ can not exists without a set of activities matched by a structural pattern $\texttt{SP}_c$. Apart from $\texttt{TP}_c$, linkage information can be statically matched against processes, thus it is possible to determine *enabled* and *idle* constraints (as described before).

From now on we will denote a linkage $\texttt{Linkage}_c$ in the compact form:

$$\texttt{Linkage}_c : ((\mathcal{P}, \mathcal{I}_P), \texttt{SP}_c, \texttt{TP}_c)$$

Consider compliance constraint C6 depicted in Fig. 1: "`C6: Before any invasive operation, patient must be examined`".

The antecedent pattern "`existence of activity invasive operation`" within a process triggers the check whether this activity is preceded by an activity "`patient examination`". The linkage for this compliance constraints turns out as

$$\texttt{Linkage}_{C6} : ((\texttt{Invasive Surgery}, \texttt{ALL}), \texttt{SP}_{C6}, \emptyset)$$

with

$$\begin{aligned} \texttt{SP}_{C6} : & \exists a_1\, Is(a_1, \texttt{examine patient}) && \wedge \\ & \exists a_2\, Is(a_2, \texttt{conduct surgery}) && \wedge \\ & a_1 \mathcal{A}^* a_2 \end{aligned}$$

where

$$\mathcal{A}^* : \textit{arbitrary activities between } a_1 \textit{ and } a_2$$

3.2 Integrating Data, Time, and Resources

Existing approaches mostly deal with control flow constraints, i.e., constraints that are only referring to structural patterns within a process [AWW11]. The only approaches that have addressed data flow aspects within process constraints are SeaFlows [KLR+10] and BPMN-Q [AWW11]. In accordance to these approaches, the data flow perspective within a process constraint can be represented as condition on the structural pattern it refers to. Consider constraint C8 from Fig. 1: "`C8: Beyond age 62, a blood test must be performed`".

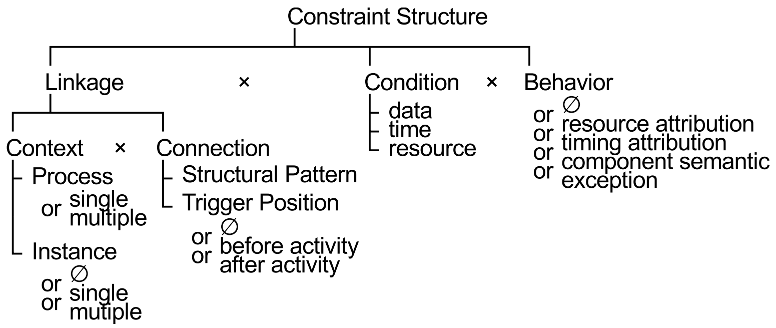


Figure 3: Process Constraint Structure (Integrated Representation)

We can see that the data flow condition "`Beyond age 62`" imposes a restriction on the structural pattern of the constraint ("`blood test`"). However, control and data flow are only two perspectives of a process. As case studies show, process constraints might also refer to conditions on time and resources, e.g., constraints C3, C4, C5, C7, C9, and C11 within the example depicted in Fig. 1. Hence, constraint integratoin requires the specification of data, time, and resource conditions on top of the linkage part of the constraint.

Definition 2 (Condition) *Let c be a process constraint with linkage $\texttt{Linkage}_c$ referring to a process type P described by process schema on linkage S_P := (A_P, E_P, D_P). The condition of c imposed on $\texttt{Linkage}_c$ is defined as*

$$\texttt{Condition}_c \subseteq \texttt{expr}(D_P) \times \texttt{expr}(\texttt{time}_c) \times \texttt{expr}(\texttt{resource}_c)$$

where

- *expr(D_P) denotes a logical expression over the data elements of P*
- *expr($time_c$) denotes a temporal expression and*
- *expr($resource_c$) denotes a logical expression over the resources associated with P (typically modeled within a organizational or resource model)*

It is important to mention that a condition cannot exist without referring to a linkage. Thus, a condition describes, under the premise of a given linkage, either (a) a combination of data, temporal or resource conditions that have to be present or (b) a combination of data, temporal or resource conditions that have to be present in order for a given behavior to be apply.

For constraint C6, for example, no specific condition is imposed on the linkage part. Hence:

$$\texttt{Condition}_{C6} : \emptyset$$

When considering constraint C3 in Fig. 1, things get more interesting, since "`After a blood test wait 4 hours before sonography`" obviously imposes a time condition on the linkage part. Less obviously, an additional condition is imposed on the data flow, since the 4 hours time frame between blood test and sonography are only required for the same patient, formally:

$$\texttt{Linkage}_{C3} : (\texttt{Invasive Surgery}, \texttt{ALL}), \texttt{SP}_{C3}, \emptyset)$$

with

$$\begin{aligned}\texttt{SP}_{C3} : &\exists a_1\ Is(a_1, \texttt{blood test}) && \wedge\\ &\exists a_2\ Is(a_2, \texttt{sonography}) && \wedge\\ &a_1 \mathcal{A}^* a_2\end{aligned}$$

and

$$\begin{aligned}\texttt{Condition}_{C3} : &\mathit{patient}(a_1) = \mathit{patient}(a_2) && \wedge\\ &\mathit{min\_time\_between}(a_1, a_2, 4h)\end{aligned}$$

Similar considerations can be made for data ("C8: `beyond age 62`") and resources (C7: `examination by same user as surgery`) which can be also represented by a condition part imposed on the linkage of a constraint.

3.3 Expressing Behavior within Process Constraints

The last missing piece is, given a certain linkage and condition, to allow for a certain assignment or behavior. Assignment becomes necessary for access constraints such as C11. Behavior specifies for example the action part of a synchronization constraint such as C10. Hence, we complete the unified constraint representation by a *Behavior* part.

Definition 3 (Behavior) *Let c again be a process constraint. For a given* $\texttt{Linkage}_c$ *and* $\mathit{Condition}_c$*, a behavior can be either empty, the attribution of resource or time information or instructions specific for process execution (e.g. exceptions).*

3.4 Summary: Linking, Condition, Behavior

The overall representation for process constraints consisting of linkage, condition, and behavior is summarized in Fig. 3. In Sect. 4 we will evaluate the unified representation against all case study constraints shown in this paper and existing related approaches.

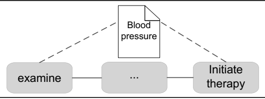
4 Evaluation

In Sect. 4.1, we show how a selection of real-world process constraints can be expressed based on the integrated representation framework . In fact, we have analyzed several application domains and related rules within SeaFlows [LRD10] throughout the last 5 years. Additionally, the examples from the medical domain (cf. Fig. 1) are analyzed.

4.1 Process Constraints from Different Domains

Medical Guidelines: A first medical guideline taken from the report "Prevention, Detection, Evaluation, and Treatment of High Blood Pressure" [U.S03] is depicted in Fig. 4-R1) together with a fragment of a corresponding treatment process. An analysis of this constraints over the process shows that `blood pressure` $\geq$ `20/10 mmHg above goal blood pressure` refers to the data perspective (data element `blood pressure`) and part `initiating therapy` refers to structural pattern `initiate therapy`).

R1: Medical Guideline - Blood Pressure
If blood pressure is > 20/10 mmHg above goal blood pressure, consideration should be given to initiating therapy with two agents, one of which usually should be a thiazide-type



R2: Medical Guideline - Vaccination
The minimum age for the first dose is six weeks. The second dose is to be administered after a minimum of 4 weeks if the first dose administered at age > 12 monts and current age < 24 months, a minimum of 8 weeks (as final dose) at age > 12 month and current age between 24 and 59 months. For healthy children, no further doses needed if the the first dose administered at age > 24 month.

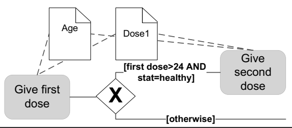


Figure 4: Medical Guidelines

The second medical guideline (cf. Fig. 4-R2) describes the vaccination of children for Pneumococcal [PT09]. This guideline refers to the data perspective, e.g., data element `age`) and structural pattern

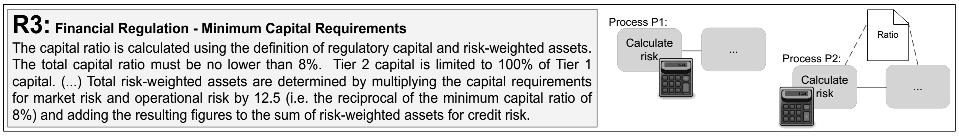
$$\begin{aligned} \mathtt{SP}_{R2} : & \exists a_1 \, Is(a_1, \mathtt{give\ first\ dose}) && \wedge \\ & \exists a_2 \, Is(a_2, \mathtt{give\ second\ dose}) && \wedge \\ & a_1 \mathcal{A}^* a_2 \end{aligned}$$

In addition, it refers to temporal aspects, i.e., certain time intervals between vaccination should be maintained.

Financial Sector: Fig. 5-R3 shows a financial regulation taken from regulatory package BASEL II [Bas06]. Here the basic question is whether "capital ratio" is a process data element or an application data element, i.e., whether the constraint refers to the data perspective or application perspective. If we assume that capital is solely application data as depicted for Process P1, the constraint above does not refer to any structural aspect of the process and hence is not considered as process constraint. By contrast, as for Process P2, the capital ratio is still calculated by the application connected with activity `calculate`. However, the value is explicitly written as process data element `ratio`. Consequently, the linkage part of the constraint refers to structural pattern `calculate`.

Authorization: Separation of duties is one example for authorization constraints [SM10]. Assume the constraint depicted in Fig. 6-R4. It refers to the functional perspective

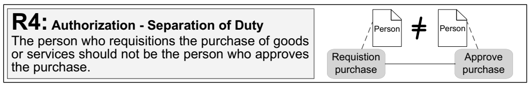
59



R3: **Financial Regulation - Minimum Capital Requirements**
The capital ratio is calculated using the definition of regulatory capital and risk-weighted assets. The total capital ratio must be no lower than 8%. Tier 2 capital is limited to 100% of Tier 1 capital. (...) Total risk-weighted assets are determined by multiplying the capital requirements for market risk and operational risk by 12.5 (i.e. the reciprocal of the minimum capital ratio of 8%) and adding the resulting figures to the sum of risk-weighted assets for credit risk.

Figure 5: Financial Regulation (BASEL II)

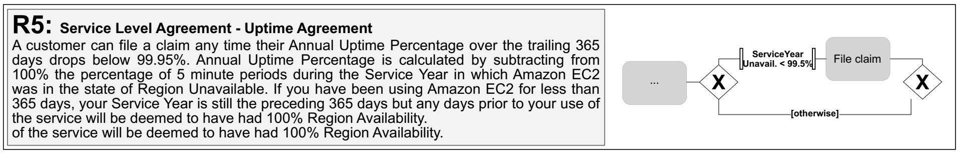
based on the corresponding process activities `requisite purchase` and `approve purchase`. Further it refers to the organizational perspective by imposing a mutual exclusion constraint on process resources.



R4: **Authorization - Separation of Duty**
The person who requisitions the purchase of goods or services should not be the person who approves the purchase.

Figure 6: Authorization

Service Level Agreements (SLAs): The following constraint (cf. Fig. 7-R5) was taken from http://aws.amazon.com/ec2-sla/. Its linkage refers to the corresponding process via structural pattern `file claim`, to the data perspective for the decision constraint `ServiceYear-Unavailable` < 99.5, and to the time perspective via restrictions over the trailing. Further (SLAs) can be found in regulation packages for internal quality such as Six Sigma or ITIL.



R5: **Service Level Agreement - Uptime Agreement**
A customer can file a claim any time their Annual Uptime Percentage over the trailing 365 days drops below 99.95%. Annual Uptime Percentage is calculated by subtracting from 100% the percentage of 5 minute periods during the Service Year in which Amazon EC2 was in the state of Region Unavailable. If you have been using Amazon EC2 for less than 365 days, your Service Year is still the preceding 365 days but any days prior to your use of the service will be deemed to have had 100% Region Availability.
of the service will be deemed to have had 100% Region Availability.

Figure 7: Uptime Agreement

4.2 Evaluation

The feasibility of the unified representation is evaluated by means of the overall 16 process constraints and along the constraint properties described in Section 2. For this we classify the 16 process constraint into different constraint types that embody a combination of properties as represented by the unified representation.

As depicted in Tab. 1 we identified 14 different process constraint types. The first two constraint types deal with *Resource Attribution* (e.g., roles, actors, nodes) and *Timing Information Attribution* (e.g., minimal, maximal, average duration) to process structure. All

Constraint Structure (Fig. 2)						Constraint Properties (Fig. 3)													Constraint Type	Example
Linkage				Condition	Behaviour	Application			Scope				Origin		Usage					* ... some constraints may be mentioned more than once depending on the realization in the process (e.g. data-flow v.s. tasks)
Context		Connection																		
Process	Instance	Struct. Pat.	Trigger Pos.			design-time	run-time	change-time	structure	data	time	resource	external	through exec.	compliance	attribution	behaviour	meta		
X	X	X	∅	~	e.g. ROLE:=Doctor		X		X	~	~	X	X			X			Resource Attribution	C11
X	X	X	∅	~	e.g. DURATION:=12 minutes		X	~	X	~	X	~	X			X			Timing Information Attribution	
X	X	X	∅	X	∅	X		~	X	~	~	~	~	~	X				Business Compliance Constraints	
X	X	X	∅	X	∅		X		X	~	X	~	X			X			Separation/Binding of Duty	C4,R4
X	X	X	∅	X	∅		X	X	X	X	~	~	X	~	X		~		Data Constraints	C1*,C8,R1
X	X	X	∅	X	~ exception, e.g. send warning		X	X	X	~	X	~	X	~	X		~		Temporal Constraints	C3,C9,R2
X	X	X	∅	∅	∅	X		~	X				X	~	X				Structural Costraints	C1*,C6,C7
X	X	X	∅	∅	∅		X		X				X	~	X		~		Structural Constraints (run-time)	C2,R3
X	X	X	~	~	~ exception, e.g. send warning	~	X	~	X	~	~	~	X	~	~		~		Security	
X	X	X	X	X	comp. semantic, e.g. delay spec.		X		X	~	~	~	X	~			X		Synchronisation	C10
X	X	X	X	X	comp. semantic, e.g. algorithm		X		X	~	~	~	X	~			X		Active	
X	X	X	~	~	~		X	~	X	~	~	~		X	X		~		SLA (dyn. service sel.)	
X	X	X	~	~	~	~	X	~	X	~	~	~	X		X		~		SLA (service sel. at design-time)	R5
X	X	X	∅	X*	component semantic	X		X	X	~	~	~	X	~				X	Meta (check other constraints)	C5

X ... neccessary
X* ... check if e.g. behavior parts are correct
∅ ... not allowed / empty
~ ... optional

X ... default
~ ... possible
... not possible

Table 1: Constraint Types

this information is necessary at run-time, for a temporal monitor as a basis to verify if a process behaves correctly, or an RBAC system to select actors in a worklist. The only difference between these two is, that during change-time timing information may have to be adapted to account for the duration of changes. E.g. constraint C11 in Fig. 1 can represented as follows:

$$\begin{aligned}
\texttt{Linkage}_{C11} &: ((\texttt{Invasive Surgery}, \texttt{ALL}), \texttt{SP}_{C11}, \emptyset) \qquad \textit{with} \\
\texttt{SP}_{C11} &: \exists a_1\ Is(a_1, \texttt{affirm diagnosis}) \\
\texttt{Condition}_{C11} &: \emptyset \\
\texttt{Behavior}_{C11} &: \{ROLE := Doctor\}
\end{aligned}$$

For attribution $\texttt{TP}_c$ is always empty, as the matched structural pattern implies, that the attribution is available all the time.

The third constraint type describes *Business Compliance Constraints* in general. We decided to include this very generic constraint type (which should not be confused with several of the other constraint types) to show some specific characteristics we derived by studying constraint sources. Business compliance constraints, are exclusively compliance constraints, they never carry a behavioral part, they most of the time verify a certain structure of the process, with a possibility of checking also data, resource and temporal aspects. Therefore the design-time *Structural constraints* also in this list are *Business Compliance Constraints*, as well as some *Temporal Constraints* or *Data constraints* (whenever only design-time aspects are covered). *Separation / Binding of Duty*, *Resource Attribution*, and *Timing Information Attribution* are the exception to the rule. They may be very well applied at design time, just only the impact can only be seen at run-time.

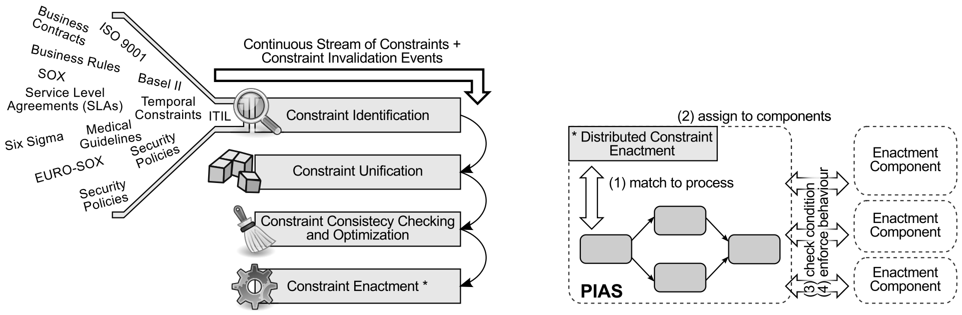


Figure 8: Unified Constraint Optimization and Checking

4.3 Application Scenarios for Integrated Process Constraint Representation

Fig. 8 illustrates how presented approach can streamline the maintenance and evolution of new process constraints within Process-Aware Information Systems (PAIS).

- Process constraints can be identified based on structural patterns and enriched with process context. At some point it may be necessary to create new processes (structural constraints often trigger the creation or evolution of existing processes).
- Constraints can be expressed and stored in an integrated manner based on the representation presend in this paper.
- Based on the integrated representation, constraint consistency checking and optimization can be supported by an automatic process. Because all constraints are available in a single consistent representation, (unlike for current approaches) it is easily possible to check if inconsistencies between constraints exists (particularly if they stem from different sources), rather than checking consistency between pure structural process constraints only. In current approaches the constraints are embedded and encapsulated in different components of the PAIS (e.g., process engine and organizational model) such that integrated checks are not possible.
- Constraint enactment is also different from current approaches. As depicted in Fig. 8 step (1) the *Linkage* can be matched on behalf of the PAIS, the checking of condition and the enactment of behavior can be move to (potentially external) components.

5 Related Work

Many approaches for checking BPC either at design or run time exist, e.g., [LSG07, SGN07]). As argued before, an integrated constraints representation has been outside

62

Representation	SeaFlows [LRD10]	DECLARE [PSvdA07]	BPMN-Q [AWW11]	Integrated Representation
Linkage	activity set	control flow pat.	control flow pat. global scope	control flow pat. process context trigger position
Condition	data	–	data	data, time, resource
Behavior	–	–	–	✓

Table 2: Comparison of Existing Approaches

the scope of these approaches so far. Validation of the integrated representation compared to selected existing approaches is presented in Table 2.

The above mentioned approaches try to ensure BPC either a-priori at design time or by detecting inconsistencies at run-time. In addition, there are also a-posteriori approaches that offer techniques to analyze process logs (i.e., data of already executed and finished processes) with respect to certain properties such as adhering to compliance constraints [vdAdBvD05]. In these approaches, different aspects logged during process execution can be checked, e.g., separation of duties. Doing so the a-posteriori approach reflects the need for unified compliance checking, not only a-posteriori, but also a-priori. Other approaches use constraints to synchronize between business processes of different type (e.g., between a chemotherapy and a radiation process) [Hei01]. Based on our approach, synchronization constraints can be unified and managed in combination with the other constraints in PAIS.

6 Summary and Outlook

This paper introduced a framework for the integrated representation of process constraints in PAIS. First of all, the basic concepts of the framework were introduced, i.e., *Linkage* that contains the structural pattern of the process the constraint refers to, the *Context*, i.e., the process types or instances for which the constraint is imposed, and the *Trigger Position* that equips the constraints with actions that migth be triggered by the constraint, e.g., for synchronization. Further, the framework was enriched by coping with data, time, and resource information provided within the constraints. Finally *Behavior* was specified that serves for resource attribution or exception handling, for example. Based on different real-world examples the feasibility of the representation has been shown. Further perspectives on the usage of the presented framework for consistency checks and compliance verification were discussed. Finally, it was shown how existing formalisms for specifying process constraints can be expressed by the representation framework at hand.

Currently we are working on an implementation representation framework on top of our adaptive process engine CPEE. Particular focus will be put on maintenance of the constraint base and an integrated verification component for the engine.

References

[AWW11] A. Awad, M. Weidlich, and M. Weske. Visually specifying compliance rules and explaining their violations for business processes. *Journal of Visual Languages & Computing*, 22(1):30–55, 2011.

[Bas06] Basler Ausschuss fuer Bankenaufsicht. Internationale Konvergenz der Eigenkapitalmessung und Eigenkapitalanforderungen, 2006.

[Hei01] C. Heinlein. Workflow and Process Synchronization with Interaction Expressions and Graphs. In *Proc. Int'l Conf. on Data Engineering*, page 243–252, 2001.

[HG09] Barbara Halle and Larry Goldberg. Business Process Models and Business Rules: How They Should Work Together. In *The Decision Model: A Business Logic Framework Linking Business and Technology*. Taylor & Francis, LLC, 2009.

[KLR$^+$10] D. Knuplesch, L.T. Ly, S. Rinderle-Ma, H. Pfeifer, and P. Dadam. On Enabling Data-Aware Compliance Checking of Business Process Models. In *Int'l Conf. Conceptual Modeling*, volume 6412, pages 332–346, 2010.

[LRD10] L.T. Ly, S. Rinderle-Ma, and P. Dadam. Design and Verification of Instantiable Compliance Rule Graphs in Process-Aware Information Systems. In *Proc. Int'l Conf. on Advanced Systems Engineering*, pages 9–23, 2010.

[LRGD09] T. Ly, S. Rinderle-Ma, K. Göser, and P. Dadam. On Enabling Integrated Process Compliance with Semantic Constraints in Process Management Systems - Requirements, Challenges, Solutions. *Information Systems Frontiers*, pages 1–25, 2009.

[LSG07] R. Lu, S. Sadiq, and G. Governatori. Compliance Aware Process Design. In *Proc. Int'l Business Process Management Workshops '07*, 2007.

[MRM11] Juergen Mangler and Stefanie Rinderle-Ma. IUPC: Identification and Unification of Process Constraints. *CoRR*, abs/1104.3609, 2011.

[NS07] K. Namiri and N. Stojanovic. Pattern-Based Design and Validation of Business Process Compliance. In *Int' OTM Conferences 2007, Part I*, volume 4803 of *LNCS*, page 59–76. Springer, 2007.

[PSvdA07] M. Pesic, H. Schonenberg, and W.M.P. van der Aalst. DECLARE: Full Support for Loosely-Structured Processes. In *Int'l Enterprise Computing Conference (EDOC'07)*, page 287–300, 2007.

[PT09] M. Peleg and S. W. Tu. Design patterns for clinical guidelines. *Artificial Intelligence in Medicine*, 47(1):1–24, 2009.

[SGN07] S. Sadiq, G. Governatori, and K. Naimiri. Modeling Control Objectives for Business Process Compliance. In *Proc. BPM '07*, pages 149–164, 2007.

[SM10] M. Strembeck and J. Mendling. Generic Algorithms for Consistency Checking of Mutual-Exclusion and Binding Constraints in a Business Process Context. In *OTM Conferences (1)*, pages 204–221, 2010.

[U.S03] U.S. Department of Health and Human Services. 7th Report of the Joint National Committee on Prevention, Detection, Evaluation, and Treatment of High Blood Pressure. Technical Report 03-5233, National Heart Lung and Blood Institute, 2003.

[vdAdBvD05] W. van der Aalst, H. de Beer, and B. van Dongen. Process Mining and Verification of Properties: An Approach based on Temporal Logic. In *Int'l OnTheMove Conferences*, volume 3761, page 130–147, 2005.