Semantic Event-Based Decision Management in Compliance Management for Business Processes

Marwane El Kharbili¹ and Nenad Stojanovic²

¹ ARIS Research, IDS Scheer AG, Altenkesseler Str. 17, D-66115 Saarbrücken, Germany. Marwane.Elkharbili@ids-scheer.com
² FZI Karlsruhe, Haid-und-Neu-Str. 10-14, 76131 Karlsruhe, Germany. Nenad.Stojanovic@fzi.de

Abstract

Compliance is a critical enterprise management concern, particularly in business process-oriented organizations. Compliance measures are often implemented using policies, of which decision-making requires high independence from implementation logic and high flexibility. This paper motivates the use of an event-centric approach for modeling policy decision-making. We introduce an event ontology part of a policy and rules ontology for compliance modeling and enforcement. The policy and rules ontology is able to take decisions depending on the compliant state of an enterprise model. In our work, events are used to support modeling complex decision-making patterns, propagation and controlling (decision-making monitoring and analysis). This paper shows how event-oriented decision making architectures can be built for compliance management and integrates such an architecture in a platform for semantic business process management.

1. Introduction

Compliance management has been identified as one core topic of research for making business process management (BPM) the enabler for more efficiency and business sustainability. In [1], the need for compliance management in BPM is motivated and the use of policies and rules for compliance modeling and enforcement is proposed. Additionally, a framework for compliance management in BPM is introduced and its architecture discussed in [1]. In [2], the authors explain how using policies and rules together with domain policies can be used for modeling compliance. In this framework, policies are used for modeling compliance measures as decisions and rules encompass the logic attached for making these decisions, depending on the context of policy enforcement. Rules are also responsible for deciding which action needs to be taken and by which entity under jurisdiction of a policy.

In a BPM-oriented enterprise, conceptual level artifacts such as business activities (EPC [6] functions) and organizational aspects are directly linked to executable artifacts such as executable business processes (BPs, e.g. written in BPEL [5]). Actions governed by compliance management policies can be modeled as part of business process models (e.g. as calls to BP activities). Hence, putting a layer of business policies on top of conceptual BP modeling layers leads to the ability of controlling enterprise behavior by constraining the possible executions of BPs (as shown by Figure 1). Nevertheless, policy decisions need to be linked to underlying layers (i.e. actions modeled respectively taken at the levels of conceptual BP models respectively) in order to be undertaken. Using events as the linking element is one possible approach since events can be modeled on the different layers of a BPM framework (as is the case in business process modeling notations such as EPC [6], or in process/workflow execution frameworks which generate event for each execution step).

Moreover, modeling decisions is still made hard for all classes of decisions which cannot be directly executed by a rule directly linked to a policy. We call these decisions simple decisions. The other classes of decisions, called composite decisions, which may request calling and executing several rules not directly linked to the policy making the decision or may require propagating a decision executed by a rule to other dependent rules for further processing. Additionally, in a policy management framework, further concerns such as policy speech acts (defined in [7] as policy interactions, e.g. delegation, revocation, which allow for dynamic modification of policies) require realization mechanisms. Policy speech acts are used to allow policies to interact with each other and proceed to changes to the policy model and are defined in existing policy frameworks (such as Rei [7] with its Rei language, other main policy languages such as K AoS or Ponder are cited in [2]). For example, a policy may need to delegate taking a decision to another policy in certain cases. This implies that the policy-based compliance framework must provide communication mechanisms that are capable of transmitting semantics (e.g. delegation semantics) to policy and rule instances. These observations motivate looking for ways of integrating the management of these three aspects within a framework for BPM compliance management.

This work places itself in the particular context of semantically-enabled BPM, which introduces more
richness of descriptions (using ontologies as semantic formalisms) into BPM in order to achieve higher mechanization [8]. We propose an ontology for modeling complex events for decision support in compliance management for SBPM. This paper also proposes a method for event-based compliance management.

The rest of this paper is organized as follows: Section 2 introduces an approach for modeling composite policy decisions using events. Section 3 introduces our event ontology, elaborates on design issues and provides modeling examples. It is followed by Section 4, where we propose an architecture for using the event ontology in compliance management. Related and future work is discussed in Section 5, and Section 6 contains final comments.

2. Policy-Based Decision Management

In order to understand the use of events in our compliance management framework, some explanations about the concrete role of policies and rules are necessary. As it can be seen on Figure 1, we consider 4 distinct layers when studying enterprise models. The top layer is a governance layers and contains aspects of the enterprise which are necessary for controlling behavior and managing change (change in regulations for instance, and its impact on the enterprise model). We seek to model compliance measures (i.e. the constraints defined by an enterprise for complying with regulations) on this layer using policies which are able to be enacted on business process models (lying on layer 2 of Figure 1). In [9], this approach to governing enterprise behavior is introduced, which is based on a ontology which allows to define deontic and alethic modalities for policies, and separates between the definition of policies and business rules. Policies are managed on a level different from business rules, in that policies do not specify the concrete logic necessary for enforcing them, thus staying independent from compliance enforcement contexts (i.e. technology, systems or languages available). Here, contexts are not formally defined yet by the ontology, which is a research gap tackled by other works [4, 10] which future versions of the ontology will have to consider.

However, several issues appear with regard to limited expressiveness (of the policy and rules ontology) when dealing with high flexibility in modeling compliance enforcement and dynamic compliance management. First of all, compliance enforcement can be done on the level of conceptual BP models, by checking the latter against constraints defined by business policy and rule instances defined in the compliance ontology (See Figure 2). Another layer of Figure 1 where compliance can be enforced is layer 3: executable processes. This can be done by dynamically checking policy and rule constraints on BP instances being executed. In both layers (i.e. design-time or run-time compliance checking/enforcement), policies may be activated, only after having been activated can a policy be checked/enforced. Policies may be first activated when the BP models present some special properties (can be modeled as a context), or be activated by some other policy, in which case we say that policy A “activated” policy B. Also, policies can delegate their task to some other policy using a formal “delegation”. Another case if policy “revocation”, where a policy retracts a “delegation” it had made before to another policy, or policy “cancellation”, which occurs when a policy decides to call off or neutralize a policy invocation it has made before. These aspects of policy management are called “policy speech acts” and are defined in the Rei framework for policy management [7] and included in the policy ontology described in [9].

These observations are also true for business rules. The semantics of rule modeling in our policy ontology are defined such that a policy can be “implemented” using a composition of several rules. For example, a certain security policy for financial transactions in a company may require calling and executing three different business rules each checking and enforcing a specific logical unit of the regulation enforcement modeled by the policy. Additionally, a business rule definition, as it is hidden for the policy that uses it, may be a composition of business rules itself, as a mean of introducing more flexibility in the definition of rules. That implies that the policy ontology offers the possibility to declaratively specify which rules can be called by which others. Imagine a policy and rules repository with hundreds of policy and rule definitions (as ontology concept instances). A policy P definition specifies that P is implemented by a set of business rules, of which rule R1 is one element. Rule R1’s implementation changes with time as it needs to check information delivered by all security firewalls available in the enterprise’s IT infrastructure. In order to stay independent from the increasing complexity of the IT infrastructure (since new firewalls are always introduced and old ones replaced), the policy and rules ontology has to allow declaratively specifying which business rules available in the rules repository have to be called by rule R1 in order to fulfill its
duty. In the current state of the ontology, this is not possible.

In our approach, we separate between decision-making and action-taking. Decisions are made by policies and auxiliary rules and can request other rules to make other decisions and/or request a responsible entity to execute some action. Our goal is to allow modeling complex decision-making by using events as a mean of linking decision-making units (business rules) to other decision-making units. In the following, we motivate this use of events by giving a use case scenario and explain our approach to realizing this. We then show how our approach integrates in the SUPER case scenario and explain our approach to realizing this.

Fig3. Decision-Making Tree: Policies, Rules, decisions, Actions.

Let us take the example of a security rule for intrusion detection. A system is under jurisdiction of a security policy P. As one person accesses a resource of system S, an intrusion detection rule (called IDR) fires and decides that the person is an intruder. Another rule (called NF), which decides which person should be notified about this intrusion and another rule (called IDP), which decides on further actions that need to be made depending on the resource accessed by the intruder should also fire. Rules IDP and NF are dependent on rule IDR. Using events, we can easily link rule IDR to the NF and IDP rules. This can be done by making rule IDR generate a well defined event and subscribing rules NF and IDP (configuring NF and IDP to listen) to this event. The event carries among other information the ID of the rule that has fired it and the protected resource being accessed. Events become thus the link that provides declarative automation is modeling complex decision-making in our compliance framework. In [3], a survey of event-driven architecture (EDA) is given that also proposes event modeling for reacting to situations and making decisions. In [4], the authors propose a logic based formalism for modeling events in an EDA. Both works recognize the same issues as we do. We think that this idea can be easily generated to all rule frameworks for modeling complex decisions.

3. Extending the Policy and Rules Ontology with Events

Next, we introduce a core ontology for modeling events for policies and rules (BPREF) of the compliance framework. Central concepts are the rule and policy concepts (Figure 4). The concepts distinguished here are meant to be generic in nature in order to be used for a wide range of applications. In particular, implementation aspects such as the logic formalism or rule/event language to be used are left out of this core ontology, and need to be described in extending ontologies.
Policies are used to model compliance measures and rules contain the necessary logic for enforcing these policies. A policy acts on a subject which itself can occur in a business process (e.g. artifact such as BP activity), or can be any type of resource (e.g. role individual or network printer) supported by the policy. Subjects have states and belong to an event scope, which is the set of subjects which can produce/be influenced by this event. For example, a resource access event is fired when the state of a resource changes from unaccessed to accessed. Rules and policies also belong to a logical event scope, which is the set of policies and rules which can trigger or are activated (policy)/invoked (rule) by this event. The input event type is used by rules as an invocation input and transport information necessary for the rule to correctly execute. The output event type is used as a mean to communicate to the compliance framework information about rule execution and transport instructions on how to further conduct the process initiated by the policy having been activated. An event can itself fire other events if necessary. An event can be itself implemented as one of two types: (i) Complex Events and (ii) Event Streams.

These both approaches to event processing are identified and implemented by existing major industrial complex event processing (CEP) solutions. Events can be event expressions which combine atomic events in logical events using Boolean operators (AND, OR, NOT, XOR) to express complex events. We give in that simply using Boolean operators for complex events is not expressive enough. First of all, complex events may be defined not only using event occurrences (e.g. \( Ev_1 \text{ AND } (Ev_2 \text{ OR } Ev_3) \)) but also on the meta-data transported by these events (e.g. \( Ev_1.\text{Att}_1 \text{ == Value}_1 \text{ AND } Ev_2.\text{Att}_2 \text{ == Value}_2 \)). Complex events could thus be a combination of event occurrences and rules expressed on the state of the system modeled using event meta-data. This shows the high inter-dependency between events and rules, as events are used to model complex business decisions, and other rules are used to model complex events. Secondly a complex event could be itself a tree of logical expression of events, with each event being at the same time an event and a node that has a sub-tree of events attached to it. This view maps directly to figure 1 where decision nodes could at the same time be business rule invocation and a call to some action taking entity. This means that the decision-making tree example given in figure 1 could be implemented using a complex event in the form of an event tree, with rules attached to each node which contain policy enforcement logic. Each node in the tree is actually a Boolean expression of the events in its sub-tree. Event patterns form reusable sets of complex events expressed in generic way, possibly using place-holders for both event types in Boolean event expressions and rules on event meta-data, as well as pre-defined event trees.


Disposing of an ontology isn’t enough for enabling real use of events for compliance management. On Figure 5, we show how a part of the architecture of the SBPM framework SUPER which allows putting our events ontology to use. SUPER is a European project seeking to establish a reference framework for SBPM, the vision of which is stated by Hepp et al. in [8]. SUPER defines a tack of ontologies [11] for modeling all aspects related to enterprise business process management. The policy and rules ontology [9] is part of this stack of ontologies, as is the core events ontology introduced in this work. The language used for describing SUPER ontologies is called WSMO (Web Service Modeling Language) [15] stems from a framework for semantic web services called WSMO [16]. At the center of the architecture shown on Figure 5, are the set of ontologies regarded. First of all the BPMO (Business Process Modeling Ontology) is the main ontology used for modeling BPs in SUPER [12], while sBPEL [13] is an ontological representation of the BPEL
standard for executable business processes [14]. The cylinder labeled “organizational ontologies” is a set of ontologies used for modeling organizational aspects of the enterprise such as roles, goals, and resources. On the middle left part of Figure 5, we distinguish three ontologies: (i) The business policy ontology, (ii) the business rules ontology and (iii) the events ontology which were already introduced. The semantic service bus is a middleware technology able to link several components of the SUPER framework available as semantic web services together and to the ontologies.

![Fig 5. An Architecture for extending SBPM with Event- and policy-based Compliance Management.](image)

The top part of the figure shows the “semantic compliance checking engine (SCE)”, which is a software component implementing series of compliance checking algorithms, which description is out of the scope of this paper. The SCE is able to use inference engines for reasoning on several ontological descriptions available in the ontologies regarded by compliance management in SUPER: the BPMO, sBPEL, Organizational Ontologies, BPRO and BPREO. The BPMO and the BPREO are written in a subset language of WSML called WSML-Flight⁶, with rule modeling abilities. It uses a intermediary software component for policy management which makes use of a semantic policy management engine (Rei [7]) and for this needs to use a translation component called "'". This policy management engine uses an additional inference engine for inferring on semantic rule descriptions called “semantic rule inference engine” capable of working directly on WSML-flight descriptions of business rules. This stack of software components and inference engines makes use of a reasoner called IRIS⁷ (Integrated Rule Inference Engine). IRIS requires using an interface for reasoning on WSML descriptions called WSML2Reasoner⁸. Finally, the event management components needed are to be seen on the bottom layer of Figure 5:

(i) Semantic Event Pipe and Converter (SEPC): This is the component that is responsible for processing semantic events fired by both the policy and rule ontologies. It is capable of transforming semantic event descriptions arriving into a format that is processable by a custom complex event processing engine. This requires trimming semantic events of non-relevant information for the CEP engine, and both processing this semantic information if it is relevant while forwarding adequate representations of events to the CEP engine.

(ii) CEP Engine (e.g. Esper): a complex event processing engine needs to be used by our architecture for providing the necessary CEP functionalities.

(iii) Event Processing Rules: this is where a representation of the rule-based description of events processing can be stored. Using such a component makes sense when using semantic rules for describing how events need to be processed, and then map these semantic event-processing rules to rules that can be interpreted by the CEP engine. It is still an aspect that is not regarded by the core events ontology for events we introduced though.

Moreover, An important piece of architecture is still missing on Figure 5. A component is needed in order to implement actions required by the CEP engine. This would be the action-taking part of our approach. As actions in SUPER will be taken by semantic elements (i.e. semantic web services (SWS)) implementing some functionality described in a dedicated ontology (e.g. the business functions ontology part of the organizational ontologies), the CEP engine needs to be wrapped in a component that enables it to directly invoke these SWS. Similar concerns have already been tackled by SBPM research [17]. Events

---

⁶ WSML-Flight is the WSML-Core language enriched with logical programming with non-monotonic negation. Another WSML subset is the WSML-Rule language. WSML-Flight is based on a logic-programming variant of F-Logic. WSML-Rule is modeling abilities. It uses a intermediary software component for policy management which makes use of a semantic policy management engine (Rei [7]) and for this needs to use a translation component called "'". This policy management engine uses an additional inference engine for inferring on semantic rule descriptions called “semantic rule inference engine” capable of working directly on WSML-flight descriptions of business rules. This stack of software components and inference engines makes use of a reasoner called IRIS⁷ (Integrated Rule Inference Engine). IRIS requires using an interface for reasoning on WSML descriptions called WSML2Reasoner⁸.


⁸ http://tools.sti-innsbruck.at/wsml2reasoner/.
need to be taken into consideration by an external software component, preferably the SCE, in order to be able to use events for compliance enforcement. Let us take the example of a policy P calling a set of business rules. using the ontology for business policies and the events ontology, the SCE knows it has the send a set of instances of events. After having been processes by the CEP engine, the signification of the result of this processing needs to be transformed into either a decision or an action. This can be done by the SCE, since the necessary information for doing it will be stored in an extension of the events ontology for describing actions, and the SCE is able to infer on these descriptions.

5. Future Work & Concluding Remarks

Our compliance management ontology stack will need to be extended for describing event processing rules, action-taking modelling, description of contexts. Additionally, We will seek to design a prototype application for modelling compliance policies using complex event processing, using policy and rule descriptions of regulations by relying on a formal model such as described in [9] or in [18], [19]), as well as using events for modelling compliance enforcement.

Our work seeks to build a compliance management framework for semantic business process management. This framework makes use of policy and rule definitions and relies on the ability to both model and enforce decisions. We discussed why event modelling is a necessary condition for attaining more flexibility in enabling compliance checking and enforcement. For this purpose, a core event ontology for supporting policy-based decision management was introduced. Our work has the aim of contributing to enable practitioners of semantic BPM to dispose of a framework for compliance management which allows them to model complex compliance policies with real-time checking abilities. Such a framework can be used for example for building intelligent business activity monitoring and analysis solutions, capable of reacting to custom defined situations.

References