Institute of Architecture of Application Systems

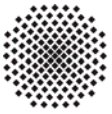
Compliance Domains: A Means to Model Data-Restrictions in Cloud Environments

Daniel Schleicher, Christoph Fehling, Stefan Grohe, Frank Leymann,
Alexander Nowak, Patrick Schneider, and David Schumm

Institute of Architecture of Application Systems,
University of Stuttgart, Germany
{schleicher, fehling, nowak, schumm, leymann}@iaas.uni-stuttgart.de

BibTeX:

```
@inproceedings {INPROC-2011-29,
   author = {Daniel Schleicher and Christoph Fehling and Stefan Grohe and Frank
Leymann and Alexander Nowak and Patrick Schneider and David Schumm},
   title = {{Compliance Domains: A Means to Model Data-Restrictions in Cloud
Environments}},
   booktitle = {Enterprise Distributed Object Computing Conference (EDOC)},
   publisher = {IEEE Xplore},
   institution = {Universit{\"a}t Stuttgart, Fakult{\"a}t Informatik,
Elektrotechnik und Informationstechnik, Germany},
   month = {August},
   year = {2011}
  }
```

Universität Stuttgart
Germany

Compliance Domains: A Means to Model Data-Restrictions in Cloud Environments

Daniel Schleicher*, Christoph Fehling*, Stefan Grohe*, Frank Leymann*, Alexander Nowak*,
Patrick Schneider§, and David Schumm*
*Institute of Architecture of Application Systems
University of Stuttgart
Germany
lastname@iaas.uni-stuttgart.de

§Fraunhofer Institute for Industrial Engineering IAO
Germany
Patrick.Schneider@iao.fraunhofer.de

***Abstract*—It is crucial for enterprises to execute business operations in a compliant way. This is especially true for IT-driven business processes as enterprises may face considerable fines when violating laws and regulation in their business processes. Through the advent of cloud computing, a new dimension of compliance requirements within the research area of compliant business process design has emerged. Data-sovereignty is one of the major compliance concerns enterprises have to deal with when moving applications and data to the cloud. Enterprises are fully responsible for their data, also when the data is not present within their IT premises anymore. This lead to the policy that specific data must not leave the IT premises of the enterprise.**

In this paper we present an approach to support the human process designer in modelling compliant business processes. We are focusing on compliance requirements which have to be considered in the field of cloud computing. These requirements have been created to meet laws and regulations. These laws and regulations are considering data which is to other countries, for example. Looking at the characteristics of these requirements, we deal with data-centric compliance rules here.

***Keywords*-Compliance; Business Process; Data-driven; Design-time;**

I. Introduction

Compliance of business processes has become more and more important for enterprises today. Compliance must be considered at every phase of the business process management life cycle. Beginning at the design phase, human business process developers should be supported in creating compliant business processes. Real world business process models tend to be quite complex, possibly consisting of hundreds of tasks [1]. Thus, it is hard to see for a human business process designer whether a business process is compliant with a certain set of compliance rules. The human business process designer should not have to bother with *manually* checking compliance during the design phase of a business process. Instead, the graphical design workbench should check each modification of the business process model whether certain compliance rules have been violated. Therefore, in the CAPE[1] project we develop a solution to automatically validate modifications of a business process model against compliance rules. In [2]–[4] we showed concepts to support the human business process designer in designing compliant business processes. These concepts were developed with control-flow-related compliance requirements in mind. An example for a control-flow-related compliance requirement is that two distinct persons have to approve a new credit application in a bank (segregation of duty). Hence, two tasks implementing these checks must always be performed within the credit application business process.

In this paper we use our findings in the domain of control-flow-related compliance of business process design and extend it to be used in the domain of data-flow related compliance rules.

Compliance has become one of the main obstacles for enterprises to enter the area of cloud computing for their IT [5], [6]. One of the most important aspects that prevent the use of cloud technologies today is *data-sovereignty*. Data is not allowed to pass certain boarders like country boarders, or enterprise boarders because other regulations may apply when the data is in another country.

Companies have to trust cloud providers that their data is kept safe and that the data is not forwarded to third parties. When they use cloud offerings they loose control over their data but not the responsibility. Thus, many enterprises consider using a hybrid cloud approach. Hybrid clouds are a combination of for example a public cloud and a private cloud where operations on non-sensitive data are performed in the public cloud. These enterprises want to keep sensitive data in a private cloud environment within their IT premises. Whereas, public clouds are used, for example, for processing power intensive tasks working on data that is not crucial for

[1] http://www.iaas.uni-stuttgart.de/forschung/projects/cape/

the core business operations of the enterprise. One example for such a task would be the aggregation and preparation of reports depending on data that has been anonymised before.

Current regulations did not keep up with the fast changes in the IT land scape [7]. Many regulations state that the owner of data is responsible for its use instead of sharing the responsibility for the data between the customer and the cloud provider. This is in many cases not applicable for cloud computing. The EU data protection directive states that personal data can only leave the countries of the EU when the third country the data is sent to provides a certain level of protection [8]. Enterprises must be aware of the consequences when sending data to a cloud. Companies abandon data-sovereignty when they send their data to a public cloud. As a consequence, they loose control over their data but they are still responsible for it. In this paper we present data-flow related compliance solutions usable in cloud environments.

In [9] a classification of data-related compliance rules is shown. The classification has the following three main classes:

- **Content of data**: This class comprises compliance problems dealing with the information stored in data-objects.
- **Relation between data states and activities**: This class comprises compliance problems dealing with activity-executions being dependent on the state of a data-object.
- **Evolution of data objects**: This class comprises compliance problems dealing with the allowed states a data-object may have during the execution of a business process.

According to [9] solutions have been developed for parts of the last two classes of data-related compliance problems shown above. The data-related problem we tackle in this paper is a sub-problem of the first class. In [9] it is named *prohibited data*. The prohibited data problem deals with data that can only be accessed by a certain set of tasks within a process model. The following passages show an example when such a problem may occur in a real world business process scenario.

Many enterprises decide to outsource parts of their business to increase efficiency and decrease costs [10]. Let us assume a possible scenario where data about blood donations done in hospitals is stored in the data-centres of these hospitals. In this case it is reasonable to execute data aggregation algorithms near these hospitals in a private cloud [11]. Whereas the examination of the aggregated data can be executed in a public cloud, for example. This leads to a privacy problem. Data that has not been anonymised before cannot be sent to a public cloud because the hospital would loose control over the data.

The contribution of this paper is the introduction of *compliance domains*, along with a formal definition and use case scenarios. We further show a prototypical implementation

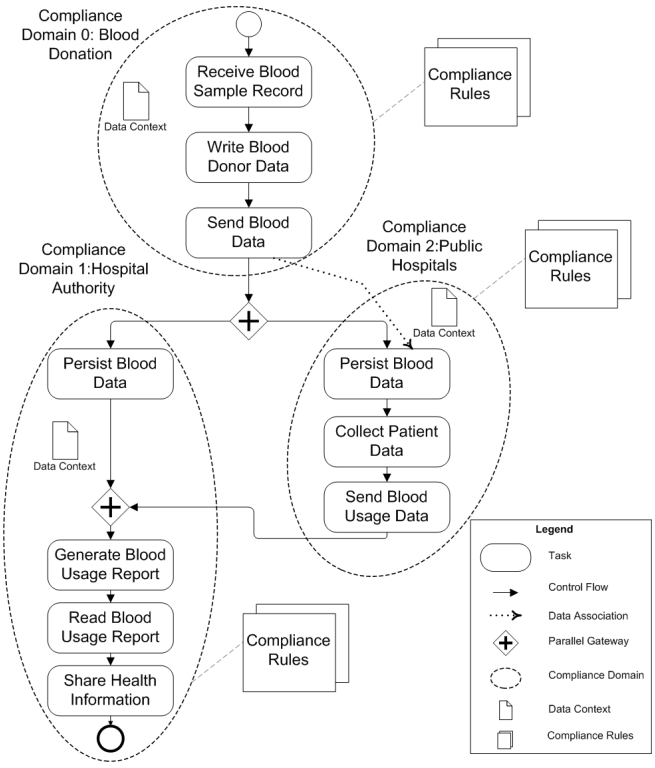


Figure 1. Business process annotated with compliance domains

of compliance domains in a graphical business process modelling workbench.

Compliance domains are a means to define data-related compliance requirements and to automatically check business processes for violations of data-related compliance rules.

The remainder of the paper is structured as follows. Section II presents a running example for our work. We use this running example to explain the new concepts presented throughout this paper. Section III presents a novel approach for data compliance in the cloud. This approach is applied in Section IV to physical cloud infrastructures. Our work on prototypes implementing the concepts is shown in Section V. Section VI discusses related work. Section VIII presents future work in the area of data-related compliance and cloud computing. Section VII concludes the paper.

II. Running Example

In this section we provide a real world scenario from the health care sector. We extended the scenario presented in [12] to be usable to explain the concepts we introduce in this paper. This scenario is used throughout the paper to explain our approach considering data-related compliance in business processes. In this scenario the Hong Kong Red Cross has to deal with sensitive data about their blood donors on the one side. On the oder side it has to send around data about the donated blood samples to public hospitals and

the hospital authority which is responsible for the data. A business process executing the transfer of data about blood donations modelled in the BPMN 2.0 [13] notation is shown in Figure 1. We chose BPMN 2.0 as the notation for the business process due to its well-defined semantic and its increasing usage in the health care domain [14]. Another reason for our decision is that BPMN is a widely adopted and understood standard. However, the business process could have also been modelled in any other language. Figure 1 shows the steps of the business process being executed when a new blood sample is collected. It is annotated with three compliance domains. In the following we show how they have been defined.

The first compliance domain is the compliance domain of the blood donation centres. Here, the data about the blood donors is handled non-anonymised. In the second compliance domain, the compliance domain of the public hospitals, the data about blood donors has restrictions imposed on it. The personal data like the name of a blood donor should not be sent with the blood samples. For the public hospitals it is only important to know the blood group and the gender of the donor of each blood sample. The third domain, the domain of the hospital authority, only needs to know the number of blood samples used by the public hospitals along with data about the number of blood samples collected by the blood donation centres. What we see in this example is that the compliance domains denote areas in the business process which are executed on different physical locations like the blood donation centres or the public hospitals.

The three compliance domains are differentiated in terms of the compliance rules that are imposed on the data they handle. Data that is exchanged between these compliance domains must be checked, so that the data handled by each compliance domain meets certain data-related compliance rules.

In the business process, the data about the sample and the blood donors is stored in an internal database. Then, the data is prepared and it is sent to the public hospitals and to the hospital authority. The data being sent to these two compliance domains must meet the compliance requirements of each compliance domain. Thus, prior to sending the data, it has to be prepared. The name of the donor among other fields has to be removed from the data that is sent to the public hospitals, for example. In the compliance domain of the public hospitals the data about the new blood samples is stored and blood usage data is sent to the hospital authority to generate a blood usage report. The hospital authority combines the data from the blood donor centres with the blood usage data from the public hospitals, generates the blood usage report and provides this information to all partners taking part in this business process.

In the following section we describe the concept of a compliance domain in detail.

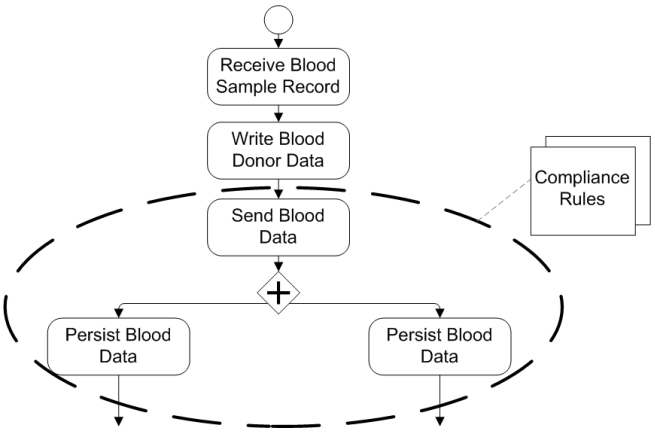


Figure 2. Example: Compliance scope

III. Data-Driven Compliance in Cloud Computing

Compliance domains are based on a concept presented in [4], called *compliance scope*. We start by presenting the concept of a compliance scope in Section III-A. Afterwards we introduce compliance domains in Section III-B and show in which cases they are useful. Following the introduction we present a formal definition of compliance domains and show a compliance verification algorithm.

A. Compliance Scope

In contrast to compliance domains, the focus of compliance scopes lies on the restriction of the *control-flow* and thus the execution order of the tasks within a compliance scope.

A compliance scope [4] is a means to annotate areas of a business process with compliance rules. The annotation is done by compliance experts. Every compliance scope carries at least one compliance rule imposing restrictions on the *control flow* of the tasks included within the compliance scope. The compliance rules are validated using a model checker. The result of the model checker is used by the design tool to notify the human business process designer, when a modification within a compliance scope has lead to a compliance violation.

The compliance rules annotated to compliance scopes are specified in linear temporal logic (LTL). LTL is the language used by a number of model checkers to define properties a system must comply with. If the contents of a compliance scope are modified, the process structures contained in that compliance scope are sent to a model checker. The model checker verifies if at least one compliance rule of the corresponding compliance scope is violated by the modification. In this case the model checker provides a counter example, showing which part of the business process model has lead to the violation. With this result the graphical business process design tool can notify the human business process designer and show which compliance rule has been violated. Figure 2 shows the first part of the BPMN 2.0

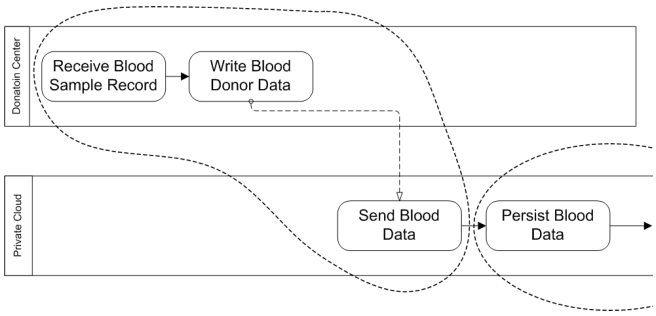


Figure 3. Example: Compliance domain and BPMN pool

business process from Section II. It is annotated with a compliance scope. The compliance rules coming with the compliance scope specify that the task *send blood data* must always be executed before either of the tasks labelled *persist blood data* can be executed. This is enforced by a model checker, not shown in the figure. For a formal description of compliance scopes we refer to [4].

B. Compliance Domains

Extending the definition of a compliance scope, compliance domains are a more generic concept. In the following we describe this concept in detail.

With compliance domains certain areas of a business process can be marked to be executed on specific runtimes. Compliance domains represent the different physical runtime infrastructures where parts of the eventual process are deployed on. An approach showing how to split a business process model and deploy it on different physical runtimes is shown in [15]. Such runtime infrastructures can be different cloud computing environments, like the Amazon EC2 [2] cloud computing service, or ordinary data-centres. The runtimes for every compliance domain are chosen at design-time. The decision is made using service level agreements (SLAs) linked with every compliance domain. We present an example of these service level agreements in the following bullet list:

- The costs for the execution of all tasks within the compliance domain must not exceed $300.
- Tasks within this compliance domain must be executed within the IT-premises of the company.

Compliance domains are a means to restrict the *data flow* in a business process using so called data-context objects.

Every compliance domain contains a data-context object, shown in Figure 1. The data-context object is based on a data-object of BPMN 2.0. In BPMN 2.0 data structures of data-objects can be defined using XML schema. The XML schema of the data-context describes the data that may enter and leave a compliance domain. Compliance rules attached to each compliance domain use the data-context to restrict the data-flow for each data-association entering or leaving the compliance domain. The data-context can be seen as a fixed format for the data being sent in and out of a compliance domain. If there is no field for sensitive data within the data-context of a compliance domain, no sensitive data can be sent to that compliance domain, since the incoming data must be stored somewhere within the data-context of the compliance domain.

A data-association (dashed line from task *send blood data* to task *persist blood data* in Figure 1) in BPMN specifies the data-flow from one task to another. We only specified one data-association in Figure 1 to keep the business process model easily readable. Other data may be generated by the tasks within a compliance domain. This data is not affected by the compliance rules as long as it does not cross the border of the compliance domain.

A possible use case where compliance domains can be used is the design of a business process to be executed on different runtimes, like a public cloud or a data-centre of an enterprise. In contrast to compliance scopes compliance domains can be used to mark certain parts of a business process to be executed on a certain runtime environment. Compliance scopes do not imply any runtime requirements as they are a pure design-time concept. The focus of compliance scopes lies on the restriction of the control-flow and thus the execution order of the tasks within a compliance scope.

Figure 4 shows how the data-context object is related to a compliance domain. Apart from this, it shows the relationships to the other parts of a business process model. A compliance domain is assigned to one or more process constructs in a business process model. Process constructs are tasks, connectors, or events in a BPMN process, for example. A compliance domain contains one or more compliance rules. Each of these compliance rules has a language indicator denoting the language which must be used to write a certain compliance rule.

Compliance domains can not be seen on the same level as pools in BPMN 2.0. In BPMN 2.0 pools are assigned to participants or more general roles like a buyer taking part in a collaboration. In contrast to pools, compliance domains are not restricted to partners. A compliance domain can cross the borders of pools, as shown in Figure 3. Here, the first part of the business process presented in Figure 1 is shown. In this example, a compliance domain is spanning two partners. This can be the case when the two partners share resources in the same cloud, for example.

In the following, we use the example of the Hong Kong Red Cross described in Section II to show the use of compliance domains: Let us take a look at Figure 1 again. This business process model represents a global business process that is meant to be separated and executed on different runtime infrastructures.

In our model an enterprise has a list of available runtime environments like data-centres of the enterprise or public clouds like Amazons EC2. Analogous to compliance domains,

[2]http://aws.amazon.com/ec2/

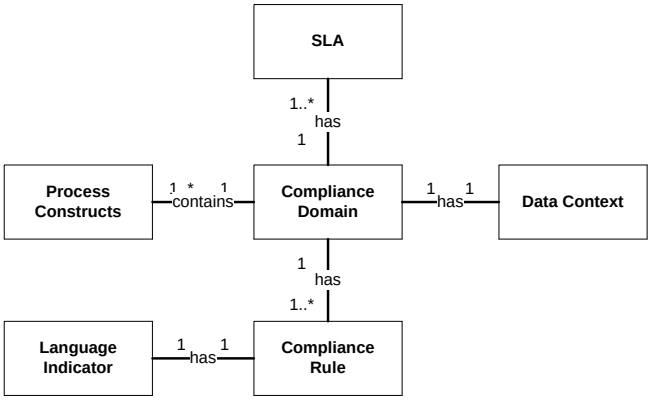


Figure 4. Compliance domain architecture model

each runtime environment is linked with a set of service level agreements. With the service level agreements linked to the compliance domains and the ones linked to the runtime environments, the actual runtime environment for each compliance domain can be negotiated.

If a public cloud environment like Amazon EC2 is chosen, for compliance domain two in Figure 1, certain data-related compliance requirements have to be met. One example of such a compliance requirement is that no sensitive data such as patient names must leave the IT premises of the hospital. This requirement is enforced by the graphical business process modelling tool. The tool notifies the human business process modeller when a data-flow is modelled that violates the compliance requirement presented above. After the development of the global business process model is finished and all tasks have been assigned to a compliance domain it can be separated and deployed on the specific runtimes chosen for each compliance domain.

In this paper we work with the last example of the list of service level agreements above because we experienced in the work with our business partners that the use of data being sent to the cloud is most important for enterprises willing to adopt cloud computing for their IT needs.

C. Formal Definition

A compliance domain can formally be seen as a hyperedge in a hypergraph [16]. The hypergraph represents the business process model being annotated with compliance domains. To get a coherent hypergraph, every task within the business process model has to be connected by a hyperedge. Thus, when a new task is inserted into a business process model the graphical business process modelling tool must ensure that the task is annotated with a compliance domain.

We begin to define a compliance domain by using the definition of a hypergraph.

$X = x_1, x_2, ..., x_n$ is a finite set of tasks in a business process model. Let E_i be a hyperedge in a hypergraph. Then a hypergraph H on X is a set of hyperedges $E_1, E_2, E_3, ..., E_m$ where

$$E_i \neq \varnothing \quad (i = 1, 2, ..., m) \tag{1}$$

and

$$\bigcup_{i=1}^{m} E_i = X. \tag{2}$$

Equation 1 means a hyperedge in a hypergraph at least connects one task. Equation 2 means that all tasks must be connected by at least one hyperedge.

Definition 3.1: A compliance domain is a tuple CD = (E, ζ, Δ, P), where

- E is a hyperedge in a hypergraph,
- ζ is a set of compliance rules,
- Δ is the data-context of the compliance domain, and
- P is a set of properties a runtime environment must fulfil in order to execute the part of the business process model contained in the compliance domain.

D. Validation of Data-Flow Between Compliance Domains

In this section we present the technical details of our approach to restrict the data-flow between compliance domains. Let us take a look at Figure 1 with the three compliance domains. The task labelled *send blood data* sends data containing information about the collected blood samples to the public hospitals and the hospital authority. The public hospitals then persist the blood data in the next step.

The data-flow in BPMN 2.0 is represented as dashed lines between tasks and data-objects. These dashed lines are called data-associations. Each data-association has two so called assignment attributes, not shown in Figure 1. The first attribute is called *from*. The second attribute is called *to*. The value of these attributes is an expression written in an expression language chosen by the human business process designer. The *from*-expression is used to evaluate the source of the assignment. It checks for example if the corresponding task has finished and has provided the data for the data-association. The *to*-expression is used to select and copy the output data of a task to a data-object or to another task. The BPMN 2.0 specification describes the use of XPath as an expression language to copy data within a business process model. For this reason and because a BPMN 2.0 business process model is entirely represented in XML, we use XPath as the language to perform the compliance checks.

In BPMN 2.0 every message and data-object has an attached XML schema. In the following we use this XML schema and XPath to evaluate the data flow between compliance domains.

The evaluation of a compliance domain is triggered when a new data-association is created with source and target being in two different compliance domains. In other words, it is created when a new data-association is crossing the borders of two compliance domains. In order to evaluate

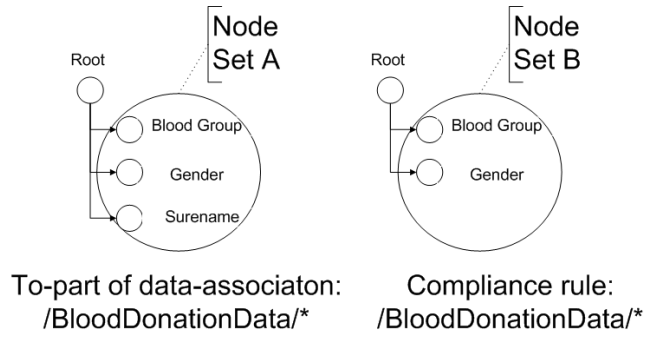


Figure 5. Example: Verification of XPath expressions

the data-flow entering a compliance domain at design-time, we work with *to*-expressions written in XPath. As described above, compliance domains are equipped with a data-context. We use this data-context and compliance rules written in XPath to validate new data-associations. With these XPath-expressions a certain set of XML-nodes is selected from the data-context of the corresponding compliance domain. This selected set of XML-nodes denotes the data that is allowed to be exchanged with the environment of the compliance domain. The compliance rule written in XPath selects a subset of this data-context. This subset of data is allowed to enter or leave the compliance domain.

Figure 5 shows on the left side the hierarchical XML structure of a schema of a message that is being sent to a compliance domain. On the right side Figure 5 shows the hierarchical structure of the schema of the data-context of the compliance domain of concern. The XPath-expression on the left side selects the set of XML-nodes (node set A) in the schema of the *to*-expression of a data-association. The XPath-expression on the right side selects the set of nodes (node set B) a compliance domain can handle. Node set B does not contain the node *Surname* because the data to be sent to the compliance domain of concern should not contain critical information like the name of a blood donor, for example.

In order to validate the nodes in the message being sent to the compliance domain, we subtract node set B from node set A, as shown in Equation 3.

$$NodeSetA \setminus NodeSetB = \{\} \tag{3}$$

The result is the node named *Surname*. The fact that the set of resulting nodes is not empty shows that there has been a violation of the the compliance rule. This result can now be presented to the human business process designer. With this information the human business process designer can change the message being sent to the compliance domain of concern.

Algorithm 1 shows the steps of the validation of data-related compliance rules in pseudo-code. The first three lines declare variables to be used in the algorithm. The variable *complianceRules* contains the compliance rules written in XPath that are attached to a compliance domain. The variable *dataCont* contains the XML schema of the data-context of the compliance domain. The variable *dataAssoc* contains the data-association to be validated.

In line one of the algorithm the *to*-property of the data-association is retrieved. The *to*-property contains the XPath expression to select the XML nodes to be copied to the compliance domain. With this *to*-property the XML node set is selected that is being copied to the compliance domain (see line two). In the for-loop beginning in line three all compliance rules are checked against the input node set. Line seven shows a popup if a compliance violation has been detected.

Algorithm 1 Validate compliance rule

```
Require: List complianceRules;
Require: Xsd dataCont; //data-context
Require: Object dataAssoc; // data-association
 1: XPath toProperty = dataAssoc.getToProperty();
 2: Set inputNodeSet = dataAssoc.getNodeSet(toProperty);
 3: for complRuleXpath in complianceRules do
 4:    Set allowed = dataCont.select(complRuleXpath);
 5:    Set result = subtract(inputNodeSet, allowed);
 6:    if result.size != 0 then
 7:       showComplianceViolation(result);
 8:    end if
 9: end for
```

E. Generation of Compliance Domains on Existing Business Process Models

In the previous sections we presented compliance domains as a concept to annotate business process models at design-time. This is useful when new business processes are created. Since companies also want to move their *existing* business processes to the cloud, or at least parts of them, we need a means to automatically annotate business processes with compliance domains. In this case compliance domains show which parts of a business process must be executed in a private cloud and which parts can be executed in a public cloud environment.

In the following we use the concept of a transitive closure on a graph [17]. We use it to calculate the compliance domains within the graph of a business process model. A directed graph $G^* = (V, E^*)$ is called the reflexive transitive closure of G if

$$(v, v') \in E^* \tag{4}$$

and if there is a path from v to v' in E. The transitive closure of a graph shows all vertices that are reachable from any vertex within that graph. We use this definition to calculate the data dependencies of tasks within a business

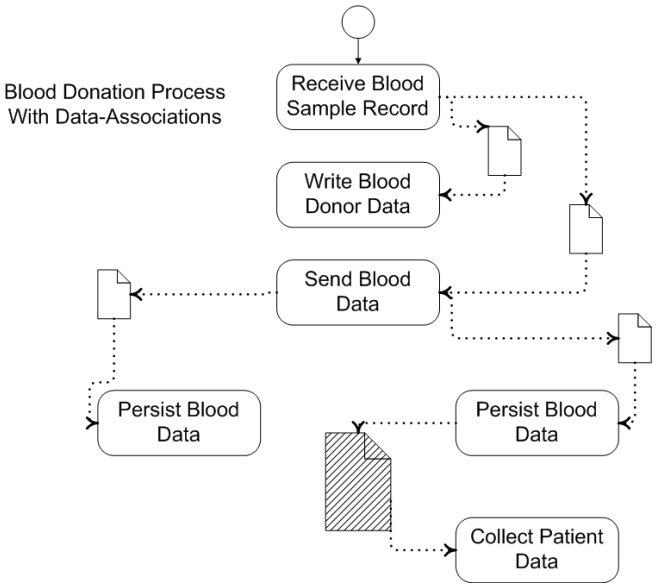


Figure 6. BPMN 2.0 business process model only showing data-flow. Hatched data-object contains non-sensitive data.

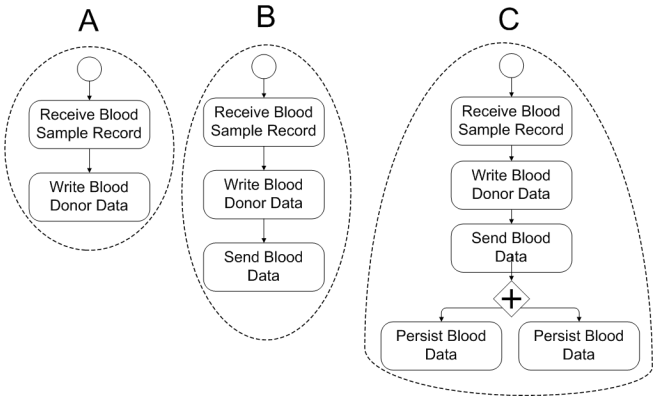


Figure 7. Growth of generated compliance domain from A to C.

process model. We construct a transitive closure of the input business process model based on the data flowing between the tasks.

Starting from a BPMN 2.0 business process model, we need to generate a graph $G = (V, E)$ that is the input for Algorithm 2. In G, V represents the set of tasks in the original BPMN 2.0 business process model. The edges of the graph E are the data-associations of the original BPMN 2.0 business process model. We do not include the control-flow connectors from the original BPMN 2.0 business process. Thus, the new graph only represents the data-flow in the business process model. This graph is shown in Figure 6. The hatched data-object contains non-sensitive data. Thus, the task *collect patient data* should not be present in the resulting compliance domain.

The goal of this approach is to show sensitive data flowing through the business process model. First, we are interested in this sensitive data that is dealt with in the business process model. In order to distinguish the sensitive data from the non-sensitive one, we assume a human compliance expert provided a data-context object containing information about sensitive data-fields. Any data-association *not* using any of these data fields is deleted from the graph. Thus, we have all data-associations which handle sensitive data.

We use the algorithm of Floyd and Warshall which is used in its original form to calculate the transitive closure of a graph. The algorithm (see Algorithm 2) requires an adjacency matrix containing the graph to be examined and a list as input. After the algorithm is run, the list contains all nodes in the input graph which are in one compliance domain. The for loop, starting in line one, is preparing the adjacency matrix. It replaces the empty diagonal of the adjacency matrix with ones denoting that each vertex $\in V$ in the graph is connected with itself. The following two for loops are used to iterate through the graph. In line six the algorithm checks if there is an edge between the vertices i and j. If this is the case it loops through the graph looking if there is an edge from j to k. If this is the case the original algorithm would add an edge from vertex i to vertex k.

We adapted this algorithm in line 9 to be able to show which vertices in G are handling sensitive data. Instead of adding a new edge to the current node which has been reached, we add the target vertex of the edge to the list of vertices handling sensitive data.

Figure 7 shows the growth of the compliance domain, from A to C. The task *collect patient data* is not included in the resulting compliance domain C because it works on non-sensitive data.

Algorithm 2 Generate Compliance Domains

```
Require: Array[][] adjMatrix;
Require: List complianceDomain;
 1: for i = 1 to numberOfNodes do
 2:     adjMatrix[i][i] = 1;
 3: end for
 4: for j = 1 to numberOfNodes do
 5:     for i = 1 to numberOfNodes do
 6:         if adjMatrix[i][j] == 1 then
 7:             for k = 1 to numberOfNodes do
 8:                 if adjMatrix[j][k] == 1 then
 9:                     complianceDomain.add(k);
10:                 end if
11:             end for
12:         end if
13:     end for
14: end for
```

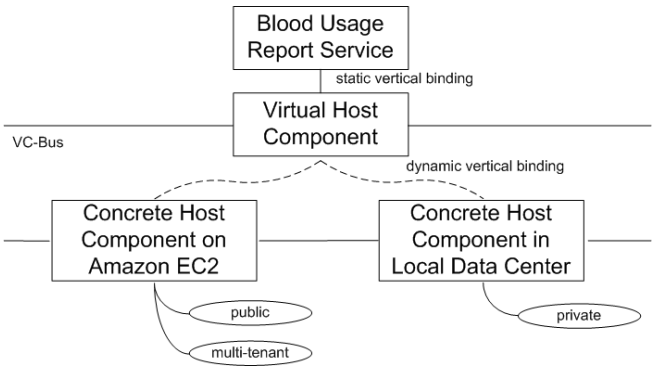


Figure 8. Binding of the Blood Report Generation Service to different Runtime Infrastructures

IV. Deployment of Compliance Domains on Runtime Infrastructures

Apart from design time checks it is advisable to include compliance checks for data-related compliance rules also at runtime. Therefore we propose the following solution.

To deploy processes and the orchestrated services on different runtimes, we use the the concept of *vertical composition* introduced in [18]. While *horizontal binding* refers to the composition of services that is specified by BPMN 2.0 processes in our scope, vertical binding enables the composition of services with different hosting infrastructures.

Instead of just selecting an available service for a step in a BPMN 2.0 process, a user may specify a custom vertical composition, too. This enables a user specific deployment of service components to different environments.

A. Deployment of Services Used in Compliance Domains

The *generate blood usage report* task in our running example invokes a service that handles this report generation. The implementation of this service depends on a runtime that is provided by the infrastructure on which it is deployed. This infrastructure is provided by a so called *host component*. Publishing, discovery, and binding of host components is handled by the *Virtual Component Bus*, also introduced in [18]. In our scenario, the services generating the blood usage report can bind against a virtual host component that can be provided by a concrete host component, present in a specific runtime environment, as depicted in Figure 8.

To be able to negotiate the binding of services to hosting environments, the hosting environments are described by a set of service level agreements. For example, a cloud like Amazon EC2 can be described as *public* and *multi-tenant*, because it is accessible to arbitrary users which share resources in the cloud. On the other hand, a local data centre can be described as being private. These service level agreements are then used to specify compliance rules associated with a compliance domain. A compliance rule may specify that services used by a certain set of tasks of a business process must reside in a private runtime environment. By means of such rules, the set of usable concrete host components to which a virtual host component can be bound is therefore constrained. This ensures a compliant deployment of services among available runtime environments.

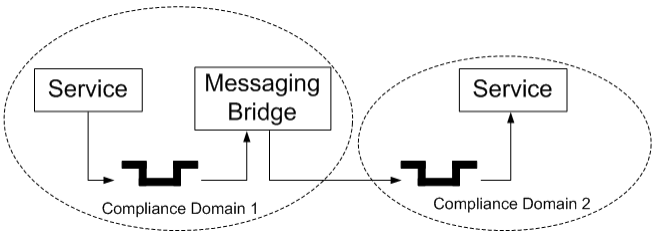


Figure 9. Integration Components used for Runtime Evaluation of Compliance Rules

B. Runtime Evaluation of Data-Flow Between Compliance Domains

Messages exchanged between processes are considered to use messaging queues for reliable communication. Similar to the services accessed by a process, a message queue is offered as a virtual service by the vertical component bus. This message queue can be bound to different hosting environments or other services providing its functionality. As an optional feature, the data flow between compliance domains can be monitored at runtime by these message queues. To do so, multiple integration components are deployed by the VC-Bus to enable inter compliance domain communication. One of these components may for example be a message queue.

As depicted in Figure 9, message queue services are used to implement compliance rules for each compliance domain to be integrated. Then, a message bridge [19] is setup to move messages from one of these message queues to the other. The direction is determined by the communication link in the corresponding BPMN 2.0 business process model. The message bridge is hosted in the same compliance domain where the messages originate and is configured with the compliance rules of that compliance domain. Prior to moving messages it evaluates these rules to ensure that the information contained in messages is allowed to leave the compliance domain. The data-related compliance rules from the business process level, can by these means be checked on yet another layer.

V. Prototype

This section shows considerations and design decisions made for the development of the design-time compliance concepts presented in this paper. The prototype is based on the web-based BPMN editor Oryx[3]. For every language the

[3]http://code.google.com/p/oryx-editor/

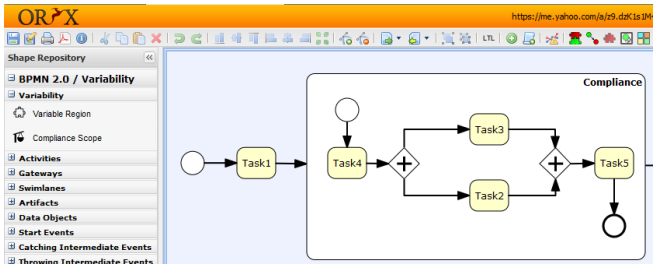


Figure 10. Graphical representation of a compliance domain

graphical elements of the language are described in so called stencil sets. Oryx provides an extension mechanism to add, remove, and modify stencils in a stencil set. We extended the BPMN 2.0 stencil set, that is bundled with Oryx, and added a new stencil representing a compliance domain. Every stencil within a stencil set has got a set of properties attached to it. We added a new property to the compliance domain stencil to be able to store corresponding compliance rules with every compliance domain. Figure 10 shows the editor window in Oryx. A running instance of the prototype along with example processes can be accessed on the prototype web site[4].

It further shows a compliance domain (white rectangle) containing several tasks. Such compliance domains can be inserted into a process model in the same way a task is inserted, for example. We also developed a graphical compliance rule wizard guiding people in defining compliance rules.

Besides the possibility to extend stencil sets, Oryx has further extension mechanisms. Almost every component in Oryx has been developed with plugins. We added a plugin which is responsible for the validation of data-related compliance rules. Events can be used in Oryx to notify the user if a modification has been done to the current business process model. The plugin we added for the validation of data-related compliance rules listens to such events. If an event occurs the plugin first of all checks if a new data-association has been inserted into the current business process model. In this case, the plugin checks whether the new data-association crosses the borders of two different compliance domains. This can be done by checking if the source and the target coordinates of the new data-association are within different compliance domains. If the new data-association crosses the border of a compliance domain, the resulting node set of the XPath-expression in the compliance rule of the corresponding compliance domain is subtracted from the resulting node set of the XPath expression at the *to*-part of the new data-association. If this calculation does not result in the empty set, the human business process designer is notified by a popup showing the data-field that has violated the compliance rule.

[4]http://www.danielschleicher.com/?page_id=7

VI. Related Work

In [20] a general approach is presented to deal with data-related compliance rules in business process design. The authors describe the problem of a state explosion when data-related compliance rules are checked with model checkers. This problem comes from the fact of the huge number of states, variables can reflect. The problem of state explosion is met by abstracting from this huge number of states merging a number of states a variable can posses to one abstract state.

A different approach for data-related compliance is presented in [21]. Here, BPMN-Q, a query language for business process models, is used to check compliance rules in business processes. BPMN-Q is extended to account for data-aspects in business process models. This approach heads in the same direction as the approach presented in [20]. The data-related compliance rules presented, are in direct association with the control-flow of the business process to be checked. This means, if a variable is in a special state the control-flow of the corresponding business process is directed in a certain direction. If the variable is in another state the control-flow of the process would have gone in the other direction.

In contrast to the approaches above, the approach presented in this paper deals with a different kind of data-related compliance rule. The approach presented in this paper deals with the higher level compliance rule of data-sovereignty.

In [9] Cabanillas et al. show which data-related compliance problems have been identified in the literature. They also show which of these problems have been addressed. However, there is no solution approach presented for a number of data-related compliance problems.

In this paper we present an approach for one of the identified data-related compliance problems that have not been addressed in previous work, yet.

Another work identifies compliance problems arising in the domain of cloud computing and business processes such as the handling of sensitive data of a company by cloud providers. The authors propose the extension of existing tools that have been developed for the Fraunhofer project *Architectures for Auditable Business Process Execution* (APEX). However, no specifics are presented on how the APEX tools will be extended and which concepts will be used.

VII. Conclusion

We presented an approach to restrict the data-flow between certain areas of a business process model at design-time. This approach is based on the novel concept of compliance domains. These compliance domains are used to annotate areas of a business process model with service level agreements. They are dividing business process models. Each part of a business process may be deployed on a different runtime. Certain service level agreements have to be met by the runtime environment where each part of the process is later executed on. If data is moved between these runtime

environments certain restrictions may apply. We showed how XPath expressions can be used to define compliance rules and how these compliance rules are evaluated. Following the evaluation of compliance rules, an approach for the mapping of compliance domains to runtime infrastructures was presented. Further, we showed a prototype based on the web based BPMN editor Oryx implementing the design-time concepts presented in this paper.

VIII. Outlook

In future work we will further extend the concept of compliance domains with a mapping of the service level agreements, coming with the compliance domains, to requirements of the IT-infrastructure. Here, we will focus on cloud environments and the provisioning of middle-ware components meeting the requirements coming from compliance domains. The provisioning infrastructure must be capable of interpreting the service level agreements of a compliance domain and provision certain middle-ware components accordingly.

Defining XPath expressions can be cumbersome. We will work on a graphical solution to define Xpath expressions. To implement this functionality we display a graphical tree view of the XML schema of a data-object. The XPath representing the compliance rule is then generated when the user clicks on a node of that XML tree.

Acknowledgment

The author D. Schumm would like to thank the German Research Foundation (DFG) for financial support within the Cluster of Excellence in Simulation Technology (EXC 310/1).

References

[1] J. Vanhatalo, H. Völzer, and F. Leymann, "Faster and more focused control-flow analysis for business process models through sese decomposition," in *Proceedings of the 5th international conference on Service-Oriented Computing*, ser. ICSOC '07, 2007.

[2] D. Schleicher, T. Anstett, F. Leymann, and R. Mietzner, "Maintaining Compliance in Customizable Process Models," in *Proceedings of the 17th International Conference on Cooperative Information Systems*, November 2009.

[3] D. Schleicher, T. Anstett, F. Leymann, and D. Schumm, "Compliant Business Process Design Using Refinement Layers," in *Proceedings of the 18th International Conference on Cooperative Information Systems*, Oktober 2010.

[4] D. Schleicher, M. Weidmann, F. Leymann, and D. Schumm, "Compliance Scopes: Extending the BPMN 2.0 Meta Model to Specify Compliance Requirements," in *Service-Oriented Computing and Applications (SOCA), 2010 IEEE International Conference on*, Dezember 2010.

[5] "Survey by IEEE and Cloud Security Allicance Details Importance and Urgency of Cloud Computing Security Standards," http://www.cloudsecurityalliance.org/pr20100301c.html, 2010.

[6] D. Mortman, "Cloud Computing Compliance: Exploring Data Security in the Cloud," http://bit.ly/cYt4Bv.

[7] T. Mather, S. Kumaraswamy, and S. Latif, *Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance*. O'Reilly Media, Inc., 2009.

[8] "Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data," http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31995L0046:en:HTML, 1995.

[9] C. Cabanillas, M. Resinas, and A. Ruiz-Corts, "On the identification of data-related compliance problems in business processes," 2010.

[10] T. L. Friedman, *The world is flat : a brief history of the twenty-first century*. Farrar, Straus and Giroux, 2005.

[11] J. Foley, "Private Clouds Take Shape," http://www.informationweek.com/news/services/business/showArticle.jhtml?articleID=209904474, 2008.

[12] T. Trojer, C. kwong Lee, B. C. M. Fung, L. Narupiyakul, and P. C. K. Hung, "Privacy-aware health information sharing," in *Privacy-Aware Knowledge Discovery: Novel Applications and New Techniques, Chapman and Hall/CRC Press*, 2010.

[13] Object Management Group, *Business Process Model and Notation (BPMN) Version 2.0*, OMG, 2010. [Online]. Available: http://www.omg.org/cgi-bin/doc?dtc/09-08-14

[14] T. Benson, "Principles of health interoperability hl7 and snomed," *Health San Francisco*, vol. 49, no. 3, pp. 25–34, 2010. [Online]. Available: http://www.springerlink.com/index/10.1007/978-1-84882-803-2

[15] R. Khalaf, "Supporting business process fragmentation while maintaining operational semantics: A bpel perspective," Dissertation, Juni 2008.

[16] C. Berge, Ed., *Hypergraphs Combinatorics of Finite Sets*. Elsevier, 1989, vol. 45. [Online]. Available: http://www.sciencedirect.com/science/article/B8GWS-4S8VKBN-1/2/04c27c2e3da24aef41d122f22008a366

[17] S. S. Skiena, *The algorithm design manual*. New York, NY, USA: Springer-Verlag New York, Inc., 1998.

[18] R. Mietzner, C. Fehling, D. Karastoyanova, and F. Leymann, "Combining horizontal and vertical composition of services," in *Service-Oriented Computing and Applications (SOCA)*, 2010.

[19] G. Hohpe and B. Woolf, *Enterprise integration patterns*. Addison-Wesley, 2004.

[20] D. Knuplesch, L. T. Ly, S. Rinderle-Ma, H. Pfeifer, and P. Dadam, "On enabling data-aware compliance checking of business process models," in *Proceedings of the 29th international conference on Conceptual modeling*, 2010.

[21] A. Awad, M. Weidlich, and M. Weske, "Specification, verification and explanation of violation for data aware compliance rules," in *Proceedings of the 7th International Joint Conference on Service-Oriented Computing*, 2009.