On the Risk Management and Auditing of SOA Based Business Processes

Bart Orriens, Willem-Jan v/d Heuvel, and Mike Papazoglou

Dept. of Information Management, Tilburg University
PO Box 90153, 5000 LE Tilburg, The Netherlands
{b.orriens,wjheuvel,mikep}@uvt.nl

Abstract. SOA-enabled business processes stretch across many cooperating and coordinated systems, possibly crossing organizational boundaries, and technologies like XML and Web services are used for making system-to-system interactions commonplace. Business processes form the foundation for all organizations, and as such, are impacted by industry regulations. This requires organizations to review their business processes and ensure that they meet the compliance standards set forth in legislation. In this paper we sketch a SOA-based service risk management and auditing methodology including a compliance enforcement and verification system that assures verifiable business process compliance. This is done on the basis of a knowledge-based system that allows integration of internal control systems into business processes conform pre-defined compliance rules, monitor both the normal process behavior and those of the control systems during process execution, and log these behaviors to facilitate retrospective auditing.

1 Introduction

SOA is an integration framework for connecting loosely coupled software modules into on-demand business processes. Business processes form the foundation for all organizations, and as such, are impacted by industry regulations. Without explicit business process definitions, flexible rule frameworks, and audit trails that provide for non-repudiation, organizations face litigation risks and even criminal penalties. Compliance regulations, such as Basel II [3], HIPAA [7], Sarbanes-Oxley (SOX) [27] and others require all organizations to review their business processes and ensure that they meet the compliance standards set forth in the legislation. This can include, but is not limited to, data acquisition and archival, document management, data security, financial accounting practices, shareholder reporting functions and to know when unusual activities occur. In all cases, these control and disclosure requirements create auditing demands for SOAs.

Internal control constitutes a fundamental cornerstone in auditing, which is used to assure business process compliance, delivering objective and independent guarantees regarding virtually all accounting aspects of service-enabled business processes, including risk management, financial checks and governance processes (Rezaee, 2007). A typical financial reporting control might mitigate the risk of misstating revenue due to inadequate physical or electronic security over sales documents and electronic files.

This helps implement a compliance regulation act, such SOX section 404, which mandates that well-defined and documented processes and controls be in place for all aspects of company operations that affect financial information and reports. To achieve this functionality requires: (i) controlling and auditing who accesses financial information, (ii) controlling and auditing what financial information is accessed, and (iii) ensuring financial information is not compromised during transmission. Due to the inherent complexity present in compliance regulations, such as SOX, most companies cannot address these requirements without a strategy for automating the integration of the diverse business processes and their accompanying internal control systems throughout the enterprise.

Existing auditing solutions and tools are hopelessly outdated and not applicable in SOA environments [20]. These are tightly coupled to the controlled application, and assume that applications are homogenous and monolithic in nature. Moreover, solutions are typically reactive in nature (noted also in e.g. [25]). That is, their focus is to detect violations after they have occurred. However, in today's business environment a pro-active approach is required in which organizations are able to control their business processes such that violations are avoided; and in the event that violations do occur, an immediate response can be carried out; for example by triggering an automated procedure to resolve a compliance problem or by notifying the business process manager. Some initial work in the area of risk management has been done (e.g. [17] and [25]), however, current proposals are still preliminary in nature. This paper introduces a service compliance methodology (SCM) that is intended to be a first step towards filling this SOA risk management and auditing void for service-enabled business processes (henceforth referred to simply as business processes). Concretely, SCM is intended to enable external control systems to be integrated into business processes during execution conform pre-defined compliance rules, monitor the behavior of these control systems in order to react to compliance violations at the moment that they occur, and log the application and outcome of the control systems for auditing purposes.

2 Risk Management and Auditing for SOAs

To provide the ability to establish control and documentation, reduce risk and error potential, in cases where service-enabled processes impact financial reporting (e.g. in end-to-end sales cycles, payment cycles or production cycles), we propose the use of a methodology based on the concept of a risk management and auditing SOA. Such SOA combines SOA with risk management and auditing principles for business processes, and relies on: 1) a risk management strategy to integrate control systems into business processes and monitor their behavior; and 2) an auditing strategy to evaluate the effectiveness of these control systems. The first ensures that business processes are executed according to predefined regulatory policies and that violations can be promptly dealt with; while the second allows for providing explicit proof of compliance enforcement and violation mitigation ro facilitate auditing.

By checking the control systems, risks can be mitigated while safeguarding service-driven processes and increasing their reliability. Auditors rely on internal control systems as they provide audit evidence that helps reduce substantive testing. Assuring the quality of internal control systems to reduce the number of auditing activities has in fact been a proven strategy since the 1970s [29]. In addition, and perhaps more importantly, auditing the internal control systems of processes within or between organizations is a required practice.

Given this rationale, SCM adopts a risk management strategy that addresses those fragments of a business process that are exposed to the risk of control weaknesses, while fewer efforts need to be spent on those process fragments (and services on which they rely) with strong controls. These items become candidates for immediate evaluation and, where necessary, remediation. For example, handling salaries might be deemed a low-risk item since they are tightly controlled by a small group of people. Revenue recognition, on the other hand, might be deemed high risk because of loosely defined recognition procedures. This strategy becomes particularly significant in large, business-critical SOA-applications.

According to the standard control definition given by ISA 315 [14], control activities performed on business processes (and therefore part of any SOA-based solution) may fall into several classes forming SOA risk management tenets:

1. Performance reviews: include reviews and analyses of actual performance versus budgets, forecasts, and prior period performance; relating different sets of data (operating or financial) to one another, together with analyses of the relationships and investigative and corrective actions; comparing internal data with external sources of information; and review of functional or activity performance.
2. Information processing control procedures: encompass application controls, which apply to the processing of individual business processes. These controls help ensure that all transactions occurred are authorized, and are completely and accurately logged and processed.
3. Physical controls: encompass the network-level security of service end-points, including adequate safeguards such as secured access/control to services; measures against data availability threats (e.g., XML attacks), and data integrity.
4. Segregation of duties: intended to reduce the opportunities to allow any person to be in a position to both perpetrate and conceal errors or fraud in the normal course of the persons duties. This is achieved by assigning access roles along a business process, logging service execution trails, and maintaining custody of services. For example, if an employee has custody of services and also accounts for them, there is a high risk of that person using the services for personal gain and adjusting their logs to cover theft (Hayes, 2005).
5. Authorization: accounting controls need to check procedures of reviewing and approving specific operations or transactions, e.g., approving the invocation of purchase orders, or change orders.

In addition, from a legislative perspective, an analysis of current compliance regulations (like Basel II [3] and Sarbanes-Oxley [27]) reveals that compliance requirements affect not only the basic structure of business processes, but also more advanced concerns such as monitoring, privacy, quality, retention, security and transactionality. Appropriate business process control activities should be integrated into process models to address these risk management issues.

To monitor business process control activities, the service compliance methodology should accommodate the following SOA auditing tenets (derived from intersecting core SOA with basic auditing principles conform [13] and [14]):

1. Independent auditing: The auditor (which can be a human or automated agent) is to be independent, but may be either internal or external to the organization(s) where the service-enabled processes execute.
2. Policing the SOA behavior: requires the ability to monitor events or information produced by the services/processes, monitoring instances of business processes, viewing process instance statistics, including the number of instances in each state (running, suspended, aborted or completed), viewing the status, or summary for selected process instances, suspend, and resume or terminate selected process instances. Of particular significance is the ability to be able to spot problems and exceptions in the business processes and move toward resolving them as soon as they occur.
3. Real-time reporting: requires the ability to disclose in real-time material events such as significant write-downs or bad debt recognition. Alerts can be represented as alarms directed to a human administrator. Alternatively, they can be real-time electronic events that in turn are used to trigger an automated remediation event like service shut-down or policy change.
4. Logging execution trails: requires the ability to log business processes and transaction execution trails to provide auditing capability and non-repudiation. Audit in an SOA transaction could involve tracking any number of activities and incidents. It must provide evidence that a particular identity accessed a specific service resource; the service consumers request satisfied the service providers security policies (communication integrity, privacy, data cleanliness, etc.); and that the service providers response satisfied security and performance contracts established with the service consumer (particularly if an SLA is specified in the policy). Secure logging of what happened, when, by whom and under what terms in an SOA communication underpins any forensic audit of a transaction.
5. Continuous auditing: There is a critical need for continuous auditing replacing the semiannual audits, which has become even more evident through new governmental regulations regarding real-time reporting requirements (SOX), that allows independent auditors to provide written assurance on a subject matter using a series of auditors reports issued simultaneously with, or a short period after, the occurrence of events underlying the subject matter [5]. The fundamental philosophy is that business processes must perform in a predictable manner accurately and precisely around target performance limits.

3 Service-Enabled Process Compliance Methodology (SCM)

The SCM methodology supports integration of internal control systems into business processes to facilitate risk management and auditing, and that meets the desiderata and constraints that were defined in section 2. The methodology adopts a formal deductive inference approach to apply compliance policies and rules to business processes by integrating appropriate control systems in a monitorable manner. The methodology is grounded on an abstract compliance enriched process model and a corresponding compliance rule specification language. The abstract, compliance enriched process model defines the basic constructs for specifying the elementary interaction arrangements, relationships and behavior of individual services that are to be assembled in a business process. Compliance requirements are annotated to these constructs using control primitives, which describe a specific usage of particular control system functionalities (e.g. in relation to monitoring or security). This allows organizations to address the identified risk management SOA tenets by incorporating appropriate control systems into their business processes.

Since the decision of whether or not to apply certain control mechanisms is often context dependent, compliance rules can be selected to express under what circumstances particular control primitives must be enforced. For example, a sales manager may only be interested in being notified about high risk sales orders made by customers with bad credit history and above $500. These kinds of compliance rule are specified using a sophisticated compliance rule specification language and packaged into compliance policies that address particular compliance legislations (or parts thereof). Business managers can select one or more compliance policy from a control directory and associate it with a business process. As this process is being carried out conform its abstract process model, the selected compliance rules will be applied to enrich the model with appropriate control primitives. This is facilitated by the knowledge base, which allows formal derivation of which control primitives are to be annotated to the model; where decisions made based on the rules are logged so they are available for future inspection.

The abstract, compliance enriched process model is declarative in nature and as such can not be directly executed. Therefore, to cater for the actual execution of business processes the abstract process model and its annotated control primitives are translated into an executable model. Here we adopt the defacto BPEL for that purpose. The translation of the elementary service interactions in the abstract process model is done by mapping its elements to BPEL equivalent constructs. Control primitives are integrated into the BPEL model by placing WS-Policy assertions on the services that are participating in the process. This ensures that only services capable of meeting the compliance requirements are engaged. Also, appropriate control activities are inserted to ensure that the BPEL process and its services actually behave in accordance with the specified control primitives. These can for example pertain to authorization aspects, segregation of duties, information processing, enforcement of system security measures, and verification of financial information in order to support the SOA risk management tenets. The resulting BPEL process is subsequently executed as normal. Because the application of compliance rules is typically runtime dependent (e.g. depending on the exact order message that was received), this procedure is followed every time that the BPEL process enters an execution scope (that is, there are activities to be executed). That is, each time an execution scope is encountered, execution is paused and the current BPEL execution model is translated into its abstract counterpart. Compliance rules are then applied to this abstract process model as described before and execution is resumed in a normal manner.

To support SOA auditing tenets 2,3 and 4 both the normal process activities in a BPEL process as well as any inserted control activities can be monitored and logged. For normal activities this is done by abstractly annotating them with appropriate monitoring control primitives. During transformation into BPEL corresponding monitoring activities are then included to enable monitoring. To accommodate the same for control activities, the abstract process model allows control primitives to be applied to other control primitives. This allows for example to define that in case a security authentication control is not passed, the response must be that this failure is logged and a notification is send to a human manager. These activities can themselves be logged as well, as such establishing a clear audit trail of the behavior of both the business process and its integrated control systems. Moreover, automated handling of control related events can be facilitated by defining appropriate responses in compliance rules (just like normal compliance rules). Furthermore, the behavior of integrated control systems can be audited independent from particular business processes. This allows both external and internal auditors (SOA auditing tenet 1) to verify the functioning of these control systems, after which they can be assured of their proper behavior when integrated into a process; which in turn allows processes to be performed in a predictable manner in accordance with compliance rules. Also, given the runtime nature of the SCM continuous auditing becomes more feasible. Finally, audit trails as well as reasoning logs can be examined to distil trends such as controls that are repeatedly not applied successfully.

Fig. 1 illustrates the SCM methodology and its individual steps. The SCM methodology works as follows: as a first step business managers selects pre-defined compliance policies and rules from the **control repository** by indicating what legislative

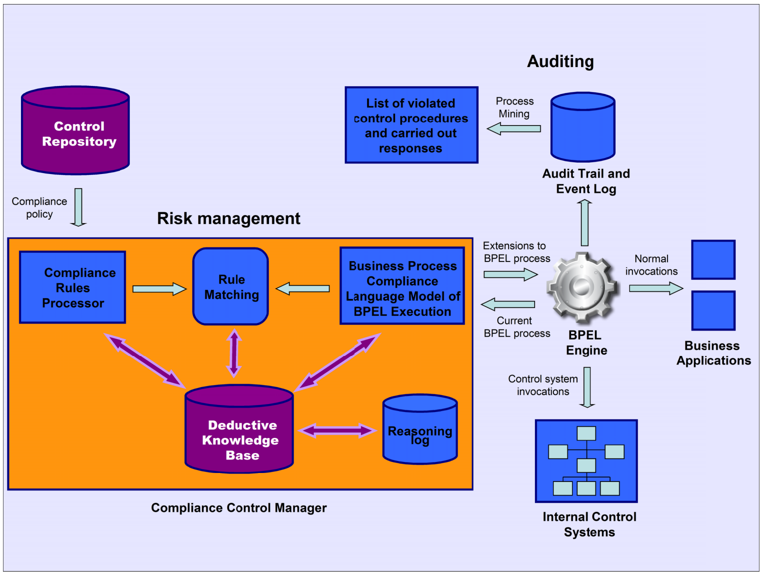
Auditing
Control Repository
List of violated control procedures and carried out responses
Process Mining
Audit Trail and Event Log
Compliance policy
Risk management
Compliance Rules Processor
Rule Matching
Business Process Compliance Language Model of BPEL Execution
Extensions to BPEL process
Current BPEL process
Normal invocations
BPEL Engine
Business Applications
Control system invocations
Deductive Knowledge Base
Reasoning log
Compliance Control Manager
Internal Control Systems

Fig. 1. SCM Methodology Overview

compliance requirements must be met by which particular business process. The **compliance control manager** receives the compliance rules collected by the **control directory** and assures that the targeted BPEL based business processes are executed in conformance to these rules. Concretely, what happens is that each time during execution that the **BPEL engine** enters an execution scope (i.e. one or more activities to be performed), the engine halts execution and sends its current execution model to the **compliance control manager**. The **compliance control manager** transforms the received execution model into a so-called Business Process Compliance Language (BPCL) based business process model. This BPCL model describes the prescriptive BPEL execution model in a declarative manner. The **compliance control manager** then carries out the rule matching activity to identify if and when certain compliance controls must be enforced based on the applicable compliance rules (also defined in terms of BPCL).

The rule matching activity is supported by the knowledge base which captures formalized models and rules and facilitates rule matching with logical inferences. Once reasoning has been completed, the **compliance control manager** contacts the **BPEL engine** in order to update its BPEL execution model by inserting control activities for enforcing any applicable control requirements, as well as adding constraints to the abstract services responsible for the process' normal activities. We adopt the WS-Policy standard for this purpose to express constraints as assertions. The **BPEL engine** then restarts execution, goes through the execution scope and carries out the activities it finds. Any added control activities are called in the same way as normal activities through service operation invocation; where these operations are implemented by special middleware services provided by **internal control systems** (e.g. offered via an enterprise service bus). Also, any constraints placed on the abstract services responsible for the execution of normal activities, are taken into account when the abstract services are bounded to actual services. This is done by comparing the stipulated WS-Policy assertions to those supported by available services (i.e. through standard service discovery and selection). As such, from the perspective of the **BPEL engine** the only thing that has changed is that more activities need to be performed.

In the remainder of this paper, we will discuss the workings of SCM in more detail. Because of space limitations we will only focus here on the specification of compliance enriched business process models, and compliance policies and rules; as such, our work on the actual application of compliance rules using the reasoning engine is omitted here. An overview of the BPCL in UML class diagram notation is shown in Fig. 2.

3.1 Modeling Compliance Enriched Business Processes

At the heart of the SCM methodology stands the Business Process Compliance Language (BPCL), which is currently under development. This language is intended to provide the constructs necessary to define compliance enriched business process models as well as compliance rules applicable to these models. The bottom part of Fig. 2 shows the portion of the BPCL that allows definition of the basic structure of business processes, and its annotation with control primitives. As can be seen a business process model is defined as a collection of process constructs. Such constructs represent building blocks from which a process model can be constructed. They are abstractly defined

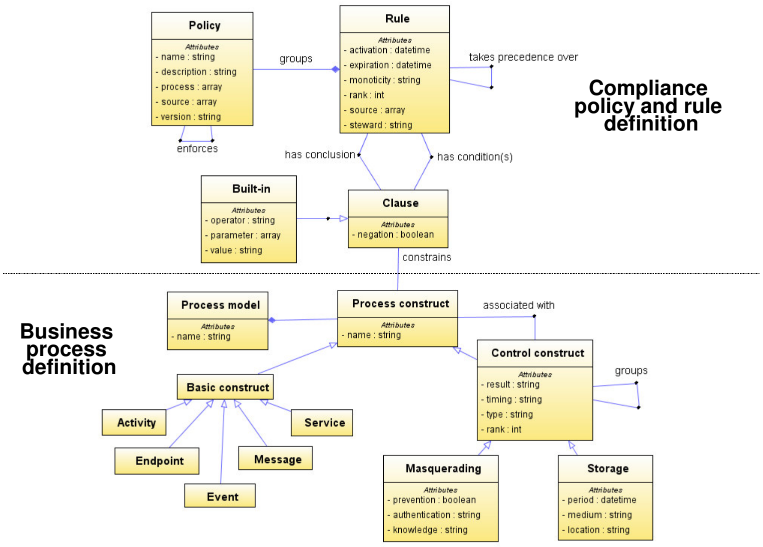

Fig. 2. Business Process Compliance Language (BPCL) Overview

in the **process construct** class. Each process construct has an unique name for identification and reference purposes. The **process construct** class has two subclasses, being **basic construct** and **control construct** class. Also abstract in nature, these classes demarcate the difference between the basic constructs making up business process models and the control primitives applicable to these constructs.

Basic process modeling constructs are themselves specialized into concrete classes representing 'standard' service enabled business process modeling constructs to capture the five viewpoints for business process modeling, being functional, locational, temporal, informational and participational view. Fig. 2 only shows the main classes in these viewpoints for reasons of clarity: **activity**, **endpoint**, **event**, **message** and **service** class. Messages represent containers of information consisting of meta-data and actual data. Meta-data comprises the information required to deliver the message and enable its processing, while payloads contain any content of the message not conveyed in its meta-data. Messages have a particular format (conform e.g. an XML schema), follow certain semantics (e.g. defined in an ontology), be in a certain language, and consist of message parts which represent snippets of information. Each message part has a name, type and value. Its type can be basic like double, integer, string, but also refer to a complex type (for example defined in the XML schema).

Messages function as the 'inputs' and 'outputs' of activities. Activities represent well-defined functions and can be dependent on one another. Activities can be complex in nature grouping other activities in parallel, loops or sequence (defined in appropriate activity class subclasses). Activities are 'carried out by' services. Services have properties like name and have associated classes capturing details concerning for example category and version. Services are 'found at' endpoints, which are related to classes capturing characteristics like network location, time zone, jurisdiction, and address information. Finally, events capture business process occurrences. Events are signaled by messages and are crucial for facilitating the monitoring of business processes. Events have an identifier, a time stamp, a severity indicating importance, and a causal vector identifying the events that caused it. The causal relations among events will follow the ordering of the activities generating the events. Also, events can be composite in nature aggregating other events; which will be conform how the activities generating events are structured into complex activities [18].

Control constructs annotate the basis process constructs with compliance related primitives. The basic characteristics of control constructs are comprised in the abstract **control construct** class. Control constructs are atomic or composite in nature. Atomic control constructs define exactly one compliance requirement, and share four attributes: 'timing', 'result', 'rank' and 'type'. The 'timing' attribute expresses when a compliance control must be enforced in relation to the normal process activities. It can be set to 'before', 'in place of' and 'after', and affects the manner in which control activities are integrated into the normal BPEL process. Timing is dependent on the type of control and the context in which it is used. For example, for an authorization control associated with an order approval activity the timing will be set to 'before'; as it does not make sense to perform authorization afterwards. The 'result' attribute of a control construct depicts the outcome of the control activity, and can be equal to 'success' or 'failure'. This attribute allows to consider the result of control activities and define appropriate responses in compliance rules.

The 'rank' attribute of the **control construct** class allows ordering in case multiple control constructs are associated with a process construct; e.g. to express that first notification of an event must be done and only then that the event is logged. To indicate that control constructs associated with the same process construct are to be carried out in parallel, their rank can be set to the same height. Lastly, the 'type' property contains the kind of control construct. We identify five main categories: *functional*, *informational*, *locational*, *participational* and *temporal* constructs, which are attached to activities, messages, endpoints, services, and events respectively. Individual control construct sub classes are then added to the BPCL to define specific control mechanisms. Fig. 2 shows two example control construct classes; e.g. the 'masquerading' control construct for a process activity, which can contain a property to depict that for authentication purposes an username/password combination must be provided as proof of knowledge. We are currently in the progress of developing a classification (and subsequent definition) of control construct for addressing a diverse range of compliance requirements. These concrete control constructs are all defined as sub-classes of the **control construct** class with appropriate attributes to define the exact requirements of the expressed control primitive (in addition to those defined in the **control construct** class). Due to space limitations we do not discuss this further here.

Different from atomic control constructs, the BPCL also facilitates definition of composite control constructs via 'groups' relations between control constructs. These constructs group other constructs (both atomic and/or composite) and apply their own control primitive to it. This allows for example to define that the notification and logging of an event must be done in an all-or-nothing manner (i.e. as an atomic transaction). Additionally, since control constructs are process constructs, they can themselves be annotated with control constructs. The interpretation of such annotation is that the control primitives expressed in the latter control constructs are applied to those with which they are associated. One important application of this is that it enables to attach monitoring constructs to other control constructs. This has the effect that the outcome of the application of these constructs is itself monitored. This empowers organizations to not only monitor the progress of their normal business process activities, but also those related to the effectuation of control mechanisms within these processes; e.g. allowing to define that if authentication for an activity fails, then a human manager must be notified. It also provides the means to express statement like that information must be stored in a secure manner; which can be captured by associating a storage control with an encryption control that itself is attached to a message. This has the effect that first the message is encrypted, after which the resulting (encrypted) message is stored.

3.2 Defining Compliance Rules and Policies

In the previous subsection we discussed the BPCL in relation to the definition of compliance enriched business process models; and explained how control constructs can be annotated to basic process modeling constructs. However, as observed in the introduction of section 3, the decision of whether or not to apply certain control mechanisms is often dependent on the specific business process conditions during execution. To accommodate for this the BPCL also provides the concepts for the definition of compliance rules and policies. Concretely, compliance rules can be specified in an expressive manner on top of the process modeling constructs. In Fig. 1, these rules are then matched against the BPCL representation of a running BPEL process to customize integration of control mechanisms into business processes. This is done with the help of the knowledge base inferencing capabilities. Both the BPCL model and the applicable compliance rules are translated into the knowledge base format, after which new facts constituting annotations of control constructs are deduced based on the current BPCL model. BPCL rules also have several attributes to facilitate their specification, categorization, application and management.

The central class in the BPCL for compliance rule definition is the **rule** class. Rules take the form `IF [conditions are satisfied] THEN [annotate control primitive]`. A rule has zero or more conditions and exactly one conclusion. Both are expressed as clauses based on the **clause** class. Clauses constrain process constructs and link compliance rules to business process models. A clause constrains a parameterized process construct. The **clause** class is specialized in the **built-in** class. Built-in clauses allow to express: 1) an evaluation of the value of a particular process construct attribute using a built-in operator and a set value; or 2) a derivation of a new value based on one or more process construct attributes (potentially from different parameterized process constructs) conform a built-in operator. Supported evaluation operators in BPCL include text and numerical comparison, membership evaluation, and, date and time operators. Derivation operators encompass addition, substraction, division and multiplication. An example is that `IF [customer order amount in an order is higher than $500], THEN [perform authentication using an username/password combination]`. To monitor the outcome of this control mechanism, another rule can then be that `IF [result of authentication equals failed], THEN [notify sales manager]`.

Rule conditions can be negated (via the clause 'negation' attribute). Negation empowers organizations to express both desired and undesired conditions. This allows for example to state that `IF [customer status is not equal to 'gold' and order amount is higher than $1000], THEN [do credit check]`. A relevant distinction in this regard is the intent of the negation. Strong, or classical, negation conveys the necessity to explicitly show that something is not true. In contrast, negation interpreted in a weak sense indicates that something is considered not true if it can not shown to be true. This is a subtle yet important difference for compliance, as in one case explicit proof is required of separation of duties whereas in the other case this is not necessary. At this point in time we only use negation in rule conditions; as we do not see direct application of compliance rules of the form "IF [conditions apply] THEN [do not annotate control primitive]". This also means that we restrict the usage of negation in rule conditions to weak negation (since no explicit negative conclusions can be drawn). This has the benefit that rules can be unambiguously be interpreted using perfect model semantics [28]; as such avoiding situations in which it is unclear which interpretation is correct.

Continuing, rules in BPCL can be monotonic or non-monotonic in nature as indicated in their 'monoticity' attribute. Non-monotonic rules (also known as defeasible rules) are common in business, for example to override standard rules with special-case exceptions, to incorporate more recent updates and etceteras [12]. In relation to compliance the matter of non-monoticity is of interest as it allows organizations to indicate to what extent it is important that a compliance rule is enforced; and consequently how grave the consequences are in case the rule is not satisfied. Monotonic compliance rules must always be met, but non-monotonic ones may be violated (albeit potentially at a cost). Monoticity also empowers organizations to prioritize their compliance rules in case they express conflicting requirements (e.g. the need for authentication contradicts with a demand for anonymity). Monotonic rules always take precedence over non-monotonic ones, while the latter can themselves be further ordered through the usage of the 'rank' property as well as 'relative prioritization' relations between rules. The effect of rule monoticity properties during inferencing is that it instructs the knowledge base to prioritize rules in case they care conflicting with one another.

Additionally, each rule has the attributes 'activation date', 'expiration date', 'source', and 'steward'. The first two express the period in which a rule is active. From a compliance point of view this is useful to ensure that rules are only enforced when appropriate. For example, the IFRS [24] requirements for financial reporting in 2007 likely differ from those of other years, and thus should not be applied for example in 2006 or 2008. Situations such as these prompt the need for some form of life cycle management to manage the status of compliance rules. The impact of activation and expiration dates is that rules will only be applied by the knowledge base if these rules are active given the internal clock of the knowledge base. In order to establish a link between a compliance rule and one or more compliance legislations, the origin(s) of a rule can be specified in the 'source' attribute. This will identify the name of a specific section/subsection of a legislation that the rule originates from (e.g. Sarbanes-Oxley section 402). This allows the categorization of compliance rules, enabling for example to select a subset of the Sarbanes-Oxley compliance rules and apply them to a business process. Finally, the delegation of responsibility for achieving particular compliance goals is another concern that is addressed in BPCL. Concretely, compliance rules have a ’steward’ attribute to identify organizational actors. This allows the responsibility for compliance enforcement to be tied to the organization’s management and operations structure.

Logically related compliance rules can be clustered into compliance policies using the **policy** class. Such policies are similar in nature to WS-Policy based policies [2], however they contain much more expressive rules than WS-Policy assertions. Currently policies can not contain multiple alternatives like in WS-Policy, though we will to include such support in the future if this is deemed useful. For identification purposes a policy has a name as well as a short textual description. Additionally, a policy has a ’source’ property, which identifies by name to what compliance legislation(s) the policy is related (typically in more general terms than the source specified for a compliance rule). Potentially a single policy can be related to multiple legislations, e.g. supporting compliance of both Basel II and Sarbanes-Oxley at the same time. Each policy also has a ’process’ property indicating to what type(s) of business process it is applicable (e.g. the purchase and payment process). Finally, there can be multiple versions of policies in existence. Differentiation between them is expressed using the ’version’ property. Based on the source, process and version properties customized packaging of compliance policies and rules can be done by organizations when accessing their control directory. This increases intuition, as it enables managers to refer to particular legislative compliance issues (rather than manually collecting compliance rules and policies).

For example, if a sales process manager wishes to apply all ’Sarbanes-Oxley section 402’ related compliance rules to the sales process, then he/she will define a request by stipulating the source (Sarbanes-Oxley section 402) and the type of process (the sales process). In response, the control directory will search its contents and retrieve the SoX compliance policy defined for the sales process by comparing against the ’source’ and ’process’ attributes of available policies. For each found policy, the control directory removes any rules not related to section 402. The control directory next groups the resulting rules into a compliance package. In case multiple versions of a Sox 402 policy were found, these are presented to the sales process manager for selection. Interestingly, this approach also allows specification of requests for compliance of a process to multiple regulations. To illustrate, the sales process manager can stipulate to apply all Sarbanes-Oxley 402 rules, as well as the Basel II related rules. The control directory will retrieve the appropriate rules for both types of legislation and merge these into a single compliance policy. In all cases, the resulting compliance policy is sent to the compliance control manager, who ensures that the contained rules will then be applied during business process execution conform the SCM methodology. Due to space limitations we do not discuss this further here.

4 Related Work

In the last years there has been an increase in attention paid to the role of compliance within business processes. A typical example is [21] which defines a formalization for internal controls and how they relate to operational processes. Similar works include [23] and [10]. Though these contribute to the insight in the relation between processes and compliance, they do not address how control mechanisms can be integrated in executable business processes. In the area of process control objectives works have chartered the implications of compliance for IT, most notably COSO [6] and COBIT [15]. COSO identifies several control activities (including authorizations, data verifications, reviews of operating performance, security of assets and segregation of duties), but does not define how to integrate them in business processes. Related, COBIT (short for Control Objectives for Information and related Technology) is useful as it identifies a large number of control objectives for business processes, which are subsequently refined into concrete application controls. However, like COSO, COBIT does not provide the means to integrate these objectives into business process models.

[25] presents a framework for the modeling of control objectives within business process structures based on a modal logic based approach using Formal Contract Language [11]. This is akin to the sketched BPCL, but we allow more expressive definition as control constructs can be associated with and/or grouped by other control constructs. [25] also advocates usage of a controls directory which holds the interpretation of compliance regulations in the specific context of an organization (given the ambiguity of such regulations). Our approach will be able to facilitate this following [8], which attaches meta-data to compliance rules to depict the relationship between specific rules and compliance legislations. Finally, [25] hints at how the approach allows assessing the degree of process compliance; and how the usage of logic allows for analyzing why a particular decision in a business process was made. The SCM approach will be able to provide the same kind of reasoning via its formal knowledge base. Moreover, given the expressive nature of the BPCL (e.g. in terms of monoticity and prioritization), the SCM is planned to facilitate more sophisticated assessment and reasoning.

[26] proposes an aspect-oriented based approach for linking compliance to business protocols (i.e. abstract business processes). This is very similar to the usage of compliance rules in this paper, where rule conditions express aspect pointcuts and the rule conclusion defines the advice. The difference is that in our approach these are externalized and administered by a separate rule engine; positioning them better for analysis and management purposes. [17] sketches a model checking method in which business process models are expressed in the Business Process Execution Language (BPEL) [1], and then transformed into pi-calculus and finite state machines. At the same time compliance rules are expressed in a graphical Business Property Specification Language, which are next translated into linear temporal logic. Then, process models are verified against the resulting statements by means of model-checking technology. [9] proposes a similar approach using semantically annotated process models based on Business Process Modelling Notation (BPMN) [22] and Computational Tree Logic. Although useful in nature, our work extends these approaches by allowing for more rich annotation of control primitives.

[8] suggests to capture business processes and Canadian privacy related legislative requirements separately using the User Requirements Notation (URN) [16]). The paper also proposes to add documentation notes to the goal model capturing the privacy requirements , similar like what is done in this paper. [4] observes that often compliance objectives are delegated within the organizational hierarchy, where they are refined in a top-down manner. The described management is similar to the one proposed in this

paper. In this regard [4] also notes that in delegation compliance requirements are often refined; e.g. via goal refinement similar to for example what is proposed for security in [19] in relation to the Tropos methodology. Our work does not address this issue yet, but we plan to include this in the future.

5 Conclusions

Business processes form the foundation for all organizations and are subject to industry regulations. Without explicit business process definitions, flexible rules frameworks, and audit trails that provide for non-repudiation, organizations face litigation risks and even criminal penalties. To address such problems we have proposed a SOA compliance methodology (SCM) based on the concept of a risk management and auditing SOA - to oversee the compliance of business processes with internal accounting control for the purpose of risk management and auditing. The results that we have presented provide an initial basic theoretical foundation for addressing the raised SOA based business process risk management and auditing tenets. Significant extensions are needed in several directions to guarantee a practical methodology. Work is needed to realize runtime support in the form of interaction with the BPEL engine to achieve integration and enforcement of control requirements during execution. Also, validation of the approach (particularly in terms of the adopted architecture and BPCL) in the context of real life case studies has to be carried out. Moreover, for demonstration purposes illustrative compliance rules and policies should be developed that address certain compliance regulations.

References

1. Alves, A., Arkin, A., Askary, A., Barreto, C., Bloch, B., Curbera, F., Ford, M., Goland, Y., Guízar, A., Kartha, N., Liu, C., Khalaf, R., König, D., Marin, M., Mehta, V., Thatte, S., van der Rijn, D., Yendluri, P., Yiu, A.: Web services business process execution language version 2.0 (April 2007)
2. Bajaj, S., Box, D., Chappell, D., Curbera, F., Daniels, G., Hallam-Baker, P., Hondo, M., Kaler, C., Langworthy, D., Nadalin, A., Nagaratnam, N., Prafullchandra, H., von Riegen, C., Roth, D., Schlimmer, J. (eds.) Sharp, C., Shewchuk, J., Vedamuthu, A., Yalçýnalp, Ü., Orchard, D.: Web services policy 1.2 framework (April 2006)
3. Basel Committee on Banking Supervision. International convergence of capital measurement and capital standards (June 2006)
4. Breaux, T., Antón, A., Spafford, E.: A distributed requirements management framework for legal compliance and accountability. Technical Report 14, North Carolina State University Computer Science (2006)
5. Canadian Institute of Chartered Accountants. Continuous auditing: research report. CICA/AICPA (1999)
6. COSO. Internal control for financial reporting - guidance for smaller public companies (2006)
7. Department of Health and Human Services. Hipaa privacy rule. US Federal Register (December 2000)
8. Ghanavati, S., Amyot, D., Peyton, L.: A requirements management framework for privacy compliance. In: Proceedings of the Workshop on Requirements Engineering (2007)

9. Ghose, A., Koliadis, G.: Auditing business process compliance. In: Proceedings of the International Conference on Service-Oriented Computing (2007)
10. Goedertier, S., Vanthienen, J.: Designing compliant business processes with obligations and permissions. In: Eder, J., Dustdar, S. (eds.) BPM Workshops 2006. LNCS, vol. 4103, pp. 5–14. Springer, Heidelberg (2006)
11. Governatori, G., Milosevic, Z.: A formal analysis of a business contract language. International Journal of Cooperative Information Systems 15(4) (2006)
12. Grosof, B., Gruninger, M., Kifer, M., Martin, D., McGuinness, D., Parsia, B., Payne, T., Tate, A.: Semantic web services language requirements (February 2008)
13. Hayes, R., Dassen, R., Schilder, A., Wallage, P.: Principles of Auditing: An introduction to international standards on Auditing. Prentice Hall/Financial Times (2005)
14. International Federation of Accountants. Handbook of International Auditing, Assurance and Ethics Pronouncements. John Wiley, Chichester (2006)
15. IT Governance Institute. Framework for control objectives: Management guidelines and maturity models (cobit 4.1) (2007)
16. ITU-T. User requirements notation (urn) – language requirements and framework. ITU-T Recommendation Z.150 (February 2003)
17. Liu, Y., Müller, S., Xu, K.: A static compliance-checking framework for business process models. IBM Systems Journal 46(2), 335–362 (2007)
18. Luckham, D.: The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems (Hardcover). Addison-Wesley Professional, Reading (2002)
19. Mouratidis, H., Giorgini, P., Manson, G.: An ontology for modelling security: The tropos approach. In: Proceedings of the 7th International Conference on Knowledge-Based Intelligent Information & Engineering Systems, Oxford, United Kingdom (September 2003)
20. Murthy, U., Groomer, S.: A continuous auditing web services model for xml-based accounting systems. Accounting Information Systems 5, 139–163 (2004)
21. Namiri, K., Stojanovic, N.: Towards a formal framework for business process compliance. In: Proceedings of the Multikonferenz Wirtschaftsinformatik (February 2008)
22. Object Management Group. Business process modeling notation (February 2006)
23. Padmanabhan, V., Governatori, G., Sadiq, S., Colomb, R., Rotolo, A.: Process modeling: The deontic way. In: Proceedings Of The Australia-Pacific Conference on Conceptual Modeling (2006)
24. PriceWaterhouseCoopers. Adopting ifrs first-time adoption of international financial reporting standards (June 2004)
25. Sadiq, S., Governatori, G., Naimiri, K.: Modeling control objectives for business process compliance. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 149–164. Springer, Heidelberg (2007)
26. Svirskas, A., Courbis, C., Molva, R., Bedzinskas, J.: Compliance proofs for collaborative interactions using aspect-oriented approach. In: Proceedings of the IEEE Congress on Services (2007)
27. US Congress. Sarbanes-oxley of 2002 (January 2002)
28. van Gelder, A., Ross, K., Schlipf, J.: The well-founded semantics for general logic programs. Journal of the ACM 38(3), 620–650 (1991)
29. Yu, S., Neter, J.: A stochastic model of the internal control system. Journal of Accounting Research 11, 273–295 (1973)