Authoring and deploying business policies dynamically for compliance monitoring

Yurdaer Doganata\textsuperscript{1}, Keith Grueneberg\textsuperscript{1}, John Karat\textsuperscript{1}, Nirmal Mukhi\textsuperscript{1}
\textsuperscript{1} IBM Research, 19 Skyline Dr., Hawthorne, New York 10532
\{yurdaer, kgruen, jkarat, nmukhi\}@us.ibm.com

\textbf{Abstract}— A policy authoring tool is integrated into a business provenance management system for dynamically authoring, deploying and monitoring compliance. The policy authoring tool enables creation of business rules in the language business people understand and deploys them into a rule engine. Once the policy is deployed, the compliance of process execution traces stored in business provenance management system can be checked against these rules and the compliance results are displayed on a dashboard. Salient features of the solution architecture, including semantic mapping of IT terminology into business vocabulary and transforming business rules into key control points in the compliance checking system, are explained. An auto-insurance claim process simulation is developed to evaluate the effectiveness of the approach and the results are presented.

\textbf{Keywords}— Policy authoring, business rules, compliance checking, business provenance

\section{I. INTRODUCTION}

In an environment where practices, polices and regulations that govern the businesses operations change fast, business people need tools to author and deploy business rules rapidly. They also need capabilities to monitor the compliance of business operations against these rules. The purpose of such tools and systems is to isolate the business people from the complexity of the IT system while bridging the gap between business and IT. In this paper, we introduce a system that enables creating business rules dynamically in a language that business people will understand and deploying these rules automatically to check the compliance of process execution traces against these rules.

New business strategies or mandates from external regulatory agencies require implementing internal control points within the IT infrastructure to monitor the compliance. Business people who are responsible for running the business operations according to rules and regulations usually are not technology experts. Implementing effective internal control points, however, depend on in depth knowledge of IT system and business application code. Traditionally requirements of internal control points are communicated to the IT organizations by business people and the implementation of internal controls is buried into the application code. The problem with the traditional approach is that implementation is time consuming and costly.

Our approach aims to eliminate the need to communicate implementation requirements of internal control points to the IT organization, hence avoiding cost and delay. We accomplish this goal by integrating a policy authoring tool that enables business people to express their rules in a language that they use in daily operations with a system that captures, correlates and stores process execution traces. These business rules consist of semantic constraints that impose conditions on how processes should be carried out. Semantic constraints are created by policy authoring tool. The required business behavior expressed by using semantic constraints are then converted into internal control points expressed at the system level in terms of IT data and deployed directly into the IT system.

Bridging the gap between the business people and the IT system requires a link between IT data and the vocabulary that business people use. We realized this by creating a semantic map between IT data and the business vocabulary when business event data is captured. Semantic mapping allows generating rules that are expressed in terms of business vocabulary which are directly linked to IT data. Note that IT data generated across multiple systems is loosely coupled within the organization and utilized by various business applications associated with different taxonomy. Hence, artifacts captured across the organizations need to be correlated under a common data model before the semantic mapping rules are applied. Another important aspect of our solution to bridge business and IT is a successful mapping of the rules created by the policy authoring tool onto a set of internal control points that can be computed against the business objectives.

The policy authoring tool that is used in this paper is the SPARCLE system which is designed to create structured rules and has been tested in customer trials\cite{1}\cite{2}. Originally SPARCLE system employed natural language parsing based on grammars to identify rule components in natural language text. In this paper, we modified SPARCLE to provide semantic constraint templates which are developed to capture the structure of rules allowable for the current application in place of grammars. Templates eliminate the efforts of building NLP structures by the business people at a price of reduced expressiveness. This approach, however, provided similar function with easier to develop and implement\cite{3}. The details of the policy authoring tool is given in Section IV. Policy authoring tools other than SPARCLE could also be used for the purpose of
this study, such as ILOG Business Rule Management System [4]. Regardless, SPARCLE does provide a usable mechanism which allows business experts to author policies without requiring extensive IT staff involvement.

Our focus in this paper is to enable the evaluation of semantic constraints through internal controls by analyzing the process execution traces collected at runtime. Hence, ensuring the traceability of the process is an important aspect of our solution. The reason is that the evolution of the process is not always predictable at the design level in particular when the activities rely on human interactions in the absence of predefined control structures. In such unmanaged or partially managed processes, the processes may evolve inconsistently with the original design goals and it may not be possible to enforce all semantic constraints at the process modeling level. In general, the context of the process is created during process execution. For this reason, it is vital to validate business rules against the process execution traces that is against the traces of what has actually happened.

The solution introduced here facilitates the integration of policy authoring tools with business provenance management systems. This way a bridge is established from policy authoring to creating, deploying and displaying the status of key control points dynamically and check the compliance against process execution traces.

The paper is structured as follows; the components and the architecture of the solution is presented in section III, the technical details of the policy authoring tool and its integration points are described in section IV, business provenance management system where the process execution trace is captured and managed is explained in section V. Mapping the semantics of the business operations to IT level view as well as mapping the semantic constraints to IT level internal controls to check compliance against execution traces are explained in section VI. Finally, in section VII, a dashboard to access compliance status information and in section VIII an auto-insurance claim simulator that is used to evaluate the bridge built between business and the IT people are presented.

II. RELATED WORK

Internal controls are necessary for continuous assurance of business goals. Recent legal requirements such as the Sarbanes-Oxley Act [5] and EuroSOX [6] increased the adoption of the practice of creating internal business controls to ensure business process compliance. Internal control processes within organizations help to ensure business processes behave according to the business objectives, external regulations or security rules [7][8]. Internal control points can be realized systematically by utilizing Enterprise Risk Management (ERM) frameworks such as COSO ERM [9][10]. The steps of the process can be summarized as identifying all business processes, assessing the risk of achieving business objectives, determining the control points to check to prevent business risk from occurring, and testing to check if control is working as intended. The implementation of these steps reveals that there is a big gap between the first and the last steps of creating internal control points. In the first step, business objectives and associated risks have to be defined. This is best done by people participating in the business activities of an organization and is done in the language of the business that they communicate among themselves. In the last step, the business objectives to be monitored have to be mapped on to IT data that is represented in schemas, class models and stored in a database as rows and columns of a table. According to the COSO framework, senior management is responsible for defining and establishing specific internal control policies and procedures. The implementation of these polices, however, are left to the IT organizations. An important aspect of the solution described here is to enable business people to generate operational rules that define business policies and practices and deploy them into the IT system automatically without involving IT people. Hence, this paper is an attempt to bridge this gap by using policy authoring tool and business provenance management technology.

The Business rules approach is extensively used to control and influence the behavior of organizations [11][10][12]. Despite the number of vendors offering business rules solutions, adoption is slowed by the overhead that is required to develop and maintain rules. Object Management Group (OMG) introduced the semantic of business vocabulary standard (SBVR) as a formal and detailed natural language declarative description of business entities [13][14]. Conceptually, the SBVR standard promises to capture business specifications in restricted natural language and represent them in formal logic so they can be machine processed. The tools that will make this standard widely accepted, however, yet to be developed. The constraint specification and formalization is out of the scope of this paper but interested reader is referred to [15].

The state of the art in business process compliance is reviewed in [16] where the approaches are classified as design time validation, runtime validation, enforcement by check points and a posteriori analysis. Our approach can be used in both runtime and a posteriori analysis, since the compliance rules are validated against process execution traces.

III. SOLUTION ARCHITECTURE AND COMPONENTS

This section presents the basic building blocks of our solution. The components of the solution architecture are depicted in Figure 1. “Business Process Modeling” component represents the model of the process for which the policies are authored. The business process model encapsulates a range of formal and informal representations of the core aspects of the business including the purpose and the activities around the purpose. It is an abstraction of the business processes without the involvement of IT, and it is a source for extracting the business semantics. Thus, the business model is both used to implement the business processes and to extract business semantics. The “Semantic mapping” component has interfaces both with the business model and it’s IT implementation, and maps the IT
terminology onto business vocabulary. An XML file and associated schema are generated as a result of this mapping. Examples of semantic mapping will be given for auto insurance claim scenario in Section VI. Semantic Mapping: Creating Business Vocabulary.

The Semantic mapping component is built at the time of IT system implementation. The data model and the associated schema developed for IT system events are used to map the IT data onto the business vocabulary. Consistent with the IT data model, the business vocabulary is grouped into the following dimensions:

Data: The business artifacts that were produced or exchanged during the business process such as documents, e-mails, database records, etc.

Tasks: The terms that represent the tasks executed during the business process

Resources: The vocabulary items that represent the actors of the business activities. A runtime component may also be considered a resource, such as a content management system.

The Policy Authoring Tool imports the business vocabulary and provides an interface to create business rules based on existing or newly created rule templates. Once the vocabulary is imported, policy templates are populated with the vocabulary relevant to the business and specific rules can be created. The details of policy authoring tool will be explained in Section IV.

A business rule is expressed as data, task, resource items and their relations consistent with the business vocabulary items. It describes which tasks should be performed, when, how and who should be involved. The deployment of a business rule into the provenance store is realized by creating multiple internal control points within the database. Provenance store contains the process execution traces captured by event data capture component. The captured execution traces are then correlated and stored as graphs. This way each compliance control point can be represented as a sub-graph connecting vertices (data, task and resource items) with edges (relations). The components interconnected to the provenance store constitute a business provenance management system. In Section V, an overview of the business provenance management system is presented.

IV. POLICY AUTHORING TOOL

Business rules are units of an overall business policy. The Policy Authoring Tool allows users to author business rules using natural language templates. Once the rules are authored, they can be deployed to an environment where the rules are executed, in our case, the business provenance management system. The rules generated from the templates can be expressed in plain English sentences which are easily understandable by business people and in formal notations understandable by IT system components. Since the structured English is provide by the templates, the rule generation process is reduced to selecting the right vocabulary.

A. Policy Templates

In order for business people to author the rules, templates need to be created by an administrator or someone who is familiar with the business process. The template consists of a policy phrase in natural language which includes one or more variables or “attributes”. The policy phrase is essentially a subject, condition, and action, with some “glue text” (text between attributes) to make the rule readable. The templates can also contain methods to map the policy phrase to the system where the policies will eventually be deployed. Once the template is authored, any business person can create rules by simply selecting the attributes from a series of drop-down menu choices as depicted in Figure 2. Many business rules can be instantiated from a single template, and policy authors might have multiple templates to work from. The example template in Figure 2 shows the structure of a policy phrase where obligations of a particular resource (or actor of the process) and the associated constraints for that obligation are expressed. In this example, the ClaimResource is a resource type and ClaimTasks is a task type. The conditions for the obligation are expressed in terms of logical and comparative relations among three different data types, namely ClaimDocument, InitialDamageEstimation and UpdatedDamageEstimation data types. For every data, task or obligation expression used in a template, a drop down list is provided for all possible choices in its category. For example, possible values for {Resource.ClaimResource} resource type are listed in the drop down menu as “CSR”, “Claim adjustor” and “Claim handler”. Similarly, two choices are given for the obligation type expressed by {operator.obligation}, namely “must” and “should”. The menu choices for the tasks to be performed by the resources are listed as “create claim document”, “determine if car should be totaled”, “request damage estimate”, and “request damage estimate update”. A policy phrase such as “Claim adjustor must request damage estimate and must decide to total if the car value is less than initial repair cost or updated repair cost estimate” is created from the template by simply selecting the values of task and data types from drop-down menu list.
The values in the drop-down menus are called the attributes of the policy and they depend on the type of business. Although the structures of policy phrases are similar for all business and they are the statements about resources, tasks and data items of a business process, the vocabulary changes from business to business. These attributes are part of the business vocabulary created during semantic mapping and imported into the policy authoring tool. Next section describes how attributes relevant for a particular business are generated and imported into the policy authoring tool.

![Policy Authoring Tool: Structure of a policy phrase in a template](image)

**1) Attributes: Business Vocabulary Items**

Before business rules are defined, a business vocabulary needs to be established. In the Policy Authoring Tool, policy attributes are populated by the business vocabulary. Attributes are used in templates to create the drop-downs in the natural language policy phrase, and to map the conditions and actions to the properties in the execution environment. The XML schema of the data model with semantic information, created by the semantic mapping component as will be explained in Section VI, is imported into the authoring tool to produce the data, resource, and task attributes as shown in Figure 3. Data, resource and task are the types of process event data captured by the business provenance system. Figure 3 shows some attributes of type Data, Resource and Task. As an example, some attributes of type Data are “Claim document”, “Police Report”, and “Invoice”. Furthermore, “Car” is an attribute of “Claim document” and “Make” is an attribute of “Car”.

Business vocabulary is naturally created when the business process is modeled based on the specification of business operations expressed in natural language. The activities, tasks, resources used to execute these tasks and data artifacts utilized in a business process constitute the sources of the vocabulary associated with the business. When the process is implemented, however, the business vocabulary is lost significantly. This is mainly because the IT level implementation details do not depend on the business vocabulary. Software developers or system integrators do not feel obliged to use the same vocabulary. Reusing the existing code for different business processes and integrating with legacy systems are some of the reasons why using the same vocabulary is not practical both at the IT and business levels.

**V. BUSINESS PROVENANCE MANAGEMENT SYSTEM**

The main function of business provenance management system is to capture and manage execution process execution traces. The IT implementation of a business process generally spans many systems and applications. The business provenance subsystem [17] monitors the various underlying systems across which the process executes, and is responsible for gathering relevant events (such as tasks being performed, data being accessed or modified and so on), and correlating these into coherent business process traces. Organizational, functional and data aspects of the business are captured by resource, task and data records, respectively. Each relevant event produced by the IT systems is stored in a provenance graph as a particular type of node or edge.

Central to this process is the development of the provenance data model, based on the IT implementation of the process. During the development of the provenance data model, the kinds of nodes and edges that are expected to be produced at runtime are modeled, based on the known types of events that the IT systems will produce. A relation between a resource record and a task record shows who was involved in executing that particular task. A relation between a data record and a task record reflects the effects of the task on the business artifact or the task dependency on its availability. A business goal often combines aspects by
description which tasks should be performed, when, how and who should be involved. Business control points associated with the business rules can be created using this description in terms of provenance graph entries. A business control point is satisfied if certain vertices and edges exist in the provenance graph. Hence, it is possible to claim that a business control point is a sub graph of the provenance graph.

Once the business rules are created by using the policy authoring tool, they are then mapped onto business control points that are represented as the sub-graphs of the provenance graph. Business control points associated with the business rules are stored in the provenance store linking associated data, task and resource items. The compliance results of process execution traces against the deployed business control points are queried from the provenance store and results are displayed in a dashboard.

Figure 1 shows the connection between “data generation/recording” client and the provenance store. The recording client processes application events, transforms them into provenance events and records them in the provenance store. A provenance event contains a subset of application data that needs to be stored as business provenance. The captured data must be relevant and specific to business control points. To avoid redundancy and possible exposure of sensitive data, recording clients do not copy all application data.

The data correlation and enrichment component in Figure 1 links and enriches the collected data to produce the provenance graph. To do so, the analytics components have access to the content of the provenance store. The enriched business data is accessed through a query interface and analyzed to verify business control points in two different styles. Firstly, a query can be deployed into the provenance store to emit results in real-time, feeding existing dashboard systems to display key performance indicators for example. Secondly, a query front-end enables visualization and navigation through the provenance graph from the outside.

VI. SEMANTIC MAPPING: CREATING BUSINESS VOCABULARY

As discussed above, the attributes of business rules that are used in the templates are created by semantic mapping. The purpose of the semantic mapping is to provide a method for the user to reconcile the business level view of the process with the IT level view. The module takes as input the Business Ontology for this process and the IT Terminology. The Business Ontology of the process specifies the artifacts associated with the process, and semantic definitions for each of them. These artifacts include the tasks that need to occur within the business process, the various pieces of data created, accessed or modified by the tasks and the actors involved.

The semantic mapping translates concepts between these two worlds: a piece of data at the Business Ontology level might be represented as a fragment of an XML message in the IT implementation (and therefore in the provenance store). The semantic mapping component provides a schema to represent these relationships, and a set of APIs to help users create, examine and manipulate the mapping. In an environment where IT event data is coming from disparate system, including legacy systems, automation of semantic map may not be possible if the underlying IT system does not have associated business taxonomy. Automation of semantic mapping requires that each IT system provides for a semantic mapping for all the business application that the system is used for.

As an example, consider the business artifact Car Value. At the business level this concept is directly used in the business vocabulary to describe policies such as 'the car value must be less than the damage estimate'. However at the IT level, the car value is a fragment of the XML element car which itself is contained within the XML message of type ClaimDocProvenanceType. We leverage XPath to reference the fragment that maps to the business artifact Car Value as follows:

```
<html><head></head><body>
...<v2:typeRef>_cla:ClaimDocProvenanceType</v2:typeRef>
...<v2:path>/car/value</v2:path>
...<v2:semantic>car value</v2:semantic>
</v2:typeSemantic>
</body></html>
```

In this example, the schema type ClaimDocProvenanceType (we omit references to namespaces and prefix except where necessary) is the type of the XML element that maps to this business concept, and specifically the child element addressed by the path /car/value refers to this concept. The semantics of car value comes from the business ontology. Given this mapping, we can safely interpret the fragment /car/value of all XML elements that are of this schema type as referring to the business concept car value.

XML elements of this type would materialize in the data being monitored as nodes and edges on the provenance graph that captures the process trace across systems. The provenance data model itself is described using a schema. Within this schema, the nodes and edges in a provenance graph are specified according to the schema types of the data that will constitute the content of the node or edge. Additionally, edges may have constraints: for example, an edge (such as an access edge) might have the constraint that the source of the edge must be a task, and the target of the edge must be data. Other constraints can limit the schema types of the source or target and so on. Here is an example of a part of the provenance data model:

```
<html><head></head><body>
...<v2:typeRef>_cla:ClaimDocProvenanceType</v2:typeRef>
...<v2:path>/car/value</v2:path>
...<v2:semantic>car value</v2:semantic>
</v2:typeSemantic>
</body></html>
```

The above XML fragment states that the provenance graph will contain a node; of class data (other possible classes are resource, task and custom). The node also says that its content will have schema type ClaimDocProvenanceType. Producers and consumers of provenance data can use this information to validate the data being recorded into the provenance store, or queried from it. Note that ClaimDocProvenanceType is the type with which business level semantics were associated above. Given this information about the data model and the semantic mapping,
we can now have a correspondence between the business artifact *car value* and a fragment of the contents of a provenance node.

```xml
<v2:edge>
  <v2:element_>cla:actorOf</v2:element>
  <v2:sourceClass>resource</v2:sourceClass>
  <v2:targetClass>task</v2:targetClass>
</v2:edge>
```

This XML fragment states that the provenance graph will contain an edge whose content will be described by the schema type *actorOf*. Additionally, the source of the edge within the provenance graph will be of class *resource*, and the target will be of class *task* (indicating that a particular resource performed a particular task).

The semantic mapping consisting of an XML document with information on the provenance data model, the business ontology and the mapping between the business and IT levels is stored in the system. As is evident from the architecture shown in Figure 1, this is what enables business users to then create policies using the business vocabulary they are accustomed to, and enables the system to transparently map these into business rules that check for the desired compliance conditions, a process that is described in section C.

### A. Rule mapping and deployment

Once the business rules are defined in the Policy Authoring Tool, they are ready to be deployed to the business provenance management system. During the deployment process, the natural language policies are transformed to a format that the business provenance system can consume. This format is then mapped to a set of rules that run within the business provenance system. The rules are designed to continuously monitor compliance conditions and create control point nodes to represent the current compliance status. We will illustrate this process with a simple example. Consider the following business policy: “Each claim must have an initial damage estimate”. This policy will be exported to the provenance graph along with some metadata that will identify the business level concepts embodied in the policy. In this case, *claim* and *initial damage estimate* are business concepts used in the policy for which we should be able to look up a mapping and discover the corresponding IT artifacts. In this particular example, it turns out that the concept *claim*, as described in the semantic mapping, is represented at the IT level by XML messages of type *ClaimDocument*. The concept *initial damage estimate* in the business vocabulary is similarly mapped to the IT level as an artifact. Additionally *must have* has a special interpretation as a standard business concept that can be interpreted by the business provenance system without an explicit mapping. It means that the two concepts *claim* and *initial damage estimate* must be related to each other (the other standard business term of this type is *may have*, wherein the concepts are not required to be related to each other, but may be). Based on this information, the rule requires the provenance system to continuously monitor business events the pattern shown in Figure 4.

Once the pattern to be searched for is inferred, the business provenance system will create two business rules. Each of these rules will result in the creation or update of a control point node. The control point node will be labeled based on the name of the corresponding business policy. In this case, if the policy is named ‘BP103.1’, we would have a correspondingly named control such as ‘KCP_BP103.1’. Here KCP stands for Key Control Point, a common term used to describe a compliance monitoring instrument. It will have a *status* field set to *compliant* or *defect* based on the current compliance status for this policy. The two rules created in this case will be as follows:

**Rule A: Test for non-compliance**

**Triggering condition:** presence of a *ClaimDocument* data node

**Action:** Creation of a node labeled *KCP_BP103.1* with status set to *Defect*, and creation of an edge to the *ClaimDocument* data node

**Rule B: Test for compliance**

**Triggering condition:** presence of a *ClaimDocument* data node linked to an *initialDamageEstimation* data node via an edge

**Action:** Update of the node labeled *KCP_BP103.1*; the status is set to *Compliant*, and an edge to the *initialDamageEstimation* data node is created

These rules are automatically generated by the business provenance system on deployment of the policy. Once generated, they immediately become operational and will monitor incoming data, continuously attempting to fire the rules and create the corresponding KCP nodes. Note that there is some dependency between the rules; Rule B is required to run after Rule A in this case. This is taken care of during the rule generation process so that the system will run these in the proper order. Thus, the effect of deploying the business policy is that the system now monitors the compliance condition continuously and through the production of a KCP node linked to the appropriate evidence makes the compliance status amenable to query via the dashboard.

### VII. Dashboard

The dashboard provides business users with a single point where they can access information about compliance with respect to defined business policies. The dashboard
itself queries the provenance store for all known control point instances (Figure 5). Each control point instance lists its compliance status and a summary explanation. The control point types and explanations are directly derived from the business policy created by the user, so the user need not translate between the business policy artifact (i.e. the intended policy) and the control point (i.e. the monitoring mechanism). Additionally, control points are linked in the provenance graph to relevant nodes that provide evidence to support the status. The dashboard groups together controls of a single type to create gauges that provide information on compliance percentage for each type of control. Users can click on a gauge to get details of the various control point instances that are summarized, and then explore each control point. This presentation makes it easy for users to tell at a glance what type of control has the lowest compliance percentage, and then investigate individual instances and examine corresponding evidence to understand why.

Figure 5  Visualizing the artifacts of auto claim process

Figure 6 shows the dashboard visualization that displays the artifacts of the auto claim process simulation where one control point (KCP1) is deployed. Different icons represent different business entity types. Attributes of each entity can be examined by clicking over the associated icon. The gauge depicted in Figure 6 displays the compliance status of process instances against KCP1. There are 3 process instances which are not compliant against the business rule: “Each claim must have an initial damage estimate”. For these process instances InitialDamageEstimation data item does not exist. Hence, the gauge shows 70% compliance.

VIII. AUTO INSURANCE CLAIM SCENARIO

We tested our approach by using an auto insurance claim process scenario. We first modeled the auto insurance claim process by using WebSphere business modeler [18] and the created a java simulation of the application based on the model. The highlights of the insurance claim process simulator are given below. The details of the process model we created for the scenario is left outside the scope of this paper.

Figure 6  Dashboard shows the status of deployed business rules

Figure 7  Part of provenance graph with associated control points

Figure 7 depicts a section of provenance graph generated by capturing IT event data from the simulator and mapping them as provenance data elements as described in section Business Provenance Management System. The nodes of the sub-graph are the relevant business artifacts and the edges show the relations among them. The hexagonal shapes are the resources (actors), rectangular shapes with gear icons are tasks and other rectangular shapes with cylindrical icons are the data elements. In the simulation scenario, claim handler determines if the car should be totaled based on the damage estimation provided by the adjustor. The claim document is updated as a result of performing the task labeled “determine if car should be totaled”. KCP1 checks if claim handler’s decision to total or repair is supported by damage estimation created by the adjustor. Hence, it is connected to the “claim handler”, “determine if car should be totaled” task, “damage estimate” and the “claim document”. The Key Control Point can carry out computations to answer questions like who made the determination to total the car and when, whether there was enough supporting evidence and whether appropriate evidence is reflected in the claim document..
One example of how simulation generates exceptional conditions is as follows: when there is an accident in certain areas, the simulator has the claim handler select a particular adjustor and an auto repair shop as his collaborators. He sends the car for a repair even though the repair cost is more than the totaled value of the car. As a result a process execution trace is generated with violations. We then use these traces to check if the business rules created and deployed automatically can detect these violations.

IX. CONCLUDING REMARKS

Managing the integrity of a business requires the management of the individual as well as the interdependent aspects of policies, processes and information. These are the core entities in an organization. This work is an attempt to demonstrate how we can build a bridge between these core entities by integrating policy authoring tools (policy) with business provenance data management systems (process, information). One reason why there is a disconnection between business and IT is that the business language of process modeling effort is lost when the processes are implemented. The details of operational rules are buried in the implementation code. Hence, semantic mapping of the IT data to business language was an essential building block of constructing a bridge that must be part of data modeling effort at the time of code development. Our future work will focus on developing techniques to automate this mapping at the time of business modeling and code implementation.

Another important aspect of the bridge is to develop a generic data model for IT data that can be extended for different businesses. Grouping the business artifacts into “data”, “task”, “resource” and “relations” types enables capturing and correlating different aspects of the business.

Using templates in policy authoring tool helped us avoid natural language processing of the rules before they are mapped onto the IT data. This approach, however, limits the types of business rules that could be instantiated since they depend on the number of available templates. We are also looking into creating rule structures by using ILOG Business Rule Management System for more flexible templates.

Once the rules are created and mapped onto internal control points and deployed, our system is fully automated to display the compliance results. Mapping business rules onto internal control points, however, is done by the experts manually. Our future research will try to solve this problem, by generating methodologies to create internal controls automatically from templates.

ACKNOWLEDGEMENTS

We would like thank Francisco Curbera for his help to define the problem of integrating provenance solution with rule authoring systems, Alexander Slominski and Paul Keyser for helping to develop the automobile insurance claim scenario code and Axel Martens for providing the visualization interface.

REFERENCES


