Gabriela Loosli

Dynamic Binding in a SOA and its Potential Implications on Compliance Verifications

An essential characteristic of a Service Oriented Architecture (SOA) is binding at runtime. Different forms of this dynamic binding of services exist. When applying the most far reaching form, there is the risk that because of changes in the inventory of services inside the repository, non compliant services will be automatically selected at runtime. In this regard, non compliant means that not all required regulations are recognised and may therefore be violated. If services are reused in a different context, there are possibly additional regulations to be complied with. Current change management approaches which are based on testing all services and applications before going into production do not solve this problem. As a result, compliance can not be guaranteed in all cases. In this paper, we introduce an approach that seeks to avoid the selection of non compliant services by means of semantic concepts.

1 Introduction

An important goal of a Service Oriented Architecture (SOA) is to support the agility of a company. A SOA is a software architecture in which services are the fundamental elements. Services are software units at different levels of granularity. They can be aggregated to create more complex services, up to the level of processes or applications (e.g., Papazoglou and Georgakopoulos, 2003; Erl, 2004, 33ff.; Krafzig et al., 2004, 58ff.; Dostal et al., 2005, 7). Agility means the flexibility to implement changes inside business processes in IT systems in a timely manner (Erl, 2004, 297; Krafzig et al., 2004, 1ff.; Aberdeen Group, 2007, 1; Johannsen and Goeken, 2007, 189; Winter and Schelp, 2007, 44ff.). Flexibility is achieved by loose coupling between services. By establishing a relationship between them only in the process sequence, they can be replaced more easily inside a process or bound into it only on demand. In the SOA concept this is even possible at runtime because the services are not linked hard coded in the source code but bound via their information in the repository. Runtime binding is also called dynamic binding and contributes essentially to loose coupling (Erl, 2004, 297; Krafzig et al., 2004, 46ff.; Dostal et al., 2005, 9; Johannsen and Goeken, 2007, 190ff.). Besides this advantage, however, a factor to consider is that by applying a dynamic selection of services one has to pay special attention to legal aspects. A random binding of a service, for example, means an out of control change of the overall system. Hence the system would be in an undefined state and therefore no longer ready for operation (Dostal et al., 2005, 261).

Although with static binding there still are unsolved problems too in relation to compliance (because, e.g., processes consist of services, which originate from different organisation(al unit)s; e.g., Papazoglou et al., 2006, 24; Johannsen and Goeken, 2007, 191; Kohnke et al., 2008, 408), this paper focuses on dynamic binding. It points out the implications the different forms of dynamic binding of services have on compliance with regulations and presents an approach which allows the context of service use to be taken into account and, as a result, the required regulations to be determined by means of semantic concepts. In this way, the selection of non compliant services is to be avoided.

The remainder of the paper is structured as follows: In Section 2 related works are reviewed and

differentiations to them are shown. Section 3 defines compliance. In Section 4, the problems with evidence of compliance in a SOA when applying different forms of dynamic binding are illustrated and solution approaches are presented. Section 5 gives a summary and an outlook.

2 Related Work

Although compliance is a highly topical issue, little attention is given to it in the SOA literature. (Dostal et al., 2005, 252ff.) devote a separate subsection to legal frameworks and also indicate the difficulty with dynamic binding in this regard. However, they state that they do not cover the topic exhaustively. They mention that all available services have to satisfy legal requirements and a replacement of services has to be taken into account and documented in advance (Dostal et al., 2005, 261); checking of processes and reuse in another context are not considered, nor are the different forms of dynamic binding. SOA literature generally pays little attention to these.

Recent publications mention the term *SOA Governance* (e.g., Aberdeen Group, 2007; Johannsen and Goeken, 2007; Keller, 2007; Schelp and Stutz, 2007; Siedersleben, 2007; Josuttis, 2008; Kohnke et al., 2008; NorthPage Research, 2009). However, this term is often used in a wider sense than simply with regard to compliance checking and also covers aspects of traditional IT management (for a delimitation of these two terms cf. Knolmayer and Loosli, 2006, 451ff.). Thus, tasks such as *aligning SOA with business objectives* (Keller, 2007, 292; Schelp and Stutz, 2007, 69; Kohnke et al., 2008, 409ff.), *optimising the IT* (Keller, 2007, 300), *successful SOA implementation* (Kalex, 2007, 330; Keller, 2007, 304; Schelp and Stutz, 2007, 69; Siedersleben, 2007, S111; Josuttis, 2008, 325) or *making design decisions* (Aberdeen Group, 2007, 3; Fabini, 2007, 314; Kalex, 2007, 330; Josuttis, 2008, 326; Kohnke et al., 2008, 410) are also considered. In this paper only compliance related tasks such as *providing IT support for determining correct figures in accounting as well as for automated controls, defining procedures for system changes during operation, avoiding uncontrolled growth of services or establishing access policies* (Knolmayer and Loosli, 2006, 452; Fabini, 2007, 313; Kalex, 2007, 330; Kohnke et al., 2008, 411) are discussed.

Furthermore, dynamic binding is not taken into consideration in these contributions. (Johannsen and Goeken, 2007, 191) mention reuse in different contexts, possible problems connected with it are, however, not considered. In contrast, Foody (2006) takes the additional regulations to be complied with into account and therefore the implications of reuse of *existing services* in *new processes*. What is missing is the reverse case, an analysis of the implications that *new services* available in the repository have on *existing processes* (see Section 4.1).

For providing evidence of compliance, there are interesting *solution approaches* which test processes, sometimes even at runtime. An overview of current work and open research questions about compliance checking in business processes is given in El Kharbili et al. (2008). However, all approaches are somehow incomplete: Some are not specifically aligned to a SOA (Giblin et al., 2005; Agrawal et al., 2006; Namiri and Stojanovic, 2007a,b; Karagiannis, 2008). Others do not test process changes at runtime (Liu et al., 2007; Namiri and Stojanovic, 2007a,b). Agrawal et al. (2006) focus on only one regulation. O'Grady (2004) suggests generic compliance services underlying all regulations (e.g., access control). It is important to remember here, however, that it is precisely the differences in detail between the various regulations that are significant for compliance. In Hepp et al. (2005), compliance assistance for instance is provided with semantic descriptions and based on these queries like *List all business processes that depend on system x.* However, queries alone are not enough and one has to be aware of every (potential) context change, so that it can be queried explicitly (see Section 4.3.1). An overview of current work in the field of semantic concepts and dynamic binding is given by Kuropka et al. (2008). The contribu-

tions refer, for instance, to faulty and incompatible services, but not specifically to compliance with regulations.

3 Compliance

The need for companies to comply with regulations is not new. What is new, however, is the proliferation, the far reaching requirements and the internationality of the rules (e.g., Dostal et al., 2005, 252ff.; Knolmayer, 2007, S98ff.). The most frequently quoted example in this context is the Sarbanes-Oxley Act (SOX). An overview of the implications of that regulation on IT systems is given by Knolmayer and Wermelinger (2006). Besides the rules for financial accounting, there are further legal requirements such as Data Protection Acts, industry specific regulations (like Basel II or REACH), or internal rules which have to be considered. For all of these different regulations, which often contain imprecisely formulated provisions and change frequently, companies have to provide evidence of compliance. Compliance can be defined as adherence to all legal, governmental or regulatory and internal rules relevant to the particular company as well as the consideration of usual market standards and rules of professional conduct for establishing greater transparency and controllability of the behaviour of a company and its employees (cf. Thelesklaf, 2001, 447; Eidg. Bankenkommission (EBK), 2006, 10; Johannsen and Goeken, 2007, 15).

IT is affected by it (apart from the rules that it has to meet directly) on the one hand through implementing of business processes in its systems, on the other hand through automated support for audit using controls and reports (i.a. Knolmayer and Wermelinger, 2006; Knolmayer, 2007). Compliance is often associated with Corporate Governance. From Corporate Governance, IT Governance is derived for the IT domain (Knolmayer and Loosli, 2006, 451; Knolmayer, 2007, S98; Schelp and Stutz, 2007, 69) and, in turn, SOA Governance for Service Oriented Architectures (Keller, 2007, 289; Schelp and Stutz, 2007, 69).

4 Compliance Check in a SOA

4.1 Possible problems

In a SOA, services are registered and discovered in repositories. Therefore it is important that the services inside the repository are compliant with the relevant regulations. A service can be used by several processes in different contexts, by *reusing* it. Services of internal as well as external providers can be deployed. First of all there is the question, what is meant by a compliant service in a SOA. This is clarified by the following example of a risk potential for the SOX:

> The company has a standard sales contract, but sales personnel frequently modify the terms of the contract. Sales personnel frequently grant unauthorized and unrecorded sales discounts to customers without the knowledge of the accounting department. These amounts are deducted by customers in paying their invoices and are recorded as outstanding balances on the accounts receivable ageing. Although these amounts are individually insignificant, they are material in the aggregate and have occurred consistently over the past few years. (PCAOB, 2009, 268)

Referring to a SOA, this means, for example, that there is a separate *discount service* which can only be invoked by authorised employees. The service undertakes a correct booking of the discount in the system and thereby reduces the invoice amount. The subsequent *invoicing service* issues the invoice for the (remaining) amount that is available in the system. Compliance of the *discount service* is ensured by checking access authorisations, conformance with the standard sales contract as well as correct booking in the system. In case of the *invoicing service* this is done by ensuring that the invoice amount is taken out of the system and can not be manually entered or changed. These checks can be done at design time so that the services are registered in the repository only after successful testing. This means that it can be addressed by an

organisational measure: An approval process for going live of services.

The acquisition of *external* services constitutes a special case. If, for example, there is a change of the service provider at runtime, the new one has to comply with the regulatory requirements as well and the service consumer has to ensure that this is the case. Thus, for compliance related tasks, the consumer has to pay attention to how the output is generated (Dostal et al., 2005, 261; Knolmayer, 2007, S101). This contradicts an important characteristic of the SOA: The separation of interface (*what*, relevant for the service consumer) from implementation (*how*, done by the service provider) (cf. e.g., Erl, 2004, 37). This problem too can be solved with an organisational measure: By restricting the available services. For instance, only services from the corporate repository are permitted to be used. External services are tested in the same way as internal ones, before they are incorporated into this repository.

Even if all services in the repository are compliant, this is not automatically the case for all processes which use these services. Namely, if services are *reused* in another process in a *context* not previously considered. Again, this situation is illustrated with an example:

A service, which saves log data, is checked for compliance before going live. Later the service is reused in an ordering process in which it also stores credit card data in addition to the general ordering information. Now, the same service is for example additionally subject to the rules of the Payment Card Industry Data Security Standard (PCI DSS), without being changed itself and being brought into production again. The only thing that has changed is the kind of use, thus the context (Foody, 2006). The PCI DSS rules prohibit, for example, the storage and retention of data like complete magnetic stripe information or PIN PA-DSS (2008).

The left side of Figure 1 shows that it is not enough to check the offer (the services and applications, which have implemented the services) on its own, but that in addition, the context (processes/applications) in which the services are used, has to be taken into account. A logical conclusion would be, as for the services, to check the (newly created or changed) processes, before they enter the productive system. Thus, again an organisational measure, an approval process for going live of processes. However, this alone is not enough as the reverse (illustrated in the right side of Figure 1) shows: An existing process 1, which needs credit card data, has two existing log services for selection, a cheap, slow one, and an expensive, fast one. Both comply with the required PCI DSS regulation. The process searches for the cheapest service (for evaluating financial performance of alternative business process configurations cf. vom Brocke, 2008). The benefit of this dynamic selection is that new or improved services can be used without any change in the code of the invoking process. If, for instance, log service 2 is improved and is newly provided as the cheapest one, then the next time process 1 is executed, this service will be automatically considered. In the initial situation, log service 1 is selected. Later on, another process 2 is created which brings in its own log service 3. Because this one has different non functional properties (price, execution time: It is even cheaper and still fast, because it omits sorting out of critical data), it is approved in addition. It does not comply with the PCI DSS regulation, but this does not present a problem for process 2, because it does not need credit card data. However, at the next invocation of process 1, this process automatically accesses the now cheapest, but for it non compliant log service 3: Because of a change in the inventory of services inside the repository, a non compliant service is automatically selected at runtime.

4.2 Implications of dynamic binding on compliance

As explained below, there are different forms of dynamic binding. Depending on the form applied, the problem with reuse in a different context ei-

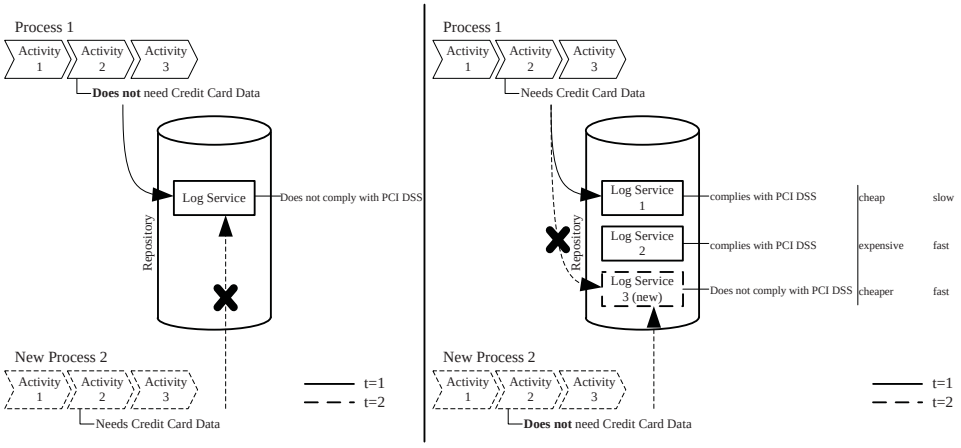

Figure 1: Implications of existing services on new processes and vice versa

ther does not arise at all or can be solved with organisational measures. Before examining these forms, the term dynamic binding is defined.

Binding of services to processes or higher level services in a SOA involves abstraction from the implementation details (as for example the physical address). This is done by not writing them hard coded in the source code of the implemented process, but in a separate document, the service description, in the repository. The source code only specifies which service description should be accessed (Pautasso and Alonso, 2005, 152ff.; Tilkov and Starke, 2007, 18). It is this abstraction which makes binding at runtime (*dynamic binding*) possible at all.

As in the example above (abstraction from the physical address), for (Tilkov and Starke, 2007, 18ff.) there is dynamic binding if a service consumer does not determine the address of the service provider until runtime. (Dostal et al., 2005, 9) go a stage further and understand by this term that at the time of code generation it is often not known which services at all will be invoked at runtime, for instance because of external influences or user preferences.

Pautasso and Alonso (2005, 158ff.) provide a more detailed breakdown of static (binding at design time) and dynamic binding: From binding at 'registration time' (the way a service and its description is catalogued in the repository affects its discovery) up to binding at 'invocation time' (the decision as to which service is used is made at the latest possible time, i.e., at the invocation of the service). They note that in most cases binding at 'invocation time' is referred to as dynamic binding. In this paper, we also define dynamic binding as binding at the latest possible time.

In addition to the *time* of binding of services (here: dynamic binding), a further distinction can be made based on how much information the source code contains in order to identify the appropriate service and its description in the repository. This means that different *forms* of dynamic binding can be distinguished.

4.2.1 Binding by name

In the simple case, exactly *one* service is uniquely identified by its *name* in the repository. This is referred to as 'binding by reference' (Pautasso and Alonso, 2005, 155) or 'runtime service lookup by name' (Krafzig et al., 2004, 63). According to Krafzig et al. (2004, 63), the service definition (how to address the service and what the output is it delivers) is known to the service consumer at design time and is considered accordingly. In that case, the dynamic element consists solely of the fact that

the physical site (the address) of the service is not determined in advance. Thus, it can change without the source code of the implemented process having to be changed. Also, there is the opportunity to provide the same service on different machines (in the service description several physical addresses, so called 'endpoints' can be indicated) and, depending on the workload, to choose the one or the other site (cf. e.g., Siedersleben, 2007, S113). Furthermore, the provider can be replaced by adapting the offer in the repository accordingly. If the service and its description from the new provider is stored under the same name (and the *old* service removed), then the process will automatically access the new service at the next invocation.

Evidence of compliance: If there is an organisational measure in place which checks the compliance of services before they are registered in the repository, this form of dynamic binding is rather unproblematic: The service is uniquely determined. The dynamic selection is solely related to different sites. The problem with reuse in a different context can not arise because another service can not be selected at runtime.

In the example with the log services, log service 1 is bound by its name. Regardless of which services are added to the repository, this one will always be accessed (right side of Figure 1). As it can be assumed that not only the services but also the processes themselves are checked for compliance with the required regulations before going into production, in the reverse case too (left side of Figure 1), the problem can be solved with this organisational measure.

4.2.2 Binding by constraints or properties

In the complex case, the services are searched and discovered in the repository by *constraints* or *properties*. Consequently, this kind is referred to as 'binding by constraint' (Pautasso and Alonso, 2005, 156) or 'runtime service discovery based on reflection' (Krafzig et al., 2004, 63ff.). Krafzig et al. (2004) add another intermediate form which they name 'runtime service lookup by properties'. While searching is indeed already done by properties here, it is only within a preselected set of services specified in the source code, not within the entire current inventory in the repository. As a preselection has already been made, in this case, unlike the open search, it is not absolutely necessary to consider semantics when searching (Krafzig et al., 2004, 63ff.; Pautasso and Alonso, 2005, 156). An example of this type is given by Spahn (2007) and vom Brocke (2008, 104ff.): In order to carry out an activity in a business process, several services are available with identical functionality but different non functional properties, like execution time or costs (preselection). As in the example of the log services, there is the possibility to choose between a slower and cheaper or a faster and more expensive service. Based on the preferences and restrictions, the compatible service composition is selected automatically. The definition, which services are functionally identical, is stored manually (Spahn et al., 2006, 5; Spahn, 2007, 316ff.; vom Brocke, 2008, 110ff.). If several possible applicable services are returned as the result of the search, these have to be evaluated and a service has to be chosen. For example, this selection can refer to the quality of services, the reputation of the provider or the costs (Berbner et al., 2005, 269ff.; Pautasso and Alonso, 2005, 156ff.; vom Brocke, 2008, 104ff.).

Evidence of compliance: In the case of the intermediate form, the situation is similar to binding by name. Instead of a single service, a set of services is given here. The selection should be justifiable (specifying the criteria or rules) by using documentation. In the case of the most far reaching form, the open search, however, the problem of reuse in a different context can not be solved solely by checking the process before going into production, as the example with the log services shows: Since on the basis of the property 'costs', at every invocation a search is again done for the cheapest service within the entire current inventory of the repository, there is a risk that due to a

change in the inventory, a non compliant service will be selected at the next invocation, such as in the example the log service 3 for the process 1 (cf. Figure 1, right side).

This form of dynamic binding is, however, rarely used in practice (Krafzig et al., 2004, 64), one reason being that it requires semantic concepts and not many implementations of these exist yet (e.g., Bell et al., 2007, 70; Haniewicz et al., 2008). However, it is desirable that, in the future, processes can be created, changed or deleted on the basis of goals, in as automated a manner as possible (e.g., Hepp et al., 2005; Polleres et al., 2006, 510). This will require searching (and binding) of services by constraints or properties. For this reason, the resulting compliance issues should be identified early and relevant solution approaches developed.

4.3 Solution approaches

4.3.1 Previous approaches and their limitations

One solution approach are the *organisational* measures mentioned in Section 4.1, namely approval processes for going live of services and processes as well as restriction of available services. However, depending on the form of dynamic binding applied, these measures are not sufficient, as explained in the previous Section. Furthermore, for reasons of economy and of error proneness, they should be supplemented or even replaced, if possible, by *technical* support. There are several *tools* on the market which are designed to support compliance. An overview of SOX software products is given by Agrawal et al. (2006). However, it should be noted that very few products contain functionalities like checking a business process at runtime (Foody, 2006). Moreover, the control activities have to be implemented manually (Agrawal et al., 2006). Whether it is possible to model in detail and to check at runtime *all* aspects of *all* regulations is questionable.

Solution approaches and their limitations for this problem have already been considered in Section 2.

At this point, the limitations are illustrated with the example of the log services on the basis of the approach from Hepp et al. (2005): At the moment process 1 (with the credit card data) goes live, the possibility of a service (log service 3), non compliant for this process, being added to the repository at a later time, would have to be already considered, and as a precaution, a query generated which lists all processes that need credit card data. When log service 3 is inserted into the repository, again one would have to remember to execute the query and to adapt, if necessary, the constraints based on which the processes are searching. This would require many (error prone) manual interventions.

In all approaches, furthermore, the regulations to be supported have to be determined manually. If a service is selected at runtime and at the same time the selection of a non compliant service is to be avoided, technical support for this problem is necessary, because, as the example of the log services shows, the regulations to be complied with are context sensitive. This means that first of all the context has to be determined, in order to derive from it, in a further step, the regulations to be complied with which then have to be met by the service.

4.3.2 Own solution approach

Request and context
Since it is a matter of the use of the service, the *context* depends on the input data, i.e., the value of the input parameters. One possibility would be to determine it on the basis of the data type of the data to be delivered; however, this presupposes that the data types are exactly defined. Another possibility would be to check the data itself for patterns; however, it is questionable whether a context could be definitely determined from that. Probably the best option is to semantically describe not only the offer, but also the *request*, the process to be invoked, which can in turn be regarded as a service. To do this, a description language has to be chosen which handles the requests separately.

Web services, the currently most frequently considered implementation form of services in a SOA, are usually described in the language WSDL. A WSDL file contains only syntactical information (i.e., *how* the service is invoked), for example the name and the data type of a parameter, but not its semantics (i.e., *what* the service does produce and what the parameter does mean). This is described in unstructured form in a text field or in external documents.

With the addition of semantic meta data, i.e., further description elements, as well as the possibilities of references within the meta data, the text description is structured. Web services, which are semantically described in this way, are called semantic web services (Dostal et al., 2005, 283; Polleres et al., 2006, 509; Bell et al., 2007, 69ff.). For automated processing, a consistent terminology for the meta data is necessary. For semantic web services there are standardised description languages. The main ones are OWL-S, WSMO and WSDL-S. They are explained and compared in Klein (2006) and Polleres et al. (2006). A more recent approach represents SAWSDL.

For our specific problem, these languages were compared in Heim (2008). The comparison focused on the types of ontologies (including expressiveness; see next paragraph), the main elements of the languages, the coverage of the activities ongoing in each phase, the documentation, the distribution of the language and the support by tools. Furthermore, additional criteria used in (Klein, 2006, 39ff., 81) were addressed. Based on the comparison, the languages WSMO and OWL-S were chosen for an implementation. These two languages were chosen specifically because they are the most common and have an integrated ontology language. The latter means that a completely new, holistic framework for semantic service description was defined (top down approach), as opposed to the extension of existing syntactical standards, in particular WSDL, with semantics (bottom up approach). With the bottom up approach, in each case external ontologies have to be referenced, which is considered rather critically in the literature (Lausen et al., 2007, 181). More details to this comparison are given in Loosli et al. (2009).

Regulations

As the next step, the relationship to the *regulations* to be complied with has to be established. An ontology is suitable for doing this. An ontology is a formal description of terms (concepts) and their relations, which is valid within a domain for a group of people (Gruber, 1993; Mädche et al., 2001; Hesse, 2002). It can also contain assertions, which allow logical conclusions (Hench and Fensel, 2008, 13). In the semantic web services description languages, references to such domain ontologies can be established. An ontology refers to the content of the elements contained in the description languages. A consistent use of the terms is especially important, if services are procured from different sources (Mädche et al., 2001, 394).

A distinguishing feature of ontologies is their expressiveness Hepp, 2008, 8ff.: The higher the expressiveness of a description language, the more complex relations can be expressed in the ontology. This allows sophisticated reasoning. As expressiveness increases, however, so too does complexity, and with it, the computational effort for use of the ontology. Also, developing such an ontology needs more effort and it is more difficult to understand (Grimm et al., 2007, 60ff.; Hepp, 2008, 8). The ontology required in this case exceeds a vocabulary and also has to enable logical conclusions to be made (from context to required regulations).

An ontology can also contain synonyms: Thus instead of the common term *credit card data*, users can also state credit card names like *Visa* or *Master Card*. For our application area (domain), these terms are synonyms. It is preferable to use existing ontologies or to take them as a basis, in order to reduce the development and maintenance effort. In our case, legal ontologies (cf. e.g., Gangemi et al., 2005) could prima facie regarded as suitable; but they are strongly oriented to juristic cases or situations, such as ‘acting in the heat of passion’. Consequently, an ontology is preferred which focuses on regulations for which companies have to

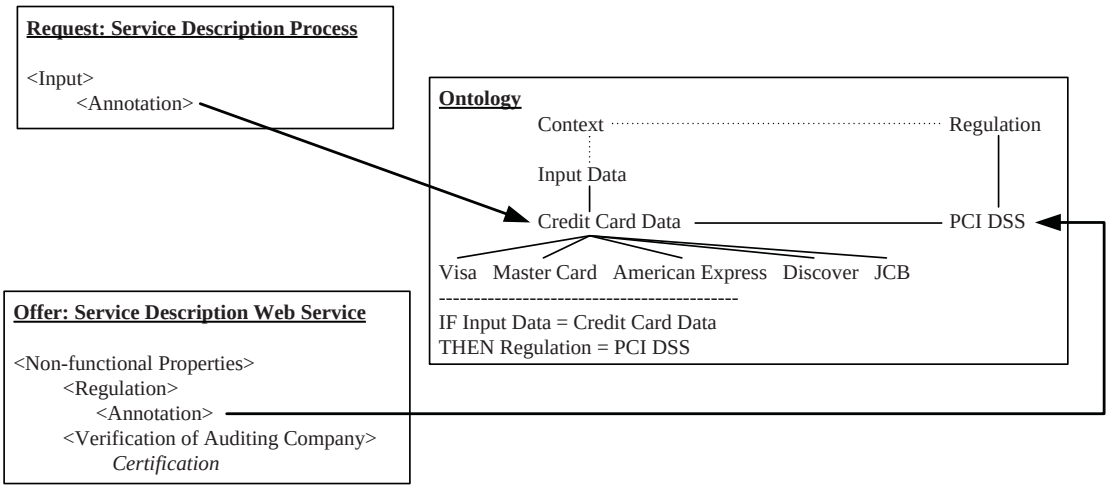

Figure 2: Conceptual solution approach

provide evidence of their compliance. For example, the ontology described in Giblin et al. (2005) could be examined for its suitability.

Offer

One of the proposed organisational measures says that all services in the repository have to be compliant. However, with such a requirement it is unclear to which regulations a service is compliant. To clarify this situation, the description of the *provided* services would have to be expanded accordingly.

In order to enable automated matching between request and offer, the same description language should be used. Also, it should be related to the same ontology. For externally sourced services, the partner company's declaration of the regulations with which their service is compliant could be approved by an auditing or certification company. This is shown in Figure 2: Both the requesting process (input data in a specific context) and the provided web service (compliant to which regulations) are semantically described and relate to the same ontology, in which the mapping from context to required regulations is made.

An ontology should be as broadly applicable as possible (depicted regulations, using companies). Since there are regulations which have to be complied with depending on the industry or other criteria, in a further step the context is extended to include the type of company. This means that different regulations may be relevant, depending on in *which company* and with *what data* the service is used. To check for compliance with a regulation (and any associated certification), auditing standards are frequently used, in which precise test procedures are specified. Auditing standards may relate to frameworks.

This is illustrated in Figure 3. The credit card data example is indicated above the rectangles. All companies (e.g., retailers) which operate with credit card data have to comply with PCI DSS. The requirements are based on the ISO/IEC 17799 standard (now developed further and published as ISO/IEC 27002). The application providers (including service providers) are audited and certificated according to the Payment Application Data Security Standard (PA-DSS), in case the software is provided externally (PA-DSS, 2008). In house developments too have to comply with PCI DSS, but for internal use no service certification (validation) is needed.

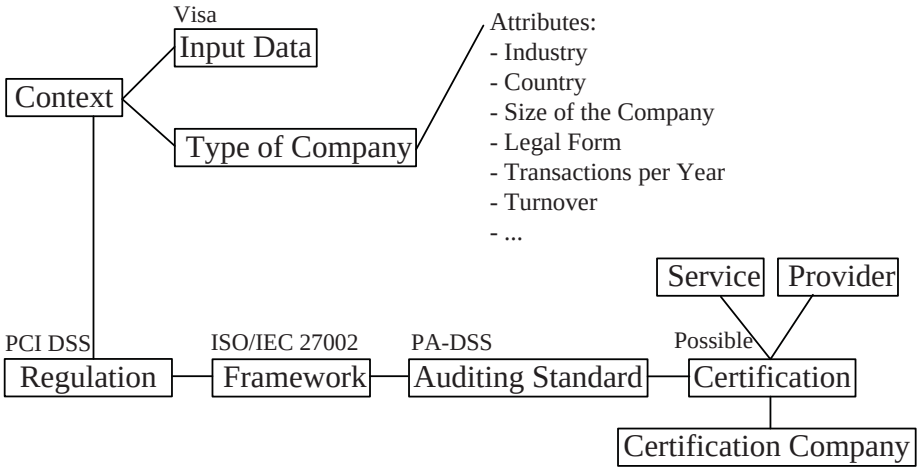

Figure 3: Conceptual ontology approach

5 Summary and Conclusion

Dynamic binding, semantic concepts and governance (Aberdeen Group, 2007, 7ff.) belong to the *grand challenges* of research with respect to Service Oriented Architectures (Papazoglou et al., 2006). Consequently, applications of semantics and, associated with this, of dynamic binding are currently largely absent in practice (Papazoglou et al., 2006; Kuropka et al., 2008, 1). However, potential problems should be already considered today.

In this paper we have shown that the different forms of dynamic binding have diverse implications on compliance with regulations. When applying the most far reaching form of dynamic binding, there is the risk that the reuse of services in a different context may result in the selection of non compliant services. Because previous approaches do not solve this problem, we presented an approach demonstrating how, by means of semantic concepts, the context of service use and, consequently, the required regulations may be determined. This shows that semantic concepts may be helpful not only for selection of services but also for evidence of compliance.

As the next important step, the ontology which was broadly outlined in Figure 3 has to be developed, preferably based on an existing ontology. For the development of the ontology, a bottom up approach was chosen (Stuckenschmidt and van Harmelen, 2005, 82): For this case and a related regulation, a first version was created; in a next step, this will be incrementally extended by considering further regulations and cases. As a basis for existing ontologies, legal ontologies seem to be ill-suited; regulation ontologies are to be examined. Thereby, the ontology description language and its expressiveness also have to be taken into account. The semantic service description languages needed for the implementation have already been selected. With the implementation, a *proof of concept* can be made.

In an additional step, the question of whether the ontology can be extracted (semi-)automatically from regulation texts could be examined. Last but not least, alternate implementation forms with established technologies (such as rules engines for example; cf. Jang and Sohn, 2004) should be examined.

References

Aberdeen Group (2007) Management and Governance: Planning for an Optimized SOA Application Lifecycle. Report, URL

http://www.aberdeen.com/summary/report/benchmark/3944_RA_SOAGov.asp, (2009-09-06)

Agrawal R, Johnson C, Kiernan J, Leymann F (2006) Taming Compliance with Sarbanes-Oxley Internal Controls Using Database Technology. In: Liu L, Reuter A, Whang KY, Zhang J (eds) Proceedings of 22nd International Conference on Data Engineering (ICDE 2006), IEEE Computer Society, Washington, p 92

Bell D, De Cesare S, Iacovelli N, Lycett M, Merico A (2007) A framework for deriving semantic web services. Information Systems Frontiers 9(1):69–84

Berbner R, Heckmann O, Mauthe A, Steinmetz R (2005) Eine Dienstgüte unterstützende Webservice-Architektur für flexible Geschäftsprozesse. WIRTSCHAFTSINFORMATIK 47(4):268–277

vom Brocke J (2008) Serviceorientierte Architekturen — SOA Management und Controlling von Geschäftsprozessen. Vahlen, München

Dostal W, Jeckle M, Melzer I, Zengler B (2005) Service-orientierte Architekturen mit Web Services. Spektrum — Akademischer Verlag, München

Eidg. Bankenkommission (EBK) (2006) Rundschreiben 06/6 — Überwachung und interne Kontrolle. URL http://www.finma.ch/archiv/ebk/d/regulier/rundsch/2006/rs_0606_d.pdf, (2009-09-06)

El Kharbili M, Alves de Medeiros AK, Stein S, van der Aalst WMP (2008) Business Process Compliance Checking: Current State and Future Challenges. In: Loos P, Nüttgens M, Turowski K, Werth D (eds) Modellierung betrieblicher Informationssysteme (MobIS 2008) — Modellierung zwischen SOA und Compliance Management, Lecture Notes in Informatics, vol P-141, Gesellschaft für Informatik, Bonn, pp 107–113

Erl T (2004) Service-Oriented Architecture – Concepts, Technology, and Design. Prentice Hall PTR, Upper Saddle River et al.

Fabini M (2007) Governance für komplexe SOA-Unternehmungen — Eine Vision für das Schweizer Gesundheitswesen. In: Starke G, Tilkov S (eds) SOA-Expertenwissen — Methoden, Konzepte und Praxis serviceorientierter Architekturen, dpunkt.verlag, Heidelberg, pp 309–323

Foody D (2006) The Challenges of SOA — Which rules are necessary and which are just nice to have. SOA Web Services Journal 6(9), URL http://webservices.sys-con.com/read/284550.htm, (2009-09-06)

Gangemi A, Sagri M, Tiscornia D (2005) A constructive framework for legal ontologies. In: Benjamins V, Casanovas P, Breuker J, Gangemi A (eds) Law and the Semantic Web, Springer, Berlin, Heidelberg, pp 97–124

Giblin C, Liu AY, Müller S, Pfitzmann B, Zhou X (2005) Regulations Expressed As Logical Models (REALM). In: Moens MF, Spyns P (eds) Legal Knowledge and Information Systems (JURIX 2005) — Frontiers in Artificial Intelligence and Applications, IOS Press, Amsterdam, pp 37–48

Grimm S, Hitzler P, Abecker A (2007) Knowledge Representation and Ontologies — Logic, Ontologies and Semantic Web Languages. In: Studer R, Grimm S, Abecker A (eds) Semantic Web Services — Concepts, Technologies and Applications, Springer, Berlin, Heidelberg, pp 51–105

Gruber TR (1993) A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition 5(2):199–220

Haniewicz K, Kaczmarek M, Zyskowski D (2008) Semantic Web Services Applications — a Reality Check. WIRTSCHAFTSINFORMATIK 50(1):39–46

Heim D (2008) Darstellung und Vergleich ausgewählter Semantic-Web-Services-Beschreibungssprachen. Master's thesis, Institut für Wirtschaftsinformatik, Universität Bern

Hench G, Fensel D (2008) From Web to Semantic Web. In: Fensel D, Kerrigan M, Zaremba M (eds) Implementing Semantic Web Services, The SESA Framework, Springer, Berlin, Heidelberg, pp 3–25

Hepp M (2008) Ontologies: State of the Art, Business Potential, and Grand Challenges. In: Hepp M, De Leenheer P, de Moor A, Sure Y (eds) Ontology Management — Semantic Web, Semantic Web Services, and Business Applications, Springer, Berlin, Heidelberg, pp 3–22

Hepp M, Leymann F, Bussler C, Domingue J, Wahler A, Fensel D (2005) Semantic Business Process Management: Using Semantic Web Services for Business Process Management. In: Proceedings of IEEE ICEBE 2005, IEEE Computer Society Press, Los Alamitos

Hesse W (2002) Ontologie(n) — Aktuelles Schlagwort. Informatik Spektrum 25(6):477–480

Jang M, Sohn JC (2004) Bossam: An Extended Rule Engine for OWL Inferencing. In: Antoniou G, Boley H (eds) RuleML 2004 — Rules and Rule Markup Languages for the Semantic Web, Lecture Notes in Computer Science, vol 3323/2004, Springer, Berlin, Heidelberg, pp 128–138

Johannsen W, Goeken M (2007) Referenzmodelle für IT-Governance — Strategische Effektivität und Effizienz mit COBIT, ITIL & Co. dpunkt.verlag, Heidelberg

Josuttis N (2008) SOA in der Praxis — System-Design für verteilte Geschäftsprozesse. dpunkt.verlag, Heidelberg

Kalex U (2007) Von der Geschäftsarchitektur zur SOA-Governance. In: Starke G, Tilkov S (eds) SOA-Expertenwissen — Methoden, Konzepte und Praxis serviceorientierter Architekturen, dpunkt.verlag, Heidelberg, pp 325–340

Karagiannis D (2008) A Business Process-Based Modelling Extension for Regulatory Compliance. In: Bichler M, Hess T, Krcmar H, Lechner U, Matthes F, Picot A, Speitkamp B, Wolf P (eds) Proceedings der Multikonferenz Wirtschaftsinformatik (MKWI 2008), GITO-Verlag, München, pp 1159–1173

Keller W (2007) SOA-Governance — SOA langfristig durchsetzen und managen. In: Starke G, Tilkov S (eds) SOA-Expertenwissen — Methoden, Konzepte und Praxis serviceorientierter Architekturen, dpunkt.verlag, Heidelberg, pp 289–307

Klein M (2006) Automatisierung dienstorientierten Rechnens durch semantische Dienstbeschreibungen. PhD thesis, Fakultät für Mathematik und Informatik, Friedrich-Schiller-Universität Jena, Universitätsverlag Karlsruhe, Karlsruhr

Knolmayer GF (2007) Compliance-Nachweise bei Outsourcing von IT-Aufgaben. WIRTSCHAFTSINFORMATIK 49:S98–106

Knolmayer GF, Loosli G (2006) IT Governance. In: Zaugg RJ (ed) Handbuch Kompetenzmanagement — durch Kompetenz nachhaltig Werte schaffen, Haupt Verlag, Bern, Stuttgart, Wien, pp 449–457

Knolmayer GF, Wermelinger T (2006) Der Sarbanes-Oxley Act und seine Auswirkungen auf die Gestaltung von Informationssystemen. In: Siegel T, Klein A, Schneider D, Schwintowski HP (eds) Unternehmungen, Versicherungen und Rechnungswesen, Duncker&Humblot, Berlin, pp 513–536

Kohnke O, Scheffler T, Hock C (2008) SOA-Governance — Ein Ansatz zum Management serviceorientierter Architekturen. WIRTSCHAFTSINFORMATIK 50(5):408–412

Krafzig D, Banke K, Slama D (2004) Enterprise SOA — Service-Oriented Achitecture Best Practices. Prentice Hall PTR, Upper Saddle River et al.

Kuropka D, Tröger P, Staab S, Weske M (eds) (2008) Semantic Service Provisioning. Springer, Berlin, Heidelberg

Lausen H, Lara R, Polleres A, de Bruijn J, Roman D (2007) Description — Semantic Annotation for Web Services. In: Studer R, Grimm S, Abecker A (eds) Semantic Web Services — Concepts, Technologies and Applications, Springer, Berlin, Heidelberg, pp 179–209

Liu Y, Müller S, Xu K (2007) A static compliance-checking framework for business process models. IBM Systems Journal 46(2):335–361

Loosli G, Heim D, Knolmayer GF (2009) IT-Governance bei Wiederverwendung von Services. In: Fischer S, Maehle E, Reischuk R (eds) INFORMATIK 2009, Beiträge der 39. Jahrestagung der Gesellschaft für Informatik, Gesellschaft für Informatik, Bonn, Lecutre Notes in Informatics, pp 478; 3675–3689

Mädche A, Staab S, Studer R (2001) Ontologien. WIRTSCHAFTSINFORMATIK 43(4):393–395

Namiri K, Stojanovic N (2007a) A Model-driven Approach for Internal Controls Compliance in Business Processes. In: Hepp M, Hinkelmann K, Karagiannis D, Klein R, Stojanovic N (eds) Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007), CEUR-WS.org, Aachen, CEUR Workshop Proceedings, vol 251, pp 40–43

Namiri K, Stojanovic N (2007b) Applying Semantics to Sarbanes Oxley Internal Controls Compliance. In: Koschke R, Herzog O, Rödiger KH, Ronthaler M (eds) INFORMATIK 2007: Informatik trifft Logistik, Beiträge der 37. Jahrestagung der Gesellschaft für Informatik, Gesellschaft der Informatik, Bonn, Lecture Notes in Informatics, vol P-109, pp 222–226

NorthPage Research (2009) SOA Governance Resource Guide. URL http://soagovsource.com/, (2009-09-06)

O'Grady S (2004) SOA Meets Compliance: Compliance Oriented Architecture. Study, URL http://redmonk.com/public/COA_Final.pdf, (2009-09-06)

PA-DSS (2008) Payment Application Data Security Standard (PA-DSS). Payment Card Industry (PCI) Security Standards Council, URL http://www.pcisecuritystandards.org/security_standards/pa_dss.shtml, (2009-09-06)

Papazoglou MP, Georgakopoulos D (2003) Service-Oriented Computing. Communications of the ACM 46(10):25–28

Papazoglou MP, Traverso P, Dustdar S, Leymann F, Krämer BJ (2006) Service-Oriented Computing Research Roadmap. In: Cubera F, Krämer BJ, Papazoglou MP (eds) Proceedings of Dagstuhl Seminar 05462 — Service Oriented Computing (SOC), Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), URL http://drops.dagstuhl.de/opus/volltexte/2006/524/pdf/05462.SWM.Paper.524.pdf, (2009-09-06)

Pautasso C, Alonso G (2005) Flexible Binding for Reusable Composition of Web Services. In: Gschwind T, Assmann U, Nierstrasz O (eds) Proceedings of the 4th International Workshop on Software Composition (SC 2005), Springer, Berlin, Heidelberg, pp 151–166

PCAOB (2009) AUDITING STANDARD No. 2: An Audit of Internal Control Over Financial Reporting Performed in Conjunction with An Audit of Financial Statements. Public Company Accounting Oversight Board (PCAOB), URL http://www.pcaobus.org/Rules/Rules_of_the_Board/Auditing_Standard_2.pdf, (2009-09-06)

Polleres A, Lausen H, Lara R (2006) Semantische Beschreibung von Web Services. In: Pellegrini

T, Blumauer A (eds) Semantic Web — Wege zur vernetzten Wissensgesellschaft, Springer, Berlin, Heidelberg, pp 505–524

Schelp J, Stutz M (2007) SOA Governance. HMD — Praxis der Wirtschaftsinformatik 43(253):66–73

Siedersleben J (2007) SOA revisited: Komponentenorientierung bei Systemlandschaften. WIRTSCHAFTSINFORMATIK 49:S110–117, Sonderheft

Spahn M (2007) Ein heuristisches Verfahren zur dienstgütebasierten Optimierung flexibler Geschäftsprozesse. In: Braun T, Carle G, Stiller B (eds) Proceedings der 15. Fachtagung Kommunikation in Verteilten Systemen (KiVS 2007), Springer, Berlin, Heidelberg, pp 315–322

Spahn M, Berbner R, Heckmann O, Steinmetz R (2006) Ein heuristisches Optimierungsverfahren zur dienstgütebasierten Komposition von Web-Service-Workflows. Tech. rep., Technische Universität Darmstadt, URL ftp://ftp.kom.e-technik.tu-darmstadt.de/pub/papers/SBHS06-1-paper.pdf, 2009-09-06

Stuckenschmidt H, van Harmelen F (2005) Information Sharing on the Semantic Web. Springer, Berlin, Heidelberg

Thelesklaf D (2001) Outsourcing von Compliance-Dienstleistungen — Compliance als Teil des Risk Managements. Der Schweizer Treuhänder 75(5):447–452

Tilkov S, Starke G (2007) Einmaleins der serviceorientierten Architekturen. In: Starke G, Tilkov S (eds) SOA-Expertenwissen — Methoden, Konzepte und Praxis serviceorientierter Architekturen, dpunkt.verlag, Heidelberg, pp 9–36

Winter R, Schelp J (2007) Agilität und SOA. In: Starke G, Tilkov S (eds) SOA-Expertenwissen — Methoden, Konzepte und Praxis serviceorientierter Architekturen, dpunkt.verlag, Heidelberg, pp 41–47

Gabriela Loosli

University of Bern
Engehaldenstrasse 8
CH-3012 Bern
Switzerland
Gabriela.Loosli@iwi.unibe.ch