High-Level Requirements-Driven Business Process Compliance

No Author Given

No Institute Given

Abstract. Regulatory compliance often prioritizes adherence to explicit rules, overlooking the qualitative aspects of rule implementation. Existing goal-driven compliance approaches often focus on mapping tasks and events to enable process discovery and model repair without evaluating the qualitative factors influencing compliance goals. This study introduces a goal-driven framework that integrates requirements engineering techniques to ensure business process compliance at design time. It merges process and goal dimensions to validate the fulfillment of high-level compliance requirements, such as reliability, efficiency, and appropriateness. The framework involves modeling business requirements as a process model, capturing compliance requirements in a goal model, and synchronizing the models to determine if the intended high-level requirements are satisfied. The framework was validated through an initial implementation, demonstrating how organizations can systematically integrate and verify both procedural and high-level compliance.

Keywords: Compliance Checking · Business Process Compliance · Goal Modeling · Requirements Engineering.

1 Introduction

Emerging trends in Business Process Management have emphasized the need to move beyond mere procedural conformance and explicitly address quality attributes in compliance verification. Conventional approaches typically assess whether prescribed tasks or rules have been executed but fail to consider how these tasks align with high-level organizational objectives [12, 17]. Typically, the languages utilized to verify process compliance are either at a low-level [1, 23] or designed to describe business rules [11, 15]. However, these languages are highly implementation-oriented and closely tied to the details of a business process. They focus on depicting events rather than expressing high-level requirements. For instance, consider an invoice approval process: a conventional compliance check approach verifies that every invoice passes through the *Approve Invoice* task (procedural conformance). However, this says nothing about *the transparency of the approval decision*. If company policy requires that the reason for approval be clearly documented and accessible, merely confirming that the approval step occurred without recording its rationale is not enough to demonstrate compliance with that higher-level requirement. Therefore, incorporating

other techniques from the field of Requirements Engineering may help to capture and evaluate these qualitative dimensions, thereby enabling business analysts to verify not only "what" is done but also "how well" it is accomplished.

Prior research [2, 4, 10, 16, 18, 19] has explored linking goal frameworks and process models to manage high-level requirements, but it often focuses on compliance of the events or relies only on mapping process activities to tasks. Despite these efforts, several challenges remain. While some methodologies for systematically integrating goals and processes exist, no widely accepted standard framework has been established. Additionally, translating legal texts into structured goal/process models remains an open challenge [14], as it requires expertise in both legal and process modeling domains. Finally, ensuring traceability between goals and business processes is complex, given that goal models evolve independently from process models, making continuous alignment difficult.

To address these limitations, we propose a comprehensive, goal-driven compliance framework that operates during the design phase. By systematically translating high-level legal and business requirements into actionable intentional elements, our approach maintains traceability between evolving process models and their corresponding goals. Specifically, we establish a structured association between process activities and goal model elements, enabling analysts to incorporate intangible factors and track their fulfillment or non-fulfillment throughout the process design lifecycle. The mapping process is not automated, we leverage the expertise of compliance and process domain experts. By formalizing the compliance framework, standard reasoning techniques can be applied from transition systems to analyze the problem, ensuring that organizations can continuously align legal and strategic intentions during the design stage, thereby bridging the gap between purely structural compliance and broader, goal-oriented perspectives. Guided by this approach, we identify the following key research questions:

- *RQ1: How can we systematically combine procedural tasks and quality attributes into a unified compliance framework?*
- *RQ2: To what degree can goal modeling be integrated with process modeling approaches to allow compliance professionals and process experts to jointly evaluate and validate organizational and regulatory objectives during the design time?*

To answer these research questions, we introduce the following artifacts:

1. A unified goal-and-process framework that integrates process activities with high-level regulatory intentions.
2. A design-time compliance checking algorithm that evaluates the synchronized execution steps of process and goal models, systematically propagating task completion effects through goal hierarchies while evaluating the satisfaction of high-level requirements.

To validate these artifacts, we provide an initial implementation consisting of a practical instance drawn from the application of data confidentiality guidelines in fintech systems, demonstrating how our approach systematically captures the interplay between process conformance and goal satisfaction.

Paper structure The remainder of this paper is organized as follows: Sections 2 and 3 present the motivation and background underlying our work; Section 4 details the proposed compliance framework at design time, including the formal modeling of process and goal elements and the synchronization mechanism; Section 5 describes the prototypical implementation, and Section 6 concludes the paper with directions for future research.

2 Motivation

To illustrate how a goal model captures the qualitative dimensions of goal satisfaction, consider the excerpt below, which outlines the regulatory requirements that financial institutions must follow, and two hypothetical financial organizations, referred to as F_i and F_j, which are aligning their business continuity plans (BCP) with the ICT and Security Risk Management Guidelines. [1] To put their BCP in place, the organizations adhere to the process described below:

§81. **[P] Financial institutions should put Business Continuity Plans in place to ensure that they can [Q]** react appropriately to potential failure scenarios and that they are able to recover the operations of their critical business activities after disruptions within a recovery time objective (RTO, the maximum time within which a system or process must be restored after an incident) and a recovery point objective. (RPO, the maximum time period during which it is acceptable for data to be lost in the event of an incident).

Fig. 1: Excerpt oF ICT and Security Risk Management Guidelines

The process begins by determining the (RTO) and (RPO) for critical systems. Next, the organization configures and deploys the appropriate disaster recovery strategy, choosing between an active-active or active-passive configuration. Finally, failover tests and disaster simulations are conducted to ensure the solution meets the defined targets. If the tests are successful, the plan is finalized; otherwise, the configuration is refined until the objectives are achieved.

(a) F_i- Deploy DR

In the case of F_j, the organization already has an approved budget for establishing its (BCP), and they do not have the Cloud Infrastructure Deployed yet. The process begins by designing the disaster recovery patterns based on the appropriate Recovery Time Objective (RTO) and Recovery Point Objectives (RPO). Next, the organization configures an Active-Active Disaster Recovery (DR) strategy. Then, the DR is deployed. Finally, the deployment must be tested, and the company must review the results.

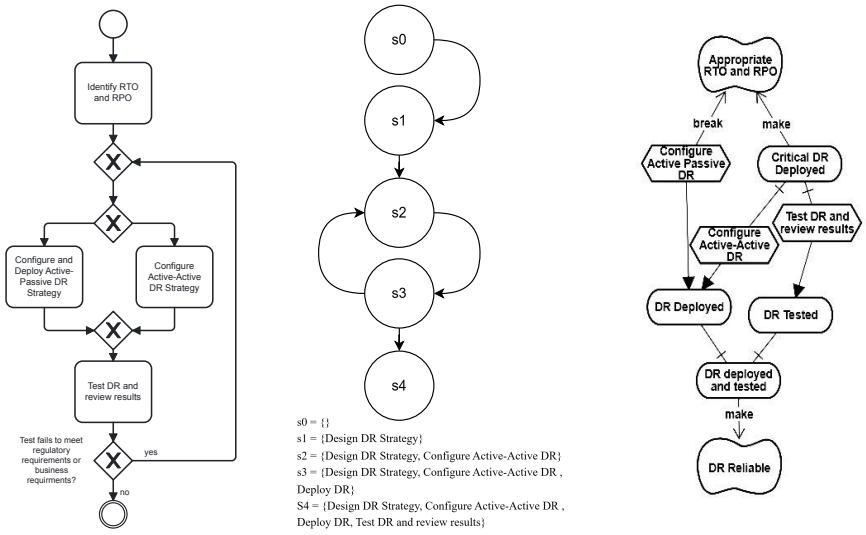
(b) F_j- Deploy DR

Fig. 2: Processes Descriptions

Quality attributes can be effectively modeled when they are measurable, making it essential to define "appropriateness" within the specific organizational context. Therefore, the quality [Q] "appropriately" is determined by the conditions [C] set in the guidelines, along with the risk appetite and cost constraints of the institution. In this case, F_i, compliance requires meeting a 5-minute RTO and near-zero data loss RPO during critical periods such as Black Friday. Given

[1] https://www.eba.europa.eu/regulation-and-policy/internal-governance/guidelines-on-ict-and-security-risk-management

the personnel capacity, an Active-Passive strategy is not an option during those periods. The qualitative aspects of both regulatory and business objectives in cloud disaster recovery (DR)[2] for critical periods are illustrated in Figure 3.



(a) F_i - $\mathcal{PM}$ for DR Configuration (b) $\mathcal{LTS}$ for F_j $\mathcal{PM}$ - DR Configuration (c) $\mathcal{GM}$ - DR in Critical Periods

Fig. 3: Operational and Qualitative Aspects of DR Strategies

As shown in Figure 3a, organization F_i may initially adopt an Active-Active Disaster Recovery (DR) strategy and meet the qualities of "*Appropriate RTO and RPO*" and "*DR Reliable.*" If Configure-Active- Passive DR activity is executed, then there is at least one reachable final state in which all the qualities can not be marked as satisfied. Therefore, Figure 3a reveals that fulfilling procedural requirements does not necessarily guarantee compliance with the qualitative aspects outlined in Figure 3c. Then, the F_i process is **not compliant** with the goal model. In contrast, the $\mathcal{LTS}$ in Figure 3b for organization F_j illustrates that from all the states, there is a reachable final state s_4 in which all the qualities are satisfied. Consequently, the process conducted by F_j (Figure 3b) is **compliant** with the goal model. While the processes depicted may vary in the level of abstraction of the goal model, the framework can also assess process and goal dimensions that are aligned at the same level of abstraction.

Qualitative attributes represent broader properties spanning multiple tasks, leading to misalignment, ad hoc assessments, and reduced traceability in compliance verification. Evaluations of qualitative compliance aspects and the inte-

[2] https://learn.microsoft.com/en-us/azure/reliability/concept-business-continuity-high-availability-disaster-recovery

gration of business process models with goals [2] remain manual mainly and lack standardization, which limits scalability and applicability. Moreover, the application domain significantly influences the relevance and type of nonfunctional requirements, and specification and analysis procedures must therefore be tailored accordingly, for example, by supporting integrated techniques for functional and qualitative requirements in domains where security aspects are classified as functional [9]. Addressing these challenges requires the systematic integration of qualitative compliance aspects into process modeling and verification to ensure procedural correctness and the achievement of high-level objectives.

3 Approach

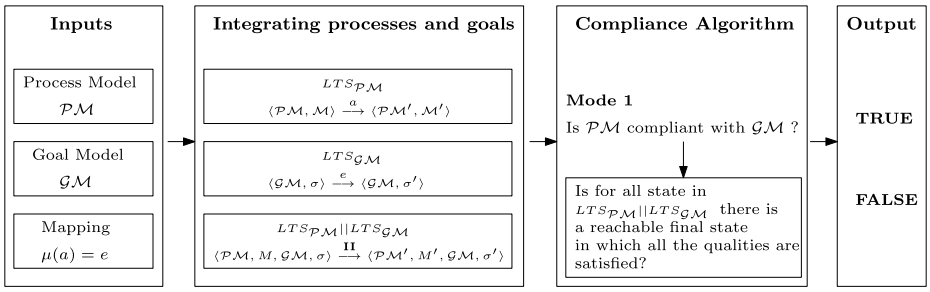


Fig. 4: High-Level Requirements Driven Compliance Checking Approach

Figure 4 provides an overview of the proposed compliance checking approach. The approach integrates three components: the process model ($\mathcal{PM}$), the goal model ($\mathcal{GM}$), and the mapping function (μ) that links process activities to intentional elements within the goal model. While the process model captures mainly operational objectives ([P]), the goal model formalizes qualitative aspects ([Q] and [C]). To systematically assess compliance, we define the labeled transition systems ($\mathcal{LTS}_{\text{gm}}$ and $\mathcal{LTS}_{\text{pm}}$) for both models, enabling a synchronized execution. The composed $\mathcal{LTS}_{\text{comp}}$ facilitates the verification of whether process execution ensures the fulfillment of compliance requirements. Compliance is determined by the check function γ, which verifies whether **for all state in the $\mathcal{LTS}_{comp}$ there is a reachable final state in which all the qualities are satisfied**. The formal properties and well-formedness criteria governing this approach are detailed in the following section.

3.1 Preliminaries

Process Models can be modeled using imperative and declarative languages, such as Business Process Model and Notation (BPMN), Petri nets, Dynamic Condition Response Graphs (DCR), and Declare. However, the framework is not

restricted to any particular notation. Therefore, we introduce a generic definition of process notation $\mathcal{A}$ (Definition 1) and a labeled transition system $\mathcal{LTS}_{\text{PM}}$ (Definition 2) that can be instantiated in different process model languages.

Definition 1. *(taken from [7].) Assume a fixed universe of actions U. A process notation is a tuple $\mathcal{A} := \langle \mathcal{P}, alph, step \rangle$ that consists of: a set of process models, $\mathcal{P}$; the labeling function, $alph : \mathcal{P} \to 2^U$; and the transition predicates $step : \mathcal{P} \times U \times \mathcal{P}$ that defines valid transitions between process model instances based on the executed activities.*

A process notation $\mathcal{A}$ is valid according to [7] if $\langle \mathcal{P}, l, \mathcal{Q} \rangle \in step$ implies both $l \in alph(\mathcal{P})$ and $alph(P) = alph(Q)$, and if also $(\mathcal{P}, l, \mathcal{Q}') \in step$ then $\mathcal{Q} = \mathcal{Q}'$, that is, *step* is action-deterministic. For non-monotonic business process model languages such as Dynamic Condition Response Graph, that modify the process model by excluding activities, Definition 1 must be extended with an additional function, $excluded : \mathcal{P} \to 2^U$.

For the sake of space, we acknowledge that a process notation $\mathcal{A}$ can be instantiated in various languages such as BPMN, DCR, and Petri nets. Assuming a business process model in some of these languages, we defined the process marking and the labeled transition system. Note that the labeled transition system follows the operational semantics of the instantiation language. For instance, in the case of BPMN, a semantics such as the one described in [5] is used.

Definition 2 (Labeled Transition System - Process Model). *A labeled transition system $\mathcal{LTS}_{\mathcal{PM}}$ is a tuple $\langle S_{\mathcal{PM}}, Act_{\mathcal{PM}}, \longrightarrow_1, s_0, F \rangle$ that consists of: a finite set of states, $\mathcal{S}_{\mathcal{PM}}$; a finite set of labels or actions, $Act_{\mathcal{PM}}$; the transition relation, $\longrightarrow_1 \subseteq S_{\mathcal{PM}} \times Act_{\mathcal{PM}} \times S_{\mathcal{PM}}$, that defines valid transitions between the states and the executed activities; the initial state, $s_0 \subseteq S_{\mathcal{PM}}$; and the set of final states, $F \subseteq \mathcal{S}_{\mathcal{PM}}$.*

Assuming a finite set of actions $Act_{\mathcal{PM}}$. Let $Act^*_{\mathcal{PM}}$ be the set of all sequences of elements of $Act_{\mathcal{PM}}$, also called *trace*. Formally, we defined the transition relation and trace as follows. For any state $s \in S_{\mathcal{PM}}$, the empty trace ϵ representing "no action" satisfies $s \xrightarrow{\epsilon} s$. For actions $a, b \in Act_{\mathcal{PM}}$ and states $s, s', s'' \in \mathcal{S}_{\mathcal{PM}}$ if $s \xrightarrow{a} s'$ and $s' \xrightarrow{b} s'$ then we write $s \xrightarrow{a,b} s''$ meaning that the trace $\theta = \langle a, b \rangle$ is the sequence of actions required to reach s'' from s. Moreover, we say that a state s'' is *reachable* from a state s and write $s \xrightarrow{*} s''$ if there exists some trace $c \in Act^*_{\mathcal{PM}}$ such that $s \xrightarrow{c} s''$

Definition 3 (Process Model Marking). *A process marking is defined as $\xi : S_{\mathcal{PM}} \to 2^X$ where $S_{\mathcal{PM}}$ represents the set of states of the process model, X is the set of process elements, and $\xi(s) \subseteq X$ represents the marking at state s (the set of flow elements currently holding tokens).*

Well-formedness criteria for a process model A process model is well-formed if for every $s \in S_{\mathcal{PM}}$ the following properties hold: *deadlock freedom* : If $s_0 \xrightarrow{*} s \Rightarrow \exists a \in Act_{\mathcal{PM}}, \exists s' \in S_{\mathcal{PM}} : s \xrightarrow{a} s'$; *option to complete*: If $s_0 \xrightarrow{*} s \Rightarrow \exists s_f \in F \subseteq$

$S_{\mathcal{PM}} : s \xrightarrow{*} s_f$; *liveness*: $s_0 \xrightarrow{*} s \Rightarrow \forall a \in Act, \exists \theta \in Act^*, \exists s' \in S : s \xrightarrow{\theta} s'$ with a executed in θ; and *boundedness:* Let ξ_0 be the initial marking and ξ any marking reachable from ξ_0. Since our process model is based on workflow nets (with token-based markings), the net is considered bounded if there exists a constant $k \in \mathbb{N}$ such that for every reachable marking ξ and every place $p \in P$, we have $\xi(p) \leq k$. This condition guarantees that the state space of the corresponding $\mathcal{LTS}_{\mathcal{PM}}$ is finite. We assume every $\mathcal{PM}$ provided as input to our algorithm is well-formed.

Goal Models provide a structured representation of how objectives should be achieved, encompassing both operational and qualitative requirements. Formally, a goal model ($\mathcal{GM}$) is defined as a simplified version of iStar [6], where tasks represent activities, goals define intended outcomes, and qualities capture high-level requirements. The hierarchical structure of goal models enables a top-down decomposition, facilitating compliance verification by systematically evaluating how process activities contribute to the fulfillment of qualitative attributes.

Definition 4 (Goal Model). *A goal model is a tuple $\mathcal{GM} := \langle IE, L \rangle$ that consists of: a set of intentional elements $IE := T \cup G \cup Q$, with T a set of tasks, G a set of goals, and Q a set of qualities; a set of intentional elements relations (also called links) $L := R \cup C$, with a set of refinement relations $R \subseteq G \cup T \times G \cup T \rightharpoonup \{And, Or\}$ and a set of contribution relations $C \subseteq G \cup T \times Q \rightharpoonup \{Make, Break\}$.*

Goal Model Evolution Goal models are effective at capturing high-level objectives and stakeholder intentions, yet they lack an inherent operational structure for execution. In contrast, process models benefit from a well-established formalism that models state evolution through transitions. To bridge the gap between these two approaches, we transform the goal model into a Labeled Transition System ($\mathcal{LTS}_{\mathcal{GM}}$). This transformation imposes a unified formal structure on the evolution of the goal model, aligning it with the operational semantics already accepted in process modeling.

Definition 5 (Labeled Transition System - Goal Model). *A labeled transition system of a goal model $\mathcal{LTS}_{\mathcal{GM}} := \langle S_{\mathcal{GM}}, IE, \longrightarrow_2, s_0, F \rangle$ is a tuple that consists of: a finite set of states $S_{\mathcal{GM}}$, where a state s is represented as $\langle \langle IE, L \rangle, \sigma \rangle$; the set of intentional elements IE; the transition relation $\longrightarrow_2$, where $\longrightarrow_2 \subseteq S_{\mathcal{GM}} \times IE \times S_{\mathcal{GM}}$ each transition in the system is labeled by an action from IE; the initial state $s_0 \in S_{\mathcal{GM}}$; and the set of final states, $F \subseteq \mathcal{S}_{\mathcal{GM}}$.*

Definition 6 (Goal Model Marking). *A goal model marking is a total function that assigns the satisfaction value to each intentional element: $\sigma : IE \rightarrow \{\top, \bot, ?\}$ where $\top$ indicates that the element is satisfied, $\bot$ that the element is not satisfied, and ? represents satisfaction status is undetermined.*

The undetermined status (?) presented above occurs in two scenarios: initial state, when the model is first initialized, every element is marked as undetermined $\sigma_0(e) =?$ for all $e \in IE$. At a later stage, if an element has not been

executed or influenced by the propagation effects of other executed elements, its status remains undefined (?).

To facilitate run-time reasoning in the goal model, we define the reachability relation $\xrightarrow{*}$ as the transitive closure of R. In our goal model, the refinement relation R is strictly top-down. This means that if $(e', e) \in R$, then e' is defined as a direct *child* of e, and no element can be both a *parent* and a *child* within the same decomposition chain. Let $Act_{\mathcal{GM}} := G \cup T$ be the union of the set tasks and goals. Formally, for any $e, e' \in Act_{\mathcal{GM}}$, we write $e \xrightarrow{*} e'$ if and only if there exists a natural number $n \geq 1$ and a trace $e_0, e_1, \ldots, e_n \in Act^*_{\mathcal{GM}}$ such that $e_0 = e, e_n = e'$ and $e_i, e_{i+1} \in R$ for all $0 \leq i \leq n$.

Definition 7 (Plan Function). *The plan function* $\pi \colon \{\mathcal{GM}\} \times Act_{\mathcal{GM}} \to 2^{Act_{\mathcal{GM}}}$ *determines which elements can be reached from* $e \in Act_{\mathcal{GM}}$ *through a finite sequence of refinement links.* $\pi(\mathcal{GM}, e)$ *is defined by* $\{e' \in Act_{\mathcal{GM}} \mid e \xrightarrow{*} e'\}$ *with the constraint that for any* $e, e' \in Act_{\mathcal{GM}}$*, if* $e' \in \pi(\mathcal{GM}, e)$*, then* $e \notin \pi(\mathcal{GM}, e')$

Well-Formedness Criteria for Goal Models A goal model ($\mathcal{GM}$) is considered well-formed if it satisfies the following conditions. First, R is acyclic and strictly hierarchical. Second, the model must be composed at least by $\mid Act_{\mathcal{GM}} \mid \geq 1$ and $\mid Q \mid \geq 1$, and there exist $e \in Act_{\mathcal{GM}}$ and $q \in Q$ such that $C(e, q) = Make$; furthermore, each goal must have a *Make* contribution link, $\forall\, q \in Q \mid l \mid \geq 1 \mid l \in C$ and $C(e, q) = Make$ ensuring traceability to higher-level goals. Third, for any intentional elements $e_1, e_2 \in Act_{\mathcal{GM}}$ and any quality $q \in Q$ for which $C(e_1, q) = Make$ and $C(e_2, q) = Break \implies \pi(\mathcal{GM}, e_1) \cap \pi(\mathcal{GM}, e_2) = \emptyset$, meaning that they do not share any common subelements. Finally, if an element e is refined by multiple elements, the link type must be the same $(e_1, e), (e_2, e) \in R \implies R(e_1, e) = R(e_2, e)$. Additionally, for every pair of elements, the number of refinement links is at most one $\mid \{l \in R \mid l = (e_1, e_2)\} \mid \leq 1$. These conditions guarantee consistency among goals and prevent conflicting contributions within the model. We assume that every $\mathcal{GM}$ provided as input to our algorithm is well-formed.

Mapping The mapping of process actions to intentional elements establishes a formal correspondence between the process model and the goal model, enabling compliance evaluation even when their levels of granularity differ.

Definition 8 ($Act_{\mathcal{PM}}$-to-$Act_{\mathcal{GM}}$ Mapping). *The mapping is a total function between process model activities and the power set of tasks and goals defined as:* $\mu \colon Act_{\mathcal{PM}} \to \mathcal{P}(Act_{\mathcal{GM}}) \cup \perp$*, where* $\perp$ *denotes that an activity is unmapped.*

For instance, an activity $a \in Act_{\mathcal{PM}}$ may be associated with a composite element such as (g, t_1, t_2), which represents that the process model activity satisfies a goal g and two activities t_1, t_2. Moreover, as process models and goals models may not share the same level of granularity, there can be some process activities

that are not mapped, meaning they do not have a corresponding element in the goal model.

Note that, based on the mapping (Definition 8) and the goal model marking (Definition 6), an element e is marked as satisfied ($\top$) either by direct execution (when $\mu(a) = ie \mid e \in ie$) or by having all its requisite sub-elements marked as satisfied. Consequently, although the goal model is hierarchically decomposed, the activation of an element does not depend solely on the satisfaction of elements of its plan, thus allowing intermediate nodes to be directly mapped and activated.

Well-Formedness Criteria for μ A valid mapping μ must satisfy two conditions: First, when a process activity $a \in Act_{\mathcal{PM}}$ maps to two or more elements, and they contribute directly or indirectly (they are part of a decomposition chain) to a quality, their contribution type must be identical. Formally, for any activity $a \in Act_{\mathcal{PM}}$ with $\mu(a) = ie$ and $|ie| \geq 2$, let $e_1, e_2 \in ie$. For every quality $q \in Q$, if there exist $e_3, e_4 \in \mathcal{P}(Act_{\mathcal{GM}})$ with defined contribution links $(e_3, q), (e_4, q) \in C$ such that $e_1 = e_3$ or $e_1 \in \pi(\mathcal{GM}, e_3)$ or $e_2 = e_4$ or $e_2 \in \pi(\mathcal{GM}, e_4)$, then it must hold that $C(e_3, q) = C(e_4, q)$. Second, for every pair of elements $e_1, e_2 \in ie$ such that $(e_1, e_2) \in R$ and $R(e_1, e_2) =$ AND it must hold for all $a \in Act_{\mathcal{PM}}$ it is not the case that bot e_1 and e_2 are in $\mu(a)$. In other words, if $\mu(a)$ contains one of these elements, that is acceptable; however, it must never contain both. We assume that every μ provided as input to our algorithm is well-formed.

3.2 Framework formalization

Instantiating the Process Model for BPMN (adapted from [22]) A BPMN model is a tuple $\mathcal{PM} := \langle F_e,\ S_{fe},\ type \rangle$. Here, F_e is a finite set of flow elements partitioned into *tasks* (T), *events* (E), and *gateways* (G), which are pairwise disjoint; $S_{fe} \subseteq F_e \times F_e$ is a finite set of *sequence-flow edges* linking flow elements; *type*: $(E \cup G) \to \{\text{start, end, AND, XOR, OR}\}$ defines *gateway/event types*.

Labeled Transition System - BPMN Recall Definition 2, we rely on a labeled transition system ($LTS_{\mathcal{PM}}$) that encodes the semantics of these workflow operators such as the one described in [5]. In this system, the transition relation is defined such that a transition $s \xrightarrow{t} s'$ occurs when a task t is enabled. During this transition, tokens are consumed from the current state and produced in the resulting state. In particular, a flow element in F_e is activated if its incoming sequence flow is satisfied, leading to a new marking $\xi(s')$.

Marking Similarly to Petri nets, BPMN employs operational semantics where markings determine the process state, and tokens traverse sequence flows to enable and trigger activities or gateways [8]. Activities and exclusive gateways fire when at least one incoming sequence flow contains a token, while parallel gateways require all incoming flows to be marked before firing. Recall Definition 3 the marking function is defined as follows: $\xi : S_{\mathcal{PM}} \to 2^{F_e}$ where $S_{\mathcal{PM}}$ represents the set of process model states, F_e is the set of flow elements in the process model, and $\xi(s) \subseteq F_e$ represents the marking at state s (the set of flow elements currently holding tokens).

Instantiating the Goal Model Recall Definition 4 the goal model $\mathcal{GM} := \langle IE, L\rangle$ captures both operational objectives and qualitative compliance requirements. The intentional elements IE. For the definitions of the transition rules, we refer to the goal model marking σ and the labeled transition system $LTS_{\mathcal{GM}}$ defined in Section 3.1 Initially, the marking assigns $\sigma_0(e) =?$ for all $e \in \mathcal{GM}$, meaning that the satisfaction of the intentional elements is undefined at the beginning. We write $\sigma(e) = \top$ to indicate that e is satisfied. A state transition, denoted as $\langle \mathcal{GM}, \sigma\rangle \xrightarrow{e} \langle \mathcal{GM}, \sigma'\rangle$, occurs when an intentional element $e \in IE$ is executed, moving the system from state s to s', updating the marking accordingly ($\sigma' = \sigma[e \mapsto \top]$). A run is a sequence $s_0 \xrightarrow{e} s_1 \xrightarrow{e} \ldots \xrightarrow{e_n-1} s_n$ where $e \in IE$ and s_{i+1} is reached from s_i via the transition relation $\longrightarrow_2$. The run is intended to complete the execution of the goal model if the final state s_n belongs to F, the set of designated final states, and if the goal model marking in s_n reflects that all the qualities have been satisfied.

Operational Semantics of the $LTS_{\mathcal{GM}}$ Let $\langle\langle IE, L\rangle, \sigma\rangle$ represent a goal model state s. The elements of the goal model ($\mathcal{GM}$) remain invariant throughout the execution, meaning that the structure of intentional elements and their relationships do not change over time. Therefore, to simplify the notation in the following rules, we represent the state $\langle\langle IE, L\rangle, \sigma\rangle$ as $\langle \mathcal{GM}, \sigma\rangle$. For the inference rules, we assume that the involved elements x and y belong to $Act_{\mathcal{GM}}$, $q \in Q$, and there exists a link $l = (x, y) \in L$, unless otherwise specified in the premise. These rules defined below formalize the conditions under which the transitions occur.

Goals and Tasks Activation rules differentiate between the execution of intentional elements and the absence of such elements: *Rule* [$P_{\text{non-ie}}$] specifies that when no intentional element is executed, the system state remains unchanged, while *Rule*[P_{ie}] describes that when an intentional element x is present in the state σ, its execution updates the marking so that $\sigma'(x) = \top$, note that the activation of x does not rely solely on the satisfaction of the sub-elements, given that we can map directly intermediate nodes.

$$\frac{}{\langle \mathcal{GM},\, \sigma\rangle \xrightarrow{\epsilon} \langle \mathcal{GM},\, \sigma\rangle} \qquad (\mathrm{P}_{non\text{-}ie})$$

$$\frac{x \in \sigma}{\langle \mathcal{GM}, \sigma\rangle \xrightarrow{x} \langle \mathcal{GM}, \sigma[x \mapsto \top]\rangle} \qquad (\mathrm{P}_{ie})$$

The *Refinement Propagation*rules define how parent elements become enabled based on the status of their direct successors: *Rule* [P_{AND}] requires that if x is refined via an AND-link will be enabled only when all its direct successors are satisfied, whereas *Rule*[P_{OR}] stipulates that a goal or task refined via an OR-link is enabled as soon as at least one direct successor is satisfied.

$$\frac{R(x,y) = \text{AND} \qquad x, y \in \sigma \quad \forall\, x :\ \sigma(x) = \top \wedge (x, y) \in R}{\langle \mathcal{GM}, \sigma\rangle \xrightarrow{y} \langle \mathcal{GM}, \sigma[y \mapsto \top]\rangle} \qquad (\mathrm{P}_{\text{AND}})$$

$$\frac{R(x,y) = \text{OR} \qquad x, y \in \sigma \quad \exists\, x :\ \sigma(x) = \top \wedge (x, y) \in R}{\langle \mathcal{GM}, \sigma\rangle \xrightarrow{y} \langle \mathcal{GM}, \sigma[y \mapsto \top]\rangle} \qquad (\mathrm{P}_{\text{OR}})$$

The *Contribution Propagation* Achievement rules govern the propagation of contributions from goals to quality attributes: *Rule* $[P_{Make}]$ ensures that if x contributes to a quality attribute y via a $Make$ contribution, satisfying x automatically satisfies y, and *Rule* $[P_{Break}]$ indicates that if x contributes to y through a *Break* link, then satisfying x results in the negation of y.

$$\frac{C(x,y) = \text{Make} \quad \sigma(y) \in \{\top, ?\} \quad y \in Q \qquad x, y \in \sigma \quad \exists\, x : \ \sigma(x) = \top \wedge (x,y) \in C}{\langle \mathcal{GM}, \sigma \rangle \xrightarrow{y} \langle \mathcal{GM}, \sigma[y \mapsto \top] \rangle} \quad (\text{P}_{\text{Make}})$$

$$\frac{C(x,y) = \text{Break}, \quad \sigma(y) \in \{\bot, ?\} \quad y \in Q, \qquad x, y \in \sigma \quad \exists\, x : \ \sigma(x) = \top \wedge (x,y) \in C}{\langle \mathcal{GM}, \sigma \rangle \xrightarrow{y} \langle \mathcal{GM}, \sigma[y \mapsto \bot] \rangle} \quad (\text{P}_{\text{Break}})$$

The *Back Propagation* rules ensure that the deactivation of intentional elements occurs only after overriding the satisfaction value of q rather than immediately when an element becomes unsatisfied. When a Make contribution should prevail, the corresponding action updates the state by marking the quality attribute as satisfied and deactivating any intentional elements that conflict with that quality, as well as their refinement chain ($[BP_{fulfill}]$). In the same way, when a Break contribution overrides a conflicting Make contribution, the action marks the quality as unsatisfied and deactivates any intentional elements associated with the Make contribution, along with their refinement chain ($[BP_{deny}]$).

$$\frac{C(x,y) = \text{Make}, \quad \sigma(y) = \bot, \quad y \in Q, \qquad x, y \in \sigma, \quad \exists\, x : \ \sigma(x) = \top \wedge (x,y) \in C}{\langle \mathcal{GM}, \sigma \rangle \xrightarrow{y} \langle \mathcal{GM}, \sigma' \rangle} \quad \text{with } \sigma' = \sigma\Big[y \mapsto \top, \ \forall x \text{ with } C(x,y) = \text{Break} : x \mapsto \bot, \ \forall z \in \pi(\mathcal{GM}, x) : z \mapsto \bot\Big] \quad (\text{BP}_{\text{fulfill}})$$

$$\frac{C(x,y) = \text{Break}, \quad \sigma(y) = \top, \quad y \in Q, \qquad x, y \in \sigma, \quad \exists\, x : \ \sigma(x) = \top \wedge (x,y) \in C}{\langle \mathcal{GM}, \sigma \rangle \xrightarrow{y} \langle \mathcal{GM}, \sigma' \rangle} \quad \text{with } \sigma' = \sigma\Big[y \mapsto \bot, \ \forall x \text{ with } C(x,y) = \text{Make} : x \mapsto \bot, \ \forall z \in \pi(\mathcal{GM}, x) : z \mapsto \bot\Big] \quad (\text{BP}_{\text{deny}})$$

Mapping This approach maps process activities to leaf and intermediate nodes in the goal hierarchy, aligning goal and process models. The model includes goals, tasks, and soft-goals, with hierarchical dependencies enabling compliance evaluation through goal satisfaction propagation. Qualitative requirements are quantifiable, ensuring structured and verifiable compliance assessment. Consequently, organizations can compare multiple process models against predefined goal models that share common qualitative requirements and activities, facilitating objective validation. For example, an insurance company aiming to *obtain customer consent* may define multiple means to achieve this goal, such as *consent by phone* or *physical consent*. If a process model includes only the activity *Obtain Electronic Consent*, this may still satisfy the overarching goal. Currently, activity mapping is performed manually, leveraging the expertise of process

3.3 Compliance Checking

The compliance assessment involves verifying whether the execution of process activities ensures the fulfillment of high-level compliance requirements represented in the goal model. By leveraging a synchronized Labeled Transition System (LTS) (Definition 9) that integrates both process and goal model states, we systematically assess whether all quality attributes are eventually satisfied. This section formalizes the compliance verification mechanism and defines the operational semantics governing the transition between system states.

Definition 9 (Labeled Transition System - PMGM). *A Labeled transition system is a tuple* $LTS_{comp} := \langle S_{comp}, IE, \xrightarrow{\|}, s_0, F \rangle$ *for compliance verification of high-level requirements is a composition of the LTS of the process and goal model* $LTS_{pm} \| LTS_{gm}$. *The* LTS_{comp} *consists of: a set of states* $S_{LTS_{comp}}$, *with a state* $s = \langle \mathcal{PM}, \xi, \mathcal{GM}, \sigma \rangle$; *IE the set of intentional elements; a transition relation* $\xrightarrow{\|} \subseteq \mathcal{S}_{LTS_{comp}} \times IE \times \mathcal{S}_{LTS_{comp}}$, *that defines how the system moves from one state to another; an initial state* $s_0 = \langle \mathcal{PM}, \xi_0, \mathcal{GM}, \sigma_0 \rangle$; *and a set of final statesF* $\subseteq \mathcal{S}_{comp}$.

The operational semantics of the transition relation $\xrightarrow{\|}$ is defined by the following rule:

$$\frac{\langle PM,\, \xi\rangle \xrightarrow{a}_1 \langle PM',\, \xi'\rangle \qquad \langle \mathcal{GM},\, \sigma\rangle \xrightarrow{e}_2 \langle \mathcal{GM},\, \sigma'\rangle}{(PM,\, \xi,\, \mathcal{GM},\, \sigma) \xrightarrow{e}_{\|} (PM',\, \xi',\, \mathcal{GM},\, \sigma')} \quad \text{with } \mu(a) = ie \text{ and } e \in ie.$$

(Synchronized-Step)

A transition is considered valid if it preserves the properties of the *step* function, as defined in Section 4.1. Specifically, this entails that for any transition $\big(\mathcal{PM}, \xi, \mathcal{GM}, \sigma\big) \xrightarrow{\|} \big(\mathcal{PM}', \xi', \mathcal{GM}, \sigma'\big)$, the transition relation in the composed system must inherit the determinism and consistency constraints imposed by *step*. This ensures that state evolution adheres to the defined execution semantics, preserving action determinism, process consistency, and the integrity of state updates within both the process and goal models. Consequently, compliance verification remains aligned with the structured evolution of the process execution. To determine whether a process model aligns with compliance requirements, we define a compliance check function (Definition 10). This function evaluates if, for every $s \in \mathcal{LTS}_{comp}$, there is a reachable final state in which all the qualities are satisfied.

Definition 10 (Compliance Check function). *The compliance checks function is defined as follows:*

$$\gamma = \begin{cases} \textit{true}, & \textit{if } \forall s \in S_{LTS_{comp}},\ \exists s' \in F \subseteq S_{LTS_{comp}} \\ & \quad \textit{such that } s \xrightarrow{*} s' \textit{ and } \forall q \in Q,\ \sigma_{s'}(q) = \top, \\ \textit{false}, & \textit{otherwise.} \end{cases}$$

Having established the formal rules for process execution and goal satisfaction, we now present the complete compliance-checking algorithm.

Algorithm 1: Compliance Checking Algorithm

Input: Goal Model $GM = \langle IE, L \rangle$, Process Model $\mathcal{PM} = \langle F_e,\ S_f,\ type\ \rangle$, Mapping μ
Output: `true` if for every state there is a reachable final state in which all the qualities are satisfied $q \in Q$; `false` otherwise

Initialize $\sigma(e) :=?$ for all $e \in IE$, $\xi_0 := [s_i = 1, \text{all other places} = 0]$;
Define initial states: goal model $s_0^{GM} = \langle GM, \sigma_0 \rangle$, process model $s_0^{PM} = \langle PM, \xi_0 \rangle$, LTS_{comp}, where $s_0^{comp} = \langle GM, \sigma_0, PM, \xi_0 \rangle$;
`// PART 1: State Space Construction`
foreach *state* s_i *in* LTS_{comp} **do**
    foreach *transition* $s_i \xrightarrow{e} s_{i+1}$ *in* LTS_{comp} **do**
        (PM) update the marking of PM
        (PM) update the process model
        (GM) update the goal model marking (apply transition rules) to generate s_{i+1}
        `// Update satisfaction of individual elements`
        `// Update goals and tasks satisfactions based on refinements`
        `// Update quality satisfactions based on contributions`
        `// Override quality satisfaction and deactivate conflicting elements based on backpropagation rules`
    end
end
`// PART 2: Compliance verification`
foreach *state* s_i *in* LTS_{comp} **do**
    if *there is no final state* s_f *reachable from* s_i *such that* $\forall q \in Q,\ \sigma_{s_f}(q) = \top$ **then**
        return *Not compliant*;
    end
end
return *Compliant*;

The algorithm first constructs the state space of the composed Labeled Transition System $\mathcal{LTS}_{comp} = \langle S_{comp}, IE, \rightarrow, s_0, F \rangle$ by iteratively applying the transition rules. Its complexity is $O(n)$ where $n = M \times K$, with $M = |S_{GM}|$ denoting the number of states in the goal model and $K = |S_{PM}|$ denoting the number of states in the process model. In the second part, compliance is verified by ensuring that for every state $s \in S_{comp}$ there exists at least one reachable final state $s' \in F$ ($s \xrightarrow{*} s'$) such that $\forall q \in Q,\ \sigma_{s'}(q) = \top$. Termination is guaranteed by the finiteness of $\mathcal{LTS}_{comp}$. Note that while $\mathcal{LTS}_{GM}$ is always finite, the finiteness of $\mathcal{LTS}_{PM}$ holds only under specific properties of the process model, such as one-boundedness in the case of Petri nets.

The algorithm is modular and extensible, allowing improvements or adaptations to be incorporated according to the specific application domain or process analysis need. For instance, it can be easily modified based on a set of traces that determine which of these are compliant with the goal model, and the mapping can be redesigned and validated by legal and process experts.

4 Prototypical Implementation

We conducted an initial implementation of our algorithm, taking a fictitious Fintech entity called "FT" acting as the data controller, which must manage cus-

tomer data in full alignment with the requirements outlined in GDPR Articles 12-22. Ten BPMN process models were adapted to the fintech context, for each compliance pattern (based on [1]). For the qualitative aspects, the GDPR articles were analyzed to extract the associated qualitative attributes such as *clear*, *transparent*, *accessible*, *timely*, *comprehensive*, *usable*, *fair* and *explainable*. The operationalization of these qualitative elements was performed using the data preparation method proposed in [9]. In total, 6 Goal Models were modeled to represent all the qualitative aspects. There is a smaller number of goal models for process models because several articles share qualitative aspects, meaning that a single goal model can be applied to multiple articles. Subsequently, the mapping of process activities to these qualitative elements was validated by synchronizing the operational process models with a goal model that encapsulated the qualitative compliance requirements. The evaluation demonstrates that all ten models were compliant at design time. However, a partial evaluation was conducted on three of the models (Article 12, 16, and 19), given the expressiveness limitations of our process models, which do not support the representation of timer events; consequently, the qualitative aspect of timeliness was not included in the evaluation.

The prototype was developed in Python3, taking as input a set of business process models (BPMN file), the goal model (JSON File), and the mapping (CSV File). It provides two evaluation options: the first assesses if the process model is compliant with the goal model by analyzing the composed Labeled Transition System, while the second simulates execution traces to determine which satisfies the high-level compliance requirements. The system included an interactive Jupyter notebook for manual inspection and analysis of process execution.

5 Related Work

Three decades of formal methods in Business Process Compliance [17] shows that process compliance has primarily concentrated on operational goals from computer science methodologies and leaves qualitative approaches largely underexplored. Nevertheless, they do not provide the necessary level of abstraction to involve legal experts, and consequently, there is no universally accepted benchmark (by both legal and process communities) to compare compliance outcomes. For instance, by design, a process can meet technical data privacy requirements by using the security extensions for BPMN. However, the process may still fail to ensure that clients fully understand and consent to data usage.

Assess qualitative compliance aspects encompass methods such as impact metrics that leverage historical data [24]; model-driven techniques utilizing UML and BPMN to represent processes and nonfunctional requirements for enhanced safety and privacy [13]; formal verification, particularly for scenarios where safety and security are critical [3]; simulation techniques that use virtual environments to uncover potential compliance failures [23]; and formal integrations with goal frameworks are employed to align stakeholder intentions with observed behaviors for process repair, discovery, or agent-based simulation [10, 20, 21]. From the

perspective of integrating business process models and organizational goals, [2, 16,18], several difficulties persist, maintaining traceability between evolving goal models and business processes remains a significant challenge as the two often evolve independently, which hinders continuous alignment.

To address these issues, we first extract business and compliance rules and operationalize the qualitative requirements following the data preparation method suggested in [9]. Next, we synchronize the evolution of process and goal models to evaluate the satisfaction of qualitative aspects. This approach ensures traceability from abstract goals and facilitates the verification that the business process models developed during design fulfill high-level qualitative requirements.

6 Conclusion and Future Work

This work demonstrates that integrating goal models from requirements engineering into compliance verification significantly enhances the traceability of qualitative compliance aspects. By explicitly capturing high-level requirements as intentional elements, our approach ensures that compliance evaluation goes beyond procedural correctness, addressing the qualitative dimensions that are often overlooked in traditional conformance checking. Furthermore, this methodology enables the validation of different process models against the same goal model, provided they share similar qualitative objectives. This adaptability allows organizations to systematically assess compliance across multiple processes while maintaining consistency in regulatory and business goal enforcement.

Improving traceability and adaptability ensures that it can be readily modified to accommodate diverse contexts and evolving requirements. our approach facilitates a more structured and scalable compliance assessment. Future work will focus on empirical validation in real-world regulatory environments, as well as the development of runtime monitoring mechanisms to extend compliance verification beyond design-time evaluation. Additionally, we aim to refine the adaptability of goal models to support more complex regulatory scenarios, ensuring that compliance assessment remains flexible and context-aware across diverse organizational settings.

References

1. Agostinelli, S., Maggi, F.M., Marrella, A., Sapio, F.: Achieving GDPR Compliance of BPMN Process Models. Springer (2019)
2. Amyot, D., Akhigbe, O., Baslyman, M., Ghanavati, S., Ghasemi, M., Hassine, J., Lessard, L., Mussbacher, G., Shen, K., Yu, E.: Combining goal modelling with business process modelling: Two decades of experience with the user requirements notation standard. Enterprise Modelling and Information Systems Architectures (2022)
3. Argyropoulos, N., Mouratidis, H., Fish, A.: Attribute-based security verification of business process models. In: 2017 IEEE 19th Conference on Business Informatics (CBI) (2017)

4. Cardoso, E.C.S., Santos, P.S., Almeida, J.P.A., Guizzardi, R., Guizzardi, G.: Semantic integration of goal and business process modeling. In: Proceedings of the 2010 International Conference on Enterprise, Business-Process and Information Systems Modeling (BPMDS/EMMSAD) (2010)
5. Corradini, F., Polini, A., Re, B., Tiezzi, F.: An operational semantics of bpmn collaboration. In: Lecture Notes in Computer Science. Springer (2015)
6. Dalpiaz, F., Franch, X., Horkoff, J.: istar 2.0 language guide (2016)
7. Debois, S., López, H.A., Slaats, T., Andaloussi, A.A., Hildebrandt, T.T.: Chain of events: Modular process models for the law. In: Dongol, B., Troubitsyna, E. (eds.) Integrated Formal Methods (2020)
8. Dijkman, R.M., Dumas, M., Ouyang, C.: Formal semantics and analysis of BPMN process models using petri nets. In: International Conference on Conceptual Modeling (ER 2008). Lecture Notes in Computer Science (2008)
9. Eckhardt, J., Vogelsang, A., Fernández, D.M.: Are "non-functional" requirements really non-functional? an investigation of non-functional requirements in practice. In: Proceedings of the 38th International Conference on Software Engineering (2016)
10. Ghasemi, M., Amyot, D.: From event logs to goals: a systematic literature review of goal-oriented process mining. Requirements Engineering (2020)
11. Governatori, G.: The regorous approach to process compliance. In: EDOC Workshops. pp. 33–40. IEEE (2015)
12. Hashmi, M., Governatori, G., Lam, H., Wynn, M.: Are we done with business process compliance: state of the art and challenges ahead. Knowledge and Information Systems (2018)
13. Köpke, J., Meroni, G., Salnitri, M.: Designing secure business processes for blockchains with secbpmn2bc. Future Generation Computer Systems (2023)
14. López, H.A.: Challenges in legal process discovery. In: ITBPM@ BPM (2021)
15. López, H.A., Debois, S., Slaats, T., Hildebrandt, T.T.: Business process compliance using reference models of law. In: International Conference on Fundamental Approaches to Software Engineering. pp. 378–399 (2020)
16. López, H.A., Massacci, F., Zannone, N.: Goal-equivalent secure business process re-engineering. In: Service-Oriented Computing - ICSOC 2007 Workshops (2009)
17. López, H.A., Hildebrandt, T.T.: Three decades of formal methods in business process compliance: A systematic literature review (2024)
18. Markovic, I., Kowalkiewicz, M.: Linking business goals to process models in semantic business process modeling. In: EDOC (2008)
19. Montali, M., Torroni, P., Zannone, N., Mello, P., Bryl, V.: Engineering and verifying agent-oriented requirements augmented by business constraints with $\mathcal{B}$-tropos. Autonomous Agents and Multi-Agent Systems (2011)
20. Morandini, M., Penserini, L., Perini, A.: Operational semantics of goal models in adaptive agents. In: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 1 (2009)
21. van Riemsdijk, M.B., Dastani, M., Winikoff, M.: Goals in agent systems: a unifying framework. In: AAMAS (2) (2008)
22. Van Gorp, P., Dijkman, R.: A visual token-based formalization of BPMN 2.0 based on in-place transformations. Information and Software Technology (2013)
23. Varela-Vaca, A.J., Gómez-López, M.T., Morales Zamora, Y., M. Gasca, R.: Business process models and simulation to enable GDPR compliance. International Journal of Information Security (2025)
24. Workneh, T.C., Sala, P., Rizzi, R., Cristani, M.: Business process compliance with impact constraints. Information Systems (2025)