A Formal Approach for Internal Controls Compliance in Business Processes

Kioumars Namiri[1], Nenad Stojanovic[2]

[1] SAP Research Center CEC Karlsruhe, SAP AG, Vincenz-Prießnitz-Str.1
76131 Karlsruhe, Germany
Kioumars.Namiri@sap.com

[2]FZI Karlsruhe, Haid-und-Neu-Str. 10-14
76131 Karlsruhe, Germany
Nenad.Stojanovic@fzi.de

Abstract. Regulatory compliance requirements in the area of Internal Controls such as Sarbanes Oxley Act force enterprises to identify, shape and document their business processes. In this context enterprises require mechanisms to ensure that their business processes implement and fulfill compliance requirements independently from business level requirements. In this paper we present a novel approach for the modeling and implementation of Internal Controls in business processes. The approach is based on the formal modeling of Internal Controls, thus it can serve as the basis for usage of logic mechanisms in the compliance verification process. The main idea is the introduction of a semantic layer in which the process instances are interpreted according to given control statements, without changing the original (business-goal driven) business processes.

Keywords: BPM, Regulatory Compliance, Formal Verification, Semantic Technologies

1 Introduction

The advent of regulatory compliance requirements in the area of Internal Controls such as Sarbanes Oxley Act 2002 (SOX) [1] requires the implementation of an effective Internal Controls system in enterprises as a management responsibility. In this context COSO (Committee of Sponsoring Organizations of the Treadway Commission) has proposed an integrated framework [2], which is recognized by regulation bodies and auditors as a de facto standard for realizing the Internal Controls System. COSO defines the Internal Controls as a "process" designed to provide reasonable assurance regarding the achievement of objectives in effectiveness and efficiency of operations, reliability of financial reporting and compliance with applicable laws and regulations. Following is a summary of the Internal Controls process:

Identify all the **significant accounts** in the company. Identify for those accounts all **relevant business processes** affecting them. Define for each relevant business process a set of **control objectives** specific to the enterprise that must hold for that process. Continuously assess the **risks** for the enterprise by their identification for each control objective. Design and implement based on the risk assessment a set of effective **controls** in order to prevent or detect the occurrence of the identified risks. The controls must be tested and used in daily operations.

Since the realization and effectiveness of the above process involves different roles such as internal and external auditors together with consultants, the introduction and operations of Internal Controls compliance (i.e. SOX 404) is considered to be expensive and time consuming [3].

An approach is required to bring a higher level of adaptability, reusability and usability in Internal Controls compliance process. The adaptability is defined as an easy and fast way for introduction of new or changed controls on business processes. The reusability is related to the possibility to describe the controls on the conceptual level in order to abstract from the concrete implementation details of the controls. The usability addresses the need of bridging the gap between the non-technical auditing consultants and technical people realizing the controls implementation.

This paper introduces an abstraction layer above a business process, in which the controls are formally modeled and evaluated against existing process models and instances. It describes a novel, semantically-driven approach for the automation of Internal Controls in an enterprise, based on their conceptual separation from Business Process Management (BPM). In this semantic layer the controls are formally modeled and evaluated against existing process instances. We see several advantages of such an approach:

- It enables usage of formal methods, like inference, for the verification of a business process's compliance to Internal Controls and SOX compliance.
- Consequently, the compliance will be performed automatically, based on the current state of parameters (instances) of a business process
- Moreover, the conceptual description of control conditions ensures the flexibility of the approach, i.e. the changes of the controls will not affect the changes in the design and execution of the original business processes.
- Finally, through another abstraction layer introduced on the top of the compliances definition, we ensure that non-experts can built on top of the domain model provided.

We are mostly concerned with automation of the so called Application Controls (AC)[1], which control business processes to support financial control objectives and to prevent or detect unauthorized transactions. However, the approach provides a general framework that can be applied with respect to any other compliance domain using BPM technology.

The paper is organized as follows: We start with a motivating scenario for a new, flexible approach for compliance management. In the third section we introduce the domain model of Internal Controls/SOX compliance. In the fourth section we present our approach using the entities introduced in the domain model, whereas the fifth section explains its implementation architecture. Related literature is discussed in section six. Concluding remarks and some future research questions are given in the last section.

[1] Some literature also use the term „Process Control"

2. Motivating Scenario

We use the Purchase-To-Pay Process (P2P) delivered by an ERP product as an example. The process starts by creating the request for a purchase order (PO) and ends when the payment of that PO is recorded in Accounting. An excerpt of P2P is illustrated in Figure 1.

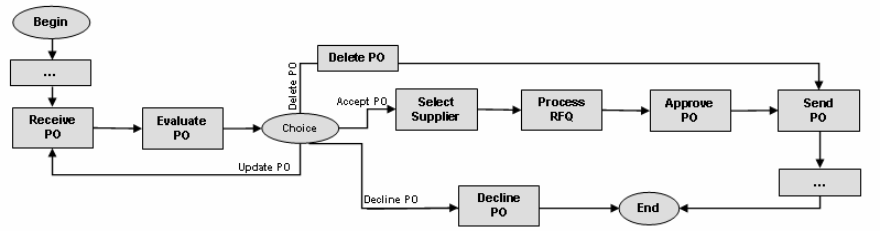

Figure 1 Purchase-To-Pay (P2P) Process: an excerpt

The Internal Controls compliance of P2P depends on enterprise specific risk assessment. Table 1 shows an excerpt of the risk assessment carried out by auditing consultants of two different enterprises. It shows their different control objectives, risks and controls on the same standard P2P-Process.

Table 1 Risk assessment on Purchase-To-Pay-Process (P2P) for 2 different enterprises

Control Objective	Risk	Application Control
Prevent unauthorized use	Unauthorized creation of POs and payments for not existing suppliers	POs higher than 5000 Euro must be double approved (*Double-Check-Control*).
Ensure adequate Supply of materials	Poor demand planning in the production	No POs higher than 5000 Euro will be approved at once.

Realizing the above introduced controls for each enterprise on the same standard P2P-Process provided by an ERP-provider means individual customization of the software implementing the P2P for each enterprise. This results in two completely different variant types, although from the business objective point of view these variants are equivalent: namely there business objective is to purchase goods.

3 Domain Model for Internal Controls Compliance

One of the main issues in the separation of the business and control objectives of a business process is that business objectives and control objectives for a business process have different life cycles and stakeholders. Figure 2 illustrates how we see the relationship between BPM and Internal Controls Management: The design of a control should control the way a business process is executed. A (re)design of a business process causes an update of risk assessment on a business process, which may lead to a new/updated set of controls incl. new tests. The business process

monitoring and verification techniques may be used to assess the effective design of controls and can serve as an input to Compliance certification.

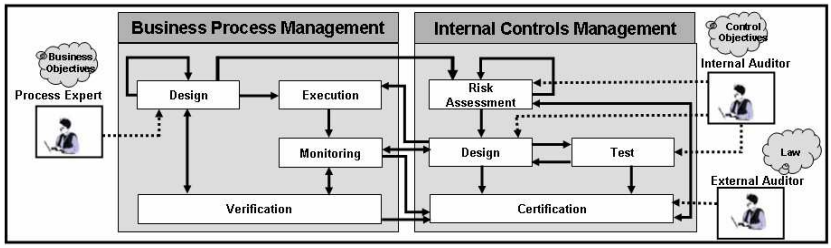

Figure 2 Relations between BPM and Internal Controls Management

Based on this view, we introduce a set of models for implementing the Internal Controls process. The entities and their relations to each other provide the terminology used to formulate logical statements representing the controls constraining the behavior of a business process. The approach itself is presented in the next section.

We enrich in following the entities resulted through our analysis of (mainly not IT-related) COSO by additional entities. These additional entities will enable the model to serve us as an operational basis for our approach later. Only those parts of COSO necessary for understanding our approach are presented Figure 3a. It shows the upper model of required entities for Internal Controls process introduced in chapter 1.

Application Control - Business Process Model

An *Application Control* (AC) controls different dimensions of the way a business process is enacted, namely the execution of its *activities*, the *Business Documents* involved and the *agents* performing an *activity* including their *authorities* (See Figure 3b).

For each *AC* at least one *Recovery Action* must have been designed, which reacts on the violation of a control. It does *not* change the designed business process logic; it rather blocks the transaction and may send a notification to an assigned responsible agent.

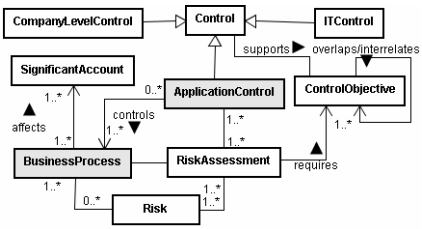

Figure 3a - The upper domain model of the Internal Controls Compliance

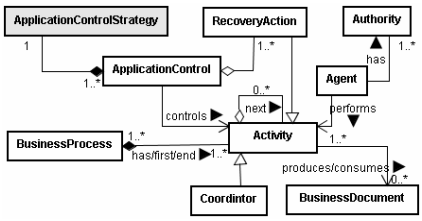

Figure 3b - Relationship between an Application Control and a Business Process

Application Control Strategy Model
An *Application Control Strategy* defines the way a control monitors the behavior of one or more activities inside a business process (Figure 4). In order to become active an *AC* requires to be triggered according to the *state* of the process parameters in a *scope*. We define further two elements of an AC strategy: *scope* and *pattern* based conceptually on the work done by Dwyer et al [5]. Although their patterns are mainly used for defining formal requirements on program specifications, they can be applied to internal controls compliance and the monitoring requirements there. For a detailed description of the scopes and patterns and their semantics please refer to [5].

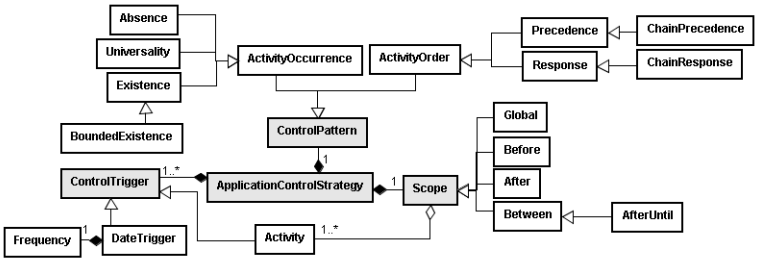

Figure 4 A Semi-formalization of the control implementation

Example: *Double-Check control* for the first enterprise (see Table 1) can be mapped to following strategy:

- *ControlTrigger* = Activity "Select Supplier"
- *Scope = Between* the activity "Select Supplier" and activity "Send PO"
- *Control Pattern = Bounded Existence* of *n=2* on activity "Approve PO"

4 The Approach

In order to realize the separation of the business and control objectives presented in Figure 2, our approach introduces another layer above business process model called "Semantic Process Mirror". According to assessed risks, a set of Application Controls is defined on that layer. Finally, by executing a business process, the semantic process layer will be continually updated with information needed for the evaluation of defined controls in order to ensure that compliance test will pass. The approach spans over there phases:

Phase 1: Semantic process mirror design phase
SemanticMirror represents a semantic layer placed on the top of the (usual) syntactical description of a business process (i.e. workflow). In this phase a model of the business process according to Figure 3b will be stored in the SemanticMirror. It will be used later during the phase 2 and 3 to infer whether the process is designed and executed according to a set of declaratively designed ACs in phase 2.

Phase 2: Application control design phase

In the following we present a set of formalizations needed for the automatic evaluation of ACs.
Control statement CS is a logical statement that describes how to carry out an AC *ac* in a business process *bp*:

$$CS(ct, bp, ac(x, cp), GS(bp, scope(M)), \text{action}_R) :=$$
$$O(ct) \wedge V(bp, ac(x, cp), GS(bp, scope)) \rightarrow Activity(bp, \text{action}_R),$$

where the formula for *CS* expresses that if a *violation V* for the given *ac* occurs (is true) after *occurrence O* of a *ControlTrigger ct* on a *Guarded Sequence GS*, then the corresponding *recovery action* action_R will be instantiated and executed on current instance of *bp* (the instance that generated the violation). We describe the parameters mentioned above: *Guarded Sequence* is a sequence of activities, which are along the *scope* of the *AC strategy* of an *ac* in a *bp*. The values for the violation of a control are calculated by evaluating the statement *ac* on the SemanticMirror, i.e. if the statement *ac* can be inferred from the set of facts contained in the SemanticMirror.

An *AC ac* expresses that a *control pattern cp* (See Figure 4) must hold if the logical condition on an entity *x* holds:

$$ac(x, cp) := condition(x) \rightarrow cp,\ x \in \{BusinessDocument, Agent\}$$

We show the formalization of the control pattern (cp) *BoundedExistence* of *n* (see Figure 4) for an *activity C* in the scope of activities defined by *GS*(*bp,scope*)*:*

$$\text{BoundedExistence}(n, C, GS(bp, scope)) :=$$

$$(\bigwedge_{i=0,..,n} \exists C_i \mid \text{InstanceOf}(C_i, C)) \wedge$$

$$(\bigwedge_{i,j=0,..,n} C_i, C_j \mid C_i != C_j) \wedge (\bigwedge_{i=0,..,n}^{n} C_i \mid C_i \in GS(bp, scope))$$

Example: Applied on the Double-check control in the P2P-Process (see scenario) the statement *ac* looks as follows:

$$\forall PO \mid \text{BusinssDocument}(PO) \wedge \text{Amount}(PO, amount) \wedge \text{greater}(amount, 5000) \rightarrow$$
$$\text{BoundedExistence}(2,$$
$$\text{ApprovePO}, GS_{\text{DoubleCheck}}(P2P, \text{Between}(\text{SelectSupplier}, \text{SendPO})))$$

Phase 3: Business process execution phase

This phase enables the bidirectional interaction between BPM and internal controls management (see Figure 1): The SemanticMirror will be updated by information about the current instance of the business process enacted and if an AC is violated, the recovery action defined in the control statement will be executed.

This approach enables dynamical application of the controls during execution phase of a business process. There is a minimum overlap between business process design and compliance design. Thus new application controls can be designed for

business processes by adding new control statements into SemanticMirror, while the original design of the business process remains unchanged, what is one of the main advantages of our approach.

5 Implementation

Beside the conceptual soundness, one of the challenges in such a kind of approaches is the possibility for their efficient and scalable implementation. There are two open issues that have to be discussed from the implementation point of view: 1) How to design and execute the business processes and 2) How to implement the SemanticMirror.

Regarding the first issue, we have selected to implement a prototype based on JBoss jBPM[2]. The basis for the implementation of SemanticMirror is the formal model of the Internal Controls (see section 3). We have decided to implement the control statements (CS) as Event-Condition-Action (ECA) rules. The Dwyer patterns and scopes [5] can be mapped to ECA rules, thus the control patterns and scopes can be mapped to them. We use JBoss Rule Engine (aka as Drools) implementing the RETE-Algorithm. Further, we are currently in the process of designing a Domain specific language (DSL) [7] based on the proposed model for Internal Controls for the auditors. The DSL expressions entered in the Internal Controls Design Tool (see Figure 5) will be mapped to the control patterns and consequently in ECA-rules before they are added into SemanticMirror.

Figure 5 illustrates the architecture of the prototype.

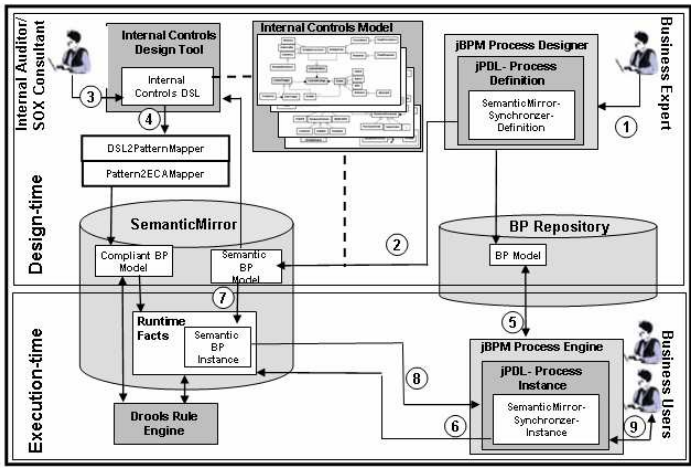

Figure 5 Architecture of the prototype

For the task of updating the SemanticMirror during execution time of business processes (Phase 3 of the approach), we use facilities provided by jBPM Engine

[2] JBoss, http://www.jboss.com

implementing the command software design pattern [6]: jBPM provides the possibility to register (during design-time) a so called *ActionHandler* to each node-class (activity) of a Process definition (called jPDL in jBPM) with additional custom functionality. Our implementation of the *ActionHandler*-Interface (*SemanticMirrorSynchronizer)* obtains a reference to the SemanticMirror and the current instance of the *execution context* provided automatically by the jBPM Process Engine to *SemanticMirrorSynchronizer* is added to the SemanticMirror.

6 Related Work

On a conceptual level our work is related to [4], where a taxonomy of risks in business processes is provided. It does not explicitly state how a risk is positioned inside the Internal Controls compliance domain and leaves the semantic link between risks, business process design and execution open.

In [8] and [9] the logic behind the obligations and permissions on a business process is made explicit in the form of temporal deontic assignments that can be used in business process design respectively their contracts. In these approaches, the constraints on business process would be designed into the business process, while we show how a designed constraint can be applied during execution time on business processes. The work done in [10] using Aspect Oriented Programming (AOP) techniques to extend the functionality of BPEL is closed to the separation of Internal Controls compliance concerns from BPM.

Software providers also offer related solutions for compliance management. [11] gives an overview and discusses the current software products in this area and their limitations.

However to our best knowledge, there is no approach which shows how Internal Controls could be declaratively formulated in terms of introducing a specific domain model for Internal Controls and showing an approach to formally declare and apply the controls separately from processes.

7 Future Research and Conclusion

In this paper we introduced a semantic based approach for conceptual modeling of Internal Controls required by regulation such as SOX. They are captured as declarative rules and deployed during execution-time on business processes. We built the model based on the de facto Internal Controls standard called COSO. Using this approach, new application controls can be defined on business processes without changing the original business logic of processes. The approach will enable definition of the controls outside of the workflow.

One concern in this context is the fact that although in our approach the recovery actions do not change the original business logic of the process, we have to verify the approach with results in the area of adaptive workflows [12]. Further we plan to detail the formalization and apply it to BPMN as target process modeling environment. Regarding the proposed architecture and the SemanticMirror synchronization

component we have to analyze and validate the performance affecting its real feasibility.

Another issue that must be addressed is the inter-control dependency: in order to become effective, a "well-designed" control may depend on existence, effective design and operation of other controls. This issue is actually also mentioned directly by law [13].

Further COSO (and also law) calls in this context to "manage the change" in the enterprise, which means among others that a new or redesigned business process should always be followed by a new risk assessment (and possibly new or updated set of controls). Today this is carried out mostly manually. We consider bringing a higher level of automation in this approach as an open research question.

References

1. Pub. L. 107-204. 116 Stat. 754, Sarbanes Oxley Act (2002)
2. Committee of Sponsoring Organizations of the Treadway Commission (COSO), Internal Control – Integrated Framework, 1992
3. T. Hartman, Foley & Lardner LLP, "The Cost of Being Public in the Era of Sarbanes-Oxley," June 2005
4. zur Muehlen, Michael; Rosemann, Michael. Integrating Risks in Business Process Models. In: Proceedings of the 2005 Australasian Conference on Information Systems (ACIS 2005), Manly, Sydney, Australia, November 30-December 2, 2005.
5. M. Dwyer, G. Avrunin, J. Corbett, Patterns in Property Specification for Finite-State Verification. In Proceedings of the 21st International Conference on Software Engineering, pages 411-420, May 1999
6. E. Gamma, R. Helm, R. Johnson and J. Vlissides, Design Patterns: Element of Reusable Object Oriented Software, Addison-Wesley, 1995
7. Marjan Mernik, Jan Heering, and Anthony M. Sloane. When and how to develop domain-specific languages. ACM Computing Surveys, 37(4):316–344, 2005
8. S. Goedertier and J. Vanthienen, Designing Compliant Business Processes from Obligations and Permissions, 2nd Workshop on Business Processes Design (BPD'06), Proceedings, 2006
9. Guido Governatori, Zoran Milosevic, and Sahzia Sadiq. Compliance checking between business processes and business contracts 10th International Enterprise Distributed Object Computing Conference (EDOC 2006). IEEE Press, 2006, pp. 221-232
10. Charfi, A. and Mezini, M. Hybrid Web Service Composition: Business Processes Meet Business Rules. In Proceedings of the 2nd International Conference on Service Oriented Computing (2004)
11. R. Agrawal, Ch. Johnson, J. Kiernan, F. Leymann: Taming Compliance with Sarbanes-Oxley Internal Controls Using Database Technology. Proc. 22nd Int'l. Conf. on Data Engineering ICDE'2006 (Altanta, GA, USA, April 3 – 7, 2006)
12. M. Reichert and P. Dadam, "ADEPTflex – Supporting Dynamic Changes of Workflows Without Losing Control," Journal of Intelligent Information Systems, 10(2) (1998)
13. Public Company Accounting Oversight Board (PCAOB), PCAOB Accounting Standard No. 2, paragraph 12