technische universität
dortmund

Using Ontologies to Analyze Compliance Requirements of Cloud-Based Processes

T. Humberg, C. Wessel, D. Poggenpohl, S. Wenzel, T. Ruhroth, J. Jürjens

Abstract

In recent years, the concept of cloud computing has seen a significant growth. The spectrum of available services covers most, if not all, aspects needed in existing business processes, allowing companies to outsource large parts of their IT infrastructure to cloud service providers. While this prospect might offer considerable economic advantages, it is hindered by concerns regarding information security as well as compliance issues. Relevant regulations are imposed by several sources, like legal regulations or standards for information security, amounting to an extend that makes it difficult to identify those aspects relevant for a given company. In order to support the identification of relevant regulations, we developed an approach to represent regulations in the form of ontologies, which can then be used to examine a given system for compliance requirements. Additional tool support is offered to check system models for certain properties that have been found relevant.

The project ClouDAT was supported by the Ministry of Innovation, Science, Research and Technology of the German State of North Rhine-Westphalia and EFRE under grant number 300267102

Using Ontologies to Analyze Compliance Requirements of Cloud-Based Processes

Thorsten Humberg[2], Christian Wessel[1], Daniel Poggenpohl[2], Sven Wenzel[2], Thomas Ruhroth[1], and Jan Jürjens[1,2]

[1] Chair of Software Engineering, Technical University Dortmund, Germany
{christian.wessel, thomas.ruhroth}@cs.tu-dortmund.de,
http://jan.jurjens.de
[2] Fraunhofer Institute for Software and Systems Engineering, Dortmund, Germany
{thorsten.humberg, daniel.poggenpohl, sven.wenzel}@isst.fraunhofer.de

Abstract. In recent years, the concept of cloud computing has seen a significant growth. The spectrum of available services covers most, if not all, aspects needed in existing business processes, allowing companies to outsource large parts of their IT infrastructure to cloud service providers. While this prospect might offer considerable economic advantages, it is hindered by concerns regarding information security as well as compliance issues. Relevant regulations are imposed by several sources, like legal regulations or standards for information security, amounting to an extend that makes it difficult to identify those aspects relevant for a given company. In order to support the identification of relevant regulations, we developed an approach to represent regulations in the form of ontologies, which can then be used to examine a given system for compliance requirements. Additional tool support is offered to check system models for certain properties that have been found relevant.

Keywords: Cloud Computing, Compliance, Business Processes, Risks, Ontologies.

1 Introduction

Cloud computing enables reduction of costs and gains flexibility by outsourcing hard- and software, but running business processes in such environments poses new issues with respect to compliance and security compared to locally hosted solutions. Legal regulations such as data protection laws or the European directive Solvency II cause additional requirements to business processes and underlying software systems. Hence, before outsourcing a process or parts of it into the cloud, the processes and systems have to be checked with respect to these requirements.

In order to support compliant and secure outsourcing of business processes into cloud environments, we developed a two-step approach based on ontologies. The concept of ontologies is used to formalize various standards that contain regulations regarding IT-security as well as compliance aspects. We show how

this can be applied to capture the content of these sources in a unified way, and detect dependencies and references between different source documents. Using this basic regulatory ontology, we enhance it with further semantic information concerning the actual aspects dealt with in particular parts of the regulatory documents.

When using this collection of information in an actual analysis of a system or process model, it is possible to identify specific situations that require further analysis to test if the constraints they impose on a process are met in the model. The constraints can be used to suggest corresponding examination methods from a repository of possible (automated) compliance or security checks.

The approach is supported by different analysis tools we developed. They are integrated into the model-based environment CARiSMA.

In the following section, we give an overview about the background concepts relevant to our approach. Section 3 presents the internal structure of our ontology as well as to how it can be created based on diverging sources of input. After formalizing the information, it can be used to investigate given models for situations to be considered (see Section 4). Section 5 presents various tools that have been integrated in the model analysis tool CARiSMA to support our analysis process. In Section 6 we present our ongoing work on the methodology. Related work is presented in section 7, followed by a discussion of our approach as well as possible future developments in Section 8.

2 Background

This section introduces some terms before we explain our approach of ontology-based compliance formalization and analysis of business processes in the next section.

2.1 Cloud Computing

Cloud computing is generally regarded to be one of the major developments in information technology in recent years. Substituting existing self-maintained hard- and software with resources rented on an on-demand basis offers significant potential benefits, in particular a possible reduction of costs as well as a gain in flexibility. Especially for small and medium-sized businesses (SMB) it might be an appealing alternative to maintaining their own data centers. The American *National Institute of Standards and Technology* (NIST) subdivides Cloud Computing into three service levels. *Infrastructure as a Service (IaaS)*, is the lowest layer providing basic virtual hardware resources. This may include virtual machines or networks. *Platform as a Service (PaaS)*, represents a middleware for writing distributed and scalable software, which resides upon the IaaS layer. *Software as a Service (SaaS)* is the top-most layer with ready-to-use software which was built using PaaS solutions.

Depending on the chosen layer type, different security needs have to be considered. If an IaaS structure was chosen, only the bare virtual hardware is provided. All security needs have to be installed and checked starting from the

operating system and ending with the software that is executed on it. All security requirements are in the hands of the customer of the cloud service. With PaaS it is necessary that the software developed on top of it meets the security standards needed for the business process. On the SaaS layer software must be chosen, that complies with the security needs and meets the compliance requirements.

The required security level further depends on other characteristics like the deployment model, e.g., private cloud versus public clouds. While private clouds can be considered relatively secure against attacks from outside they are exposed to internal attackers. Public clouds on the other hand need to be protected against attacks from the outside.

2.2 Risk & Compliance

In this paper we speak of risks meaning the components mentioned in the *IT-Grundschutz Catalogues* [1]. We do neither consider probability of occurrence nor amount of damage caused by them. Risks in our understanding are IT security related while compliance is seen as laws or internal regulations which have to be held. There exists a tight bounding between security risk, security and compliance as for example compliance rules are not met when an attacker is able to retrieve personal data from a not properly secured server.

Risks In this context a risk is defined as an IT security related vulnerability in a business process which is executed in a cloud environment. Examples are unsecured communication channels between cloud hosts, insufficient rights management or processing of confidential data.

Compliance A security analysis of the cloud computing environment should be carried out before a security requirement analysis is performed on the business operation. This will yield the maximum number of security requirements that can be met.

The conformance to compliance regulations should be audited on three levels.

Process & compliance analysis: Documents from which business processes can be derived should be analyzed. Our approach considers processes given in form of process models (e.g. UML activity diagrams or BPMN models). The risk analysis also works on less structured documents such as textual process descriptions or log files from which processes can be mined.

Design time compliance: The implementation of a business process has to comply with legal regulations and company policies. The cloud interface, the activities performed within the cloud, and the data flows between cloud and user are the major points of interest. Furthermore, the overall consistency f the processes with respect to compliance requirements should be verified.

Runtime compliance: It has to be ensured that all compliance-relevant and critical processes (esp. those outsourced into a cloud) are monitored and logged. Such a monitoring can be performed using business process mining and conformance checking [2]. Our approach enables the identification of such processes. The runtime analysis itself is not further considered in this paper.

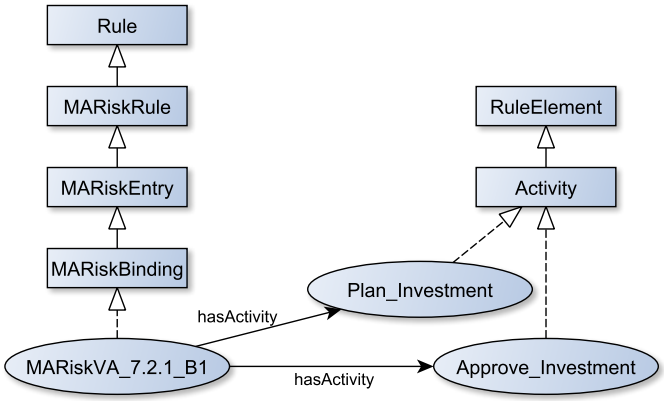

Fig. 1. Part of the ontology's class structure: ***Rule***, ***RuleElement***, and subclasses.

2.3 Ontologies

Ontologies are a widely used tool to describe incomplete information about an area. In contrast to models having a closed world assumption, ontologies employ an open world assumption, i.e. they can be extended arbitrarily. In this paper we use the *Web Ontology Language* (OWL) [3]. OWL knows three profiles which differ in expressiveness and the possibility to reason about facts in the ontology. The least expressive profile is OWL-Lite. The OWL profile OWL-DL is semantically equivalent to the Description Logic [4]. And OWL-FULL allows on one hand the largest expressiveness but is on the other hand not always decidable.

In this paper we use OWL-FULL for the sake of simplicity. It allows us to use smaller ontologies in our context.

A graphical representation of an ontology is depicted in Fig. 1: The basic elements of ontologies are the classes (also called concepts, depicted as rectangles, e.g., ***Rule***) and the individuals (ovals, e.g. *Plan_Investment*). Classes can be seen as a set of individuals sharing common properties. The membership of an individual to a class is depicted using a dashed line with a triangle arrow. Thus *Approve_Investment* is an individual in the class ***Activity***. The concepts can be related by an is-a relation (solid line with triangle arrow, e.g. ***MARiskRule*** is-a ***Rule***), describing that all individuals of a class also belong to the superclass. Classes can be defined directly or by operations like the intersection of classes.

Relationships between different individuals are expressed with roles. Roles are used as a directed property, hence, we use an arrow as a graphical representation (see. Fig. 1) together with a descriptive name.

3 Regulatory Ontology

The principle of ontologies is especially suited for formalizing compliance regulations. With the classification provided by distinct concepts, information can be

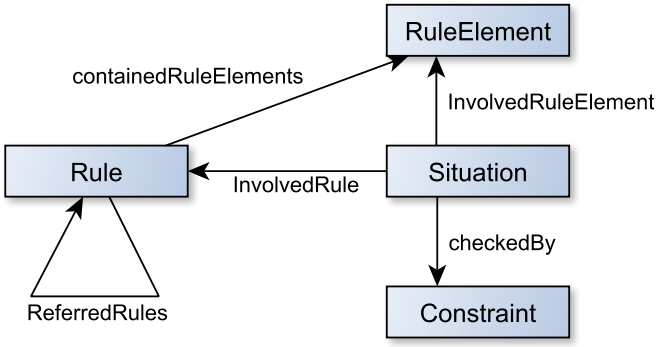

Fig. 2. Main concepts of the regulatory ontology.

represented in a structured way, allowing for its utilization in automated analysis methods. At the same time, it is still possible to capture the diverse structures found in different types of relevant input standards. Furthermore, it is possible to express relations between entities across different inputs.

An outline of our ontology used to store the compliance information is given in Figure 2. All classes to be used are derived from one of the depicted groups.

Rule Individuals of this class and its subconcepts are used to store the information directly imported from the source documents.

RuleElements contain semantical enhancements, describing different aspects a rule can refer to, e.g. referred activities and roles.

Situation represents a more complex setting given by a rule, e.g. the referenced concept of Separation of Duty.

Constraint indicates how a specific situation can be (automatically) checked in a given model.

In the following sections, these classes are explained in more detail.

3.1 Rules

Depending on the scope of the analysis, different standards are relevant. Some are focused on IT-security, e.g. the ISO 27k series [5] or the IT-Grundschutz Catalogues [1] published by the German Federal Office for Information Security (BSI). Particularly important in the field of cloud computing are regulations concerning compliance issues. Examples include national laws regarding data privacy and regulations relevant to specific sectors, e.g. Solvency II for financial institutions.

Regulations are generally given in the form of textual descriptions, with differing internal structures, thus being inconvenient for automated processing. To overcome these restrictions, it is necessary to represent their content in a unified way.

The ontology's concept of classes with inheritance offers a method to capture those structures in a way that on the one hand preserves this information, but on the other hand still provides enough uniformity to use automated analysis methods.

Two types of classes are used at this stage: Those that contain the actual content, and others that represent the *structure* the content is organised in. Both classes are derived from ***Rule***, but only the classes forming the content are evaluated in our automated processing.

Additionally, object properties are used to capture references between rules. These references may exist between rules within the same standard as well as between different input sources.

An example is shown in Figure 1: The left hand side shows how the concept ***MARiskBinding*** is derived from ***Rule*** via several steps, and instantiated by the individual *MARiskVA_7.2.1_B1*, representing a specific entry from the document.

Creating the basic individuals representing the content of a standard is mainly a transformation. The generation of this stage can in general be fully automated, depending on the format the input is available in. For example, the catalogues provided by the BSI are available in HTML format and allow for an automatic conversion into the ontology.

3.2 Rule Elements

Based on the mere representation of the content, the ontology can be enriched with further details, namely *Rule Elements*.

An individual of the concept ***RuleElement*** (or subconcept thereof) can constitute any aspect relevant for a rule contained in the ontology.

Currently we define five types of elements. New concepts can easily be added as required.

Artifact: Representing objects a standard refers to, e.g. documents or IT hardware

Role: Specific roles mentioned, e.g. executives or data protection officer

Activity: Certain action, i.e. from a business process model, e.g. reporting required information

Process: A more complex operation, potentially involving several activities

Property: Requirements demanded by a standard, e.g. confidentiality or non-repudiation

The usage of rule elements directly yields the possible application of related properties: There are property relations from individuals of class ***Rule*** (or subclasses thereof) to those individuals representing rule elements that are relevant for the particular rule.

Object properties between instances of rule elements can be added to the ontology, but are currently not evaluated in our approach.

In contrast to converting the text content, identification of rule elements requires the interpretation of the textual representation. Given this problem, it is not yet possible to fully automate this step. As a first foundation we defined a set of typical elements with associated keywords. Identifying those keywords, or synonyms thereof, can be used to generate occurences of specific elements.

In order to fully utilize the advantages of the approach, a manual enhancement is still necessary. A specialized tool for this purpose is integrated into our environment and described in Section 5.

3.3 Situations and Constraints

Based on rule elements, we define *situations* as generic patterns that can occur in an examined model. Within the ontology, a situation links to a set of rule elements (via the *InvolvedRuleElement*-property). The presence of those elements in a model indicates the possible relevance of the particular situation. Additionally, an object property links to the particular rules the situation originates from.

A popular example is the need to implement the *separation of duty* pattern within a business process: In this case, a pair of activities must not be executed by the same person. The involved elements in this case are instances of the two activities. This is modeled in the ontology as shown in Figure 3: *SeparationOfDuty* is an individual of the concept ***Situation***, described in the ***Rule*** individual *MARiskVA_7.2.1_B1*. It forbids the activities *Plan_Investment* and *Approve_Investment* to be carried out by the same person. As shown in Figure 1, both activities are instanciated from the RuleElement concept ***Activity***.

To contain this specific situation, an actual business process model must thus contain two activities corresponding to *Plan_Investment* and *Approve_Investment*. Details on how those are identified are given in section 4.

Though a situation does not in itself contain information about how it is to be tested on an actual model, it can reference to individuals of the class ***Constraint***. If a process contains a situation, the corresponding constraints of the situation have to be respected for the process to be secure. The constraints can be further parameterized using rule elements. Only rule elements applicable to the situation, i.e. associated with the rules involved in it, can be used as parameters. Constraints can vary in their implementation, ranging from textual descriptions to automated checks available in the CARiSMA framework.

Formalizing possible situations and assigning appropriate constraints is a manual task as well. It consists of identifying the necessary components and creating a new individual for this situation.

The generation of this compliance ontology is a general preparation step for our process analysis. Regardless, it is only defined by the considered input standards.

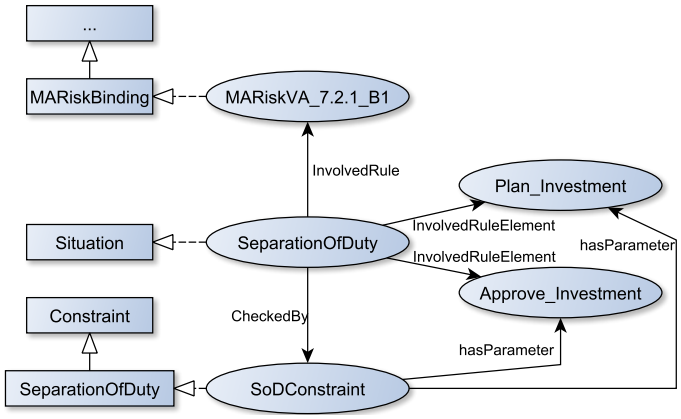

Fig. 3. Extract from the ontology, representing the Situation "Separation of Duties".

4 Process Analysis

In the previous section we showed how the ontology used by the analysis process is designed and created to support the analysis of business processes. In this section we explain how compliance and regulatory requirements in business processes are identified and verified. The input for this process is a formal model of the business process in form of e.g. a BPMN model and the ontology containing the necessary regulations.

The analysis itself consists of three steps, namely:

1. Identification of relevant rule elements (i.e. situations in which this rule elements occur) within the regulations applicable to the business process that should be analyzed.
2. Mapping of the business process model elements to rule elements of the applicable situation.
3. Verification of the rule element constraints imposed by the situation. An example would be a check testing whether a *separation of duty* constraint on the business process is met.

Figure 4 gives a global view over the analysis process. The process is discussed in more detail in what follows.

Identification of relevant rules For the first step in the analysis process it is necessary to identify situations (see Section 3.2) in the process model which have to be checked. As shown in the previous section a situation consists of one or more rules and part of the rule elements present in these rules, which may be involved persons (roles), objects or activities.

For example in the separation of duty constraint mentioned before, there are two actions involved: "*Plan_Investment*" and "*Approve_Investment*". As these are individuals of the concept ***Activity*** they can be seen as keywords during

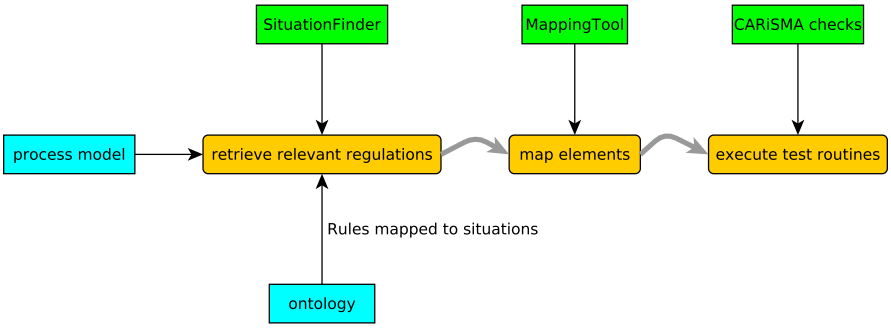

Fig. 4. Business Process Analysis Overview.

a search for the situation "separation of duty (SoD)". The idea behind this is to test the names of the activities of the business process model for accordance with individuals of the ontology. In this step all labels of all model elements (e.g. text comments, labels on transitions, etc.) of the analyzed process are tested.

If all rule elements of a situation can be found among the labels of a business process model, the according checks for this situation will be suggested to the user. Since the search for an exact match between an activity name and a name of an individual may be ineffective, we utilize word databases which consider co-occurrent words and synonyms of an activity name. Thereby we utilize the following scoring system.

First we define a constant score s. If a word of a label is found directly as a keyword of a i-th rule element it will produce a score $s_i = s$. We define a score $s_i = \frac{s}{2}$ for matches found via synonyms of the word of the label because it is not the original word but at least semantically equivalent. Matches found with co-occurrent words have the lowest score of $s_i = \frac{s}{4}$ since the original word of the label is not involved any more. The final score S is then calculated with $S = \sum_{i=1}^{n} s_i$ where n is the number of rule elements which are involved in the specific situation. M is defined as the maximum reachable score, which is built using $n \times s$, where n is the above-mentioned number of rule elements the situation is built of. We avoid too many false-positive matches by discarding those below a certain threshold t. Therefore only matches which satisfy $M - S < t$ are considered for the subsequent analysis.

In order to get an almost complete set of applicable situations for a given business process model we expand the result by other potential situations which may be of interest to test. Usually regulations have cross references to other associated regulations. These can be found easily utilizing the ontology described in the previous section.

For example the SoD mentioned above is a *Situation* individual which has an involved rule (an object property) *MaRiskVA_7.2.1_B1* which is a ***MARiskBinding*** individual. *MaRiskVA_7.2.1_B1* may have object properties to instances of

other rules. Situations associated to these rules are added to the result set and their corresponding checks are additionally recommended to the user.

Mapping of elements Every situation has one or more constraints that have to be met by the analyzed process model. Some of these constraints could be verified with the aid of specialized tools, e.g. CARiSMA. In order to realize tool-based checking of situations it is necessary to specify the parameters, i.e. to name the actors, the activities and objects used in a specific situation.

Therefore the second step in the analysis process is to map the elements of the business process model to parameters of the check. This is a manual task, because model elements can be named individually by the modeler of the process. For example the name of the *applicant* role in the separation of duty scenario may be an arbitrary value, which can not be determined automatically.

Constraint Checking After all model elements have been mapped to rule elements the constraints for a specific situation can be verified. This can be accomplished in various ways. A constraint may be verified by an automated test routine that systematically analyzes a given formal model, e.g. a BPMN or UML model with regard to whether specific requirements for a model are met.

Automated verification may be implemented as a CARiSMA check. In this case, the mapping of constraints of the ontology to corresponding checks in CARiSMA is stored in a file separate from the ontology. Doing this, the ontology itself remains reusable and uncluttered by implementation-specific information.

5 Tool Support

This section provides an overview of the various implementation efforts providing the tool support for creating and editing the regulatory ontology used during the analysis. The tools have been integrated with CARiSMA[3], our Eclipse-based environment for model analysis. First we describe the extractor used to create the ontology using various law documents. We then show the front end with which one is able to create the various other concepts and relations of the ontology. The enriched ontology can be used during process analysis to determine the checks to use during an analysis. The CARiSMA *Situation Finder Check* is one possible way to identify applicable situations. The process tool chain is visualized in Figure 5.

Law Extractor This tool parses regulatory documents of various formats to create the rule individuals of the regulatory ontology. Currently supported are the MARisk [6], the BDSG [7] and BGB [8], and the BSI catalogues [1]. The resulting OWL ontology can then be modified using the following tools.

[3] http://carisma.umlsec.de

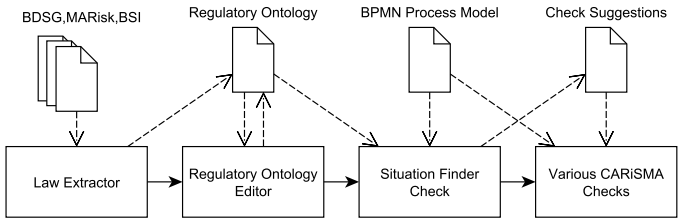

Fig. 5. Regulatory Toolchain.

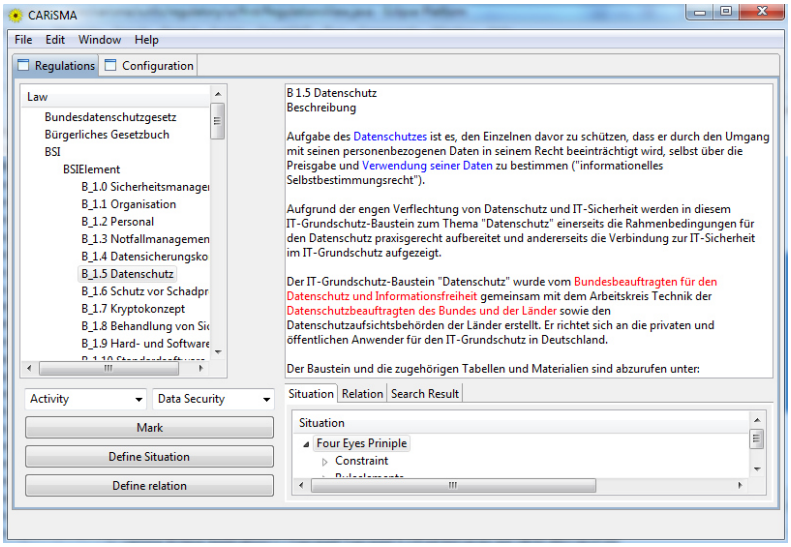

Fig. 6. Regulatory Ontology Editor.

Regulatory Ontology Editor After extracting the rules, the ontology can then be enriched with the regulatory ontology editor depicted in Figure 6. Text passages can be marked and stored as individuals of the rule element subconcepts. Situations can be created by selecting the rule elements that comprise the situation and naming it appropriately. Constraints that can be imposed on situations are parameterized with rule element subconcepts. Constraint individuals can then be assigned to situations, using the situational rule elements to set the parameters of the constraint.

In addition to the integrated GUI, we have implemented a web-based version provides the possibility for distributed access to the ontology.

Situation Finder Check There are many possible solutions to determine which situations apply during the execution of a given process. As an exemplary implementation of the methods for situation identification the Situation Finder checks if a given BPMN process model contains all the rule elements that define

a situation. If all are present in the process, the check then outputs the appropriate checks that should be run on the given model in order to verify its validity with regards to the regulations. Other methods for identifying situations could potentially involve references between rules or weighted approximations based on the types of rule elements in a situation.

Checks for Process Analysis CARiSMA provides some checks that support the automated process analysis of models. For example, the check for the abovementioned SoD constraint analyzes whether the conflicting actions are executed by different actors. In the case of a BPMN model this is achieved by checking the lanes the actions are embedded in. If the actions are in different lanes, the actors are different and the test passes. If the actions are in the same lane, the test fails and the check reports this to the user. Success or failure of the check is reported in two ways. On the one hand a textual report is generated and on the other hand the user is informed in a graphical manner by coloring of the failed actions directly in the diagram.

6 Support for Evolving Requirements

At the moment we are working on extending the approach for the use and adaption of referenced compliance documents. Many companies have internal guidelines which are built on top of existing guidelines like the BSI IT-Grundschutz Catalogues. So they can define which parts are implemented as well as the definition of new internal guidelines. Also the adaption and modification of the used guidelines is applied. Since each of these referenced or included documents can change interdependently we need a system to tackle the evolution and reaction of these evolutions.

To implement this needs, we use a technique under development in the SecVolution project [9, 10] called *layered ontologies*. Layered ontologies are an extension of the OWL import function. The OWL import allows to import OWL ontologies and thus the extension of the imported ontologies. We extend this approach by adding *modification operators* such that parts of the imported ontologies can be hidden or changed:

The hide operation is used to remove parts of the imported ontologies and the change operator can change values in the ontology.

Sometimes the imported ontologies (resp. guidelines) are built using the layered ontologies approach itself. Here, we have a problem if parts of the ontologies can be hidden by a higher layer used by the current layer. Therefore, we include an unhide operation, which reverts the effect of a hide operation (see Fig. 7). Similar, we define a reset operation for undoing the change operation.

Compatibility with the described approach is maintained by expanding all ontology imports and modifications into a single flat separate ontology. Thus this approach can be easily integrated into the approach from the leading sections.

The layered ontologies approach also includes techniques to deal with evolution of imported guidelines. Whenever an imported ontology is changed, this

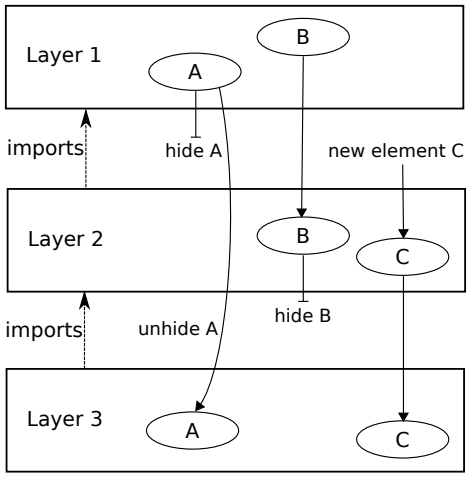

Fig. 7. Example depicting the hide and implicit add operation within three layers of ontologies.

modification can be analyzed and the using layers are semiautomatic co-evolved. User interaction is needed to provide decisions in ambiguous evolution cases as well as when new information like names are needed.

7 Related Work

The fulfillment of compliance and security requirements in business processes is essential to receive acceptance from customers. An approach to encode and check security requirements in BPMN models has been presented by Wolter et. al. [11]. However, these requirements focus only on closed systems.

Cloud computing, which has lasted on the peak of Gartner's technology hype cycle [12] for quite a while now, leads to outsourcing of processes into heterogeneous environments. The use of cloud computing technologies offers economic potential for small and medium-sized enterprises. However, serious doubts wrt. security and compliance exist [13]. Hence, security approaches for closed systems are not eligible here. Menzel et. al. [14] propose an approach to define security requirements on service orchestration level.

CloudCycle[4] is a project related to our approach. It focuses on cloud providers and offers services that allow them to guarantee their customers that they are compliant with security policies and further regulations. The approach of CloudCycle is a suitable complement for our approach. Once business processes are successfully outsourced into the cloud their security and compliance can be monitored.

[4] http://www.cloudcycle.org

Ontologies for cloud computing and cloud security have been presented by Gräuler et. al. [15]. They analyzed the different sources of risks within cloud computing environments and manifested them in an ontology. Based on that ontology, they provide a database of cloud providers that allows users to select a provider based on certain security properties. This is especially interesting for finding a suitable cloud provider after potential risks of a business process have been revealed by our approach.

Tsoumas and Gritzalis provide an ontology-based approach to organize security knowledge [16]. It is designed to enable reuse of knowledge and map requirements to implemented controls of a system. A similar approach to formalize security knowledge has been presented by Fenz and Ekelhart [17]. It focuses on representing security domain knowledge and corporate knowledge in an ontology. While we provide a systematic approach to represent the regulatory documents and to extract security or compliance requirements, the above-mentioned approaches consider only the modeling resulting security knowledge. It would be interesting to consider an integration of those approaches such that they could be used to represent the knowledge that is extracted by our approach.

Peschke et. al. [18] present the *RiskFinder* which is a precursor of our risk analysis component. It analyses UML models with respect to security relevant vocabulary. Schneider et. al. propose a heuristic search based on Bayesian filters [19]. HeRA realizes a feedback-driven approach for security analysis during requirements engineering [20]. These approaches provide powerful rules, however, they work only on single words and do not consider language databases.

Our view of IT security risks corresponds to the use in the BSI IT-Grundschutz Catalogues [1], which does not include concrete values for probabilities and possible extend of damage (or benefit) of risks. In the terminology defined by other standards, this information is included, e.g. in the ISO 27000 series [21] and NIST standard 800-39 [22].

8 Conclusions & Outlook

When outsourcing business processes into cloud environments, problems regarding security and compliance still represent major obstacles. The methodology described in this paper aims at supporting users in examining models of their systems and processes for potential risks. While a completely automated analysis still appears far from feasible, our approach and tools can aid in highlighting aspects that require further examination, either manually or tool-supported.

Using ontologies, one can take advantage of a very flexible, yet formalized, way of representing information, making it accessible for automated procedures.

Beside continuing our research regarding evolving requirements, several points seem worth considering to further develop our concept. Those include enhancement of tool-support, improvement of existing heuristics for the detection of matchings as well as support for established methods in knowledge systems, e.g. automated reasoning.

References

1. Bundesamt für Sicherheit in der Informationstechnik: BSI-Grundschutz Katalog (2006)
2. van der Aalst, W., Reijers, H., Weijters, A., Vandongen, B., Alvesdemedeiros, A., M. Song, H.V.: Business process mining: An industrial application. Information Systems, Vol. 32, No. 5, pp. 713-732. (2007)
3. W3C OWL Working Group: OWL 2 Web Ontology Language: Document Overview (Second Edition). W3C Recommendation (11 December 2012) Available at http://www.w3.org/TR/owl2-overview/.
4. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F., eds.: The description logic handbook: theory, implementation, and applications. Cambridge University Press, New York, NY, USA (2003)
5. ISO/IEC: ISO27001: Information Security Management System (ISMS) standard. Online: http://www.27000.org/iso-27001.htm (October 2005)
6. Bundesanstalt für Finanzdienstleistungsaufsicht: Mindestanforderungen an das Risikomanagement - MaRisk (October 2012)
7. Bundesrepublik Deutschland, vertreten durch das Bundesministerium der Justiz, v.d.d.B.d.J.: Bundesdatenschutzgesetz (December 1990)
8. Bundesrepublik Deutschland, vertreten durch das Bundesministerium der Justiz, v.d.d.B.d.J.: Bürgerliches Gesetzbuch (August 1896)
9. SecVolution Webpage: http://www-secse.cs.tu-dortmund.de/secse/pages/research/projects/SecVolution
10. Jürjens, J., Schneider, K.: Beyond one-shot security. In: Modelling and Quality in Requirements Engineering (Essays Dedicated to Martin Glinz on the Occasion of His 60th Birthday), Verlagshaus Monsenstein und Vannerdat (2012) 131–141
11. Wolter, C., Menzel, M., Meinel, C.: Modelling security goals in business processes. In: Modellierung. (2008)
12. Dixon, J., Jones, T.: Hype cycle for business process management. Technical report, Gartner Study (2011)
13. BITKOM: Cloud-Computing - Evolution in der Technik. Technical report, BITKOM (2009)
14. Menzel, M., Thomas, I., Meinel, C.: Security requirements specification in service-oriented business process management. In: ARES. (2009)
15. Gräuler, M., Martens, B.; Teuteberg, F.: IT-Sicherheitsmanagement im Cloud Computing - Entwicklung und Implementierung einer Ontologie. In: Proceedings zur INFORMATIK 2011. (2011)
16. Tsoumas, B., Gritzalis, D.: Towards an Ontology-based Security Management. In: Proceedings of the 20th International Conference on Advanced Information Networking and Applications (AINA). Volume 1., IEEE (2006) 985–992
17. Fenz, S., Ekelhart, A.: Formalizing information security knowledge. In: Proceedings of the 4th International Symposium on Information, Computer, and Communications Security (ASIACCS), New York, New York, USA, ACM Press (2009) 183
18. Peschke, M., Hirsch, M., Jürjens, J., Braun, S.: *Werkzeuggestützte Identifikation von IT-Sicherheitsrisiken.* In: D-A-CH Security 2011. (2011)
19. Schneider, K., Knauss, E., Houmb, S., Islam, S., Jürjens, J.: *Enhancing security requirements engineering by organizational learning.* Requirements Engineering (2011) 1–22 10.1007/s00766-011-0141-0.

20. Knauss, E., Lubke, D., Meyer, S.: *Feedback-driven requirements engineering: The Heuristic Requirements Assistant*. In: Proceedings of the 31st International Conference on Software Engineering. ICSE '09, Washington, DC, USA, IEEE Computer Society (2009) 587–590
21. ISO/IEC: ISO27005: Information technology - Security techniques - Information security risk management. Online: http://www.27000.org/iso-27005.htm (June 2008)
22. NIST, Aroms, E.: NIST Special Publication 800-39 Managing Information Security Risk. CreateSpace, Paramount, CA (2012)