Towards a Middleware and Policy-based Approach to Compliance Management for Collaborative Organizations Interactions

Laura González and Raúl Ruggia
Instituto de Computación, Facultad de Ingeniería, Universidad de la República,
J. Herrera y Reissig 565, Mdeo., Uruguay

Keywords: Compliance Management, Inter-organizational Integration Platform, Policy-based Management, Middleware.

Abstract: Organizations increasingly need to collaborate with each other in order to achieve their business goals, which requires the integration of systems running in different organizations. Such integration is usually supported by middleware-based integration platforms that enable different styles of interactions (e.g. service-oriented, message-based) between heterogeneous and distributed systems. In addition, these collaborative and integrated environments have to satisfy compliance requirements originating from different sources (e.g. laws, agreements) that may, in particular, apply to the interactions between organizations. This paper proposes a middleware and policy-based approach to compliance management for collaborative organizations interactions. The approach comprises design time mechanisms (e.g. a domain specific language, a policy language) and runtime mechanisms (e.g. a policy enforcement point, an obligations service) which extend a middleware-based integration platform. The proposal aims to promote the maintanaibility, flexibility, agility and reuse of compliance solutions in these contexts by providing the means to uniformly specify compliance requirements as well as to define how these requirements are to be managed within an integration platform.

1 INTRODUCTION

Organizations increasingly need to collaborate with each other in order to achieve their business goals. To this end, the integration of software systems running in different organizations is required as it allows carrying out distributed operations in coordinated ways (Di Nitto et al., 2008). This has led to large scale information systems which interconnect the software systems of different, autonomous and geographically distributed organizations sharing common goals (Mecella et al., 2002). This kind of collaborative organizations can be found in different contexts (e.g. e-government, e-health, e-business).

Such integration is usually supported by integration platforms which are specialized middleware-based infrastructures that ease the interconnection of heterogeneous and distributed systems (González and Ruggia, 2015). These platforms allow organizations to collaborate via different styles of interactions (e.g. service-oriented). In contexts like e-government, integration platforms are usually provided by governmental agencies with the aim of promoting the interconnection of organizations by providing services which generate economy of scale (González et al., 2012).

In turn, these collaborative and integrated environments have to satisfy compliance requirements originating from different sources (e.g. laws, agreements, standards) that may apply to each organization as well as to the whole system (Papazoglou, 2011). In particular these requirements may apply to the interactions carried out between organizations. For example, data exchange interactions may have requirements about data privacy or data quality (González et al., 2016).

Although organizations should carry out their own controls to comply with these requirements, there are reasons for performing such controls at integration platforms. First, these platforms may serve as a supervisory party in order to detect situations in which organizations fail to perform the required controls and to act accordingly (e.g. establishing sanctions). Second, some compliance controls may require information which is only available at the platform (e.g. previous interactions). Finally, performing compliance controls in the integration platform, externally to the organization, enables to carry out a non-invasive approach and to generate economy of scale.

Furthermore, while integration platforms provide mechanisms for compliance management in cross-organizational interactions, integration experts within

González, L. and Ruggia, R.
Towards a Middleware and Policy-based Approach to Compliance Management for Collaborative Organizations Interactions.
DOI: 10.5220/0006466504140420
In *Proceedings of the 12th International Conference on Software Technologies (ICSOFT 2017)*, pages 414-420
ISBN: 978-989-758-262-2

organizations have to tackle these issues by developing case-by-case solutions (e.g. removing data in service responses due to privacy issues). Such solutions have limited flexibility and maintainability as well as reusability across areas (e.g. reusing data privacy solutions to enforce quality of service (QoS)).

This paper proposes a middleware and policy-based approach to compliance management for collaborative organizations interactions. On one hand, the approach comprises design time mechanisms including a domain-specific language (DSL), a policy language as well as transformations which allow leveraging existing compliance languages and automatically generating compliance policies. On the other hand, runtime mechanisms consist of components (e.g. a policy enforcement point) which extend an integration platform with policy-based mechanisms in order to enforce the generated compliance policies. The design of these components is based on the well-known Enterprise Integration Patterns (EIP) (Hohpe and Woolf, 2003) (e.g. message transformation) which are usually provided by integration solutions (e.g. Enterprise Service Bus). The proposal aims to promote the maintanaibility, flexibility, agility and reuse of compliance solutions in these contexts by providing the means to uniformly specify compliance requirements and to define how these requirements are to be managed within an integration platform.

The rest of the paper is organized as follows. Section 2 presents background and related work. Section 3 describes the compliance approach. Section 4 presents a life cycle view of the approach. Finally, Section 5 presents conclusions.

2 EXISTING KNOWLEDGE

This section presents background and related work.

2.1 Integration Platforms

The development of large-scale software systems, which often requires the integration of heterogeneous and distributed applications, is usually supported by integration platforms (González and Ruggia, 2015). Integration platforms are specialized infrastructures which provide connectivity and integration capabilities in order to facilitate the integration of systems, in particular, running in different organizations. With this scheme, rather than interacting directly systems interact by exchanging messages through the platform. These messages may be processed by mediation flows in order to perform validations (e.g. regarding access control), monitor interactions (e.g. services response time) and solve integration issues (e.g. data model transformations).

Integration platforms usually have support for building solutions (i.e. mediation flows) based on the well-known EIPs (Hohpe and Woolf, 2003). These patterns, which cover areas such as message transformation and routing, provide a technology-independent way to design and document integration solutions. Integration platforms are implemented through middleware technologies, such as Web Services, Messsage-oriented Middleware and Enterprise Service Bus (ESB) (Chappell, 2004). Finally, the advent of cloud computing promoted the development of cloud-based integration alternatives, such as Integration Platform as a Service (iPaaS), which provide solutions for different cloud integration requirements (e.g. cloud to on-premise, cloud to cloud, on-premise to on-premise)(Pezzini and Lheureux, 2011).

2.2 Policy-based Management

Policy-based management is an approach for guiding the behavior of system elements by the specification of policies (Han and Lei, 2012). The key components of this approach are (Moore et al., 2001)(Westerinen et al., 2001): Policy Repository (PR), Policy Administration Point (PAP), Policy Decision Point (PDP) and Policy Enforcement Point (PEP). Policies are defined using a PAP and stored in a PR. The PDP takes policy decisions, based on the existing policies, for itself or for other components (e.g. a PEP). Finally, the PEP is responsible for enforcing these decisions.

In particular, XACML (OASIS, 2013) follows this approach for access control management. XACML defines a policy language for specifying access control policies as well as the format for requesting and returning access control decisions. The general idea of XACML is that all resource requests pass through the PEP, which obtains access control decisions (e.g. permit, deny) from the PDP. The PEP is responsible for enforcing such decisions as well as for performing the obligations that they may include.

A policy is usually defined through rules which are based, in most policy languages, on one of these paradigms: Event–Condition–Action and Condition–Action (Han and Lei, 2012).

2.3 Compliance Management

Compliance management involves different activities (El Kharbili, 2012)(Reichert and Weber, 2012), in particular: i) compliance modeling deals with extracting requirements from regulations and representing them in a formal and machine-readable way, ii) com-

pliance monitoring continuously checks the compliance of systems while they are running , and iii) compliance enforcement refers to performing actions for avoiding the violation of compliance requirements.

Automating compliance is beneficial to organizations given that, in general, the frequency of compliance audits, monitoring and reporting leads to a more successful compliance management (Sackmann et al., 2008). To this end, methods and mechanisms are required at different abstraction levels (e.g. laws, policies, software systems) (Sackmann et al., 2008).

Compliance may be achieved by design and/or by detection (Sackmann et al., 2008). The "by design" approach comprises refining regulations to the deepest system layer so that non-compliance becomes technically impossible. The "by detection" approach comprises monitoring the operation of systems in order to check if they adhere to the applicable requirements. A comprehensive solution for automating compliance may benefit from both approaches given that applying only one of them may not be technically or economically feasible (Sackmann et al., 2008).

2.4 Related Work

Compliance management has been increasingly addressed during the last decade. In (Sackmann et al., 2008) and in (El Kharbili, 2012) policy-based approaches for compliance management are proposed. The COMPAS project (Tran et al., 2012) developed a view-based and model-driven approach for business process compliance. The C3PRO, (Knuplesch et al., 2013) project provides compliance solutions focusing in inter-organizational business processes.

The distinguished characteristics of our proposal, compared to the former and other work (Hashmi et al., 2016), are: i) the focus on requirements that apply to inter-organizational interactions in collaborative environments, ii) policy-based runtime compliance components provided via an integration platform that processes all the interactions, iii) design time mechanisms promoting the maintanability, flexibility, agility and reuse of compliance solutions, iv) non-invasive solutions (i.e. not requiring deployments in organizations) and v) a compliance management life cycle including adapted activities for the specific context.

3 COMPLIANCE APPROACH

This section presents the proposed approach which, leveraging a middleware-based platform and policy-based mechanisms, allows compliance management in collaborative organizations interactions.

3.1 Working Context Characterization

Figure 1 shows the context of this work: organizations collaborating through an integration platform to achieve common goals (González and Ruggia, 2015).

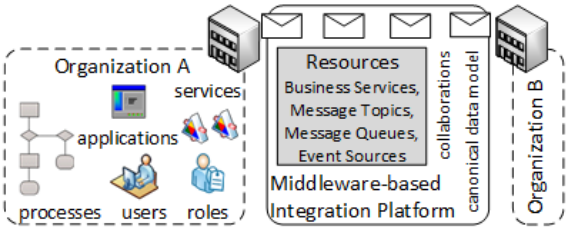

Figure 1: Working Context.

In this context, organizations collaborate with each other by publishing resources (e.g. business services) in the platform and by using resources provided by other organizations. In order to use a resource (e.g. invoke a service) organizations exchange messages through the platform using different interaction styles (e.g. service-oriented). For instance, an organization may send a SOAP message in order to invoke a service provided by other organization. Two or more organizations may agree to collaborate via a predefined set of messages by specifying collaborations.

Internally, each organization may have its own applications, services and processes which are leveraged by users with different roles within the organization. These internal elements support, on one hand, the operation of the resources published in the platform and, on the other hand, may leverage resources provided by other organizations to accomplish their goals.

Messages exchanged through the platform include metadata specifying their origin (e.g. organization) and destination (e.g. a service) as well as providing message identification information (e.g. an id). The platform uses these metadata to route messages to the corresponding resources as well as to perform controls (e.g. security controls). Messages may also include business data structured according to the canonical data model handled by the platform. Figure 2 presents an example showing the structure that messages exchanged through the platform have.

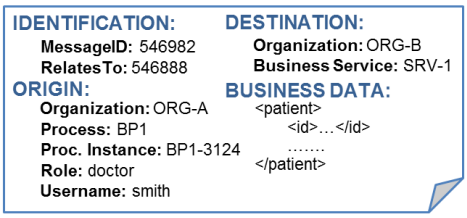

Figure 2: Message Structure.

In this case, the metadata included in the message

specify that a user (smith) is executing a process (BP1) within its organization (ORG-A) which requires invoking a service (SRV-1) of other organization (ORG-B). The role of the user (doctor), the process instance (BP1-3124) and a RelatesTo value, indicating that this message is related to a previous one, are also specified in the metadata. Finally, the message also includes business data (w.r.t. a patient).

3.2 Proposed Compliance Approach

Figure 3 presents a conceptual view of the compliance management approach, which comprises design-time and runtime mechanisms linked by compliance policies, as proposed in (Sackmann et al., 2008).

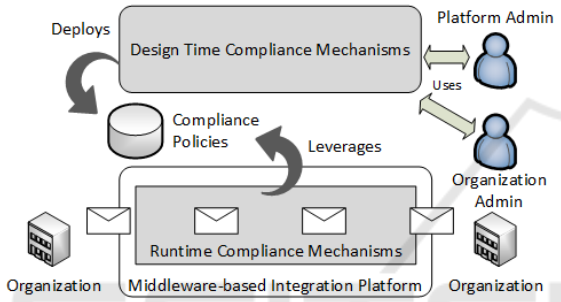

Figure 3: Conceptual View of the Approach.

In particular, design-time mechanisms allow specifying and deploying compliance policies which are then leveraged by an integration platform in order to manage compliance requirements at runtime. These mechanisms are: a Compliance DSL (CompDSL), a Compliance Policy Language (CompPL), transformations to CompDSL and from CompDSL to CompPL, and a Compliance Design Tool (CompDT).

CompDSL allows specifying compliance requirements, applying to inter-organizational interactions, in terms of the elements (e.g. messages, services) handled by integration platforms and independently of the compliance requirements' source or area. Transformations to CompDSL take compliance requirements expressed in different languages (e.g. WSLA) and generate CompDSL requirements. CompPL allows specifying how compliance requirements are to be monitored and enforced by a middleware-based platform. Transformations to CompPL take CompDSL requirements and generate a default set of CompPL policies which can then be modified as required. Finally, CompDT provides support for managing all the aforementioned elements and for deploying CompPL policies into a PR, so they can be monitored and enforced at runtime by a middleware-based platform.

In turn, runtime mechanisms comprise components which extend a middleware-based platform in order to provide policy-based solutions for managing compliance within inter-organizational interactions at runtime. These components are based on the typical architecture of policy-based solutions and include: a Compliance PEP (CompPEP), a Compliance PDP (CompPDP), a Compliance Obligation Service (CompOS) and a Compliance PIP (CompPIP).

All interactions are first processed by the CompPEP which enforces compliance policies according to the decisions taken by the CompPDP. In particular, CompPEP may let messages pass through, may filter messages or may route messages to the CompOS. The CompPDP takes compliance decisions based on information included in the request or obtained through the CompPIP. Finally, the CompOS executes obligation actions according to the decisions taken by the CompPDP.

3.3 Analysis of Technical Feasibility

The runtime components of the solution are not tied up to a specific middleware technology. This provides flexibility as the proposal may be implemented using different integration solutions (Kai Wähner, 2013). Also, the development of these components can leverage native integration capabilities (e.g. EIPs).

The technical feasibility of the runtime components has been partially evaluated in our previous work, which addressed compliance requirements in specific areas by leveraging ESB capabilities.

In (González and Ruggia, 2011) an ESB is extended with adaptation capabilities in order deal with QoS requirements. This proposal uses an Adaptation Gateway with similar capabilities to those of the CompPEP, given that if QoS issues are detected, adaptation actions may be applied. These actions (e.g. message transformation) are based on the EIPs and are similar to the ones that may be applied by the CompOS. The proposal was completely implemented using JBossESB (González et al., 2013).

In (González et al., 2016) solutions based on an ESB and XACML are proposed in order to deal with data privacy requirements in e-government platforms. Obligation actions based on the EIP can be applied on messages by using ESB capabilities. For instance, an obligation action may remove a datum from a message if it can not be shared. This proposal was completely implemented using Switchyard ESB.

Performance tests have been run and presented in both proposals which allows concluding that this type of solutions do not introduce a considerable overhead (e.g. regarding response time) in service invocations.

4 COMPLIANCE LIFE CYCLE

This section describes the proposal from a life cycle point of view.

4.1 Identifying Requirements

The first phase of the life cycle is the identification of compliance requirements of interest. They may originate from different sources (e.g. laws, standards) and may cover different areas (e.g. QoS, data privacy / quality, interaction flows). This task should be performed by business and/or compliance experts.

These compliance requirements should be then specified in a language. While most of compliance sources use natural language, there are machine readable formats for compliance areas such QoS (e.g. QuaLa) and interaction flows (e.g. BPMN). Figure 4 presents a SLA with a QoS requirement specified in the QuaLa DSL developed in the COMPAS project.

```
SampleSLA {  SampleService {
      ProcessingTime<=1min => smsto ...
}}
```

Figure 4: Specification of QoS Requirements in QuaLa.

The SLA in Figure 4 specifies the maximum processing time allowed for the service (one minute) and that an SMS has to be sent if this requirement is not fulfilled.

4.2 Specifying CompDSL Requirements

The second phase of the life cycle is the specification of requirements using CompDSL. If requirements are specified in a machine readable form and the required transformations are defined, requirements may be automatically translated to CompDSL. Otherwise, they have to be manually specified in CompDSL.

CompDSL is a DSL for specifying compliance requirements of inter-organizational interactions performed through integration platforms. These requirements can be monitored by processing all the incoming and outgoing messages in these platforms and can be enforced by performing actions (e.g. transformations) over them. CompDSL bases the specification of compliance requirements in the elements handled by these platforms (e.g. messages, services). This way, CompDSL provides a common specification language for compliance requirements of inter-organizational interactions, which may cover different compliance areas. Note that the specification of some requirements may not be possible in CompDSL, given that they can not be monitored or enforced by the processing of messages within integration platforms.

A CompDSL requirement consists of various elements: compliance scope, message definitions, message conditions, message requirements and compliance actions. A compliance scope determines the subset of messages processed by the platform over which the requirement has to be applied (e.g. the whole platform, an organization, a service). Message definitions declare the messages required to specify the compliance requirement. For example, a service invocation requirement may declare two message definitions in order to represent the request and response. Message conditions specify conditions on message definitions to define groups of messages over which the compliance requirement is to be evaluated. For example, a condition may correlate request messages with response messages. Message requirements allow specifying a compliance requirement in terms of message properties. For example, a response time compliance requirement may be specified in terms of the timestamps of correlated request and response messages. Finally, compliance actions may also be specified in order to monitor, enforce or notify compliance violations. For instance, if the response time of a service is not the expected, an email may be sent to an administrator.

Figures 5 shows how the compliance requirement specified using QuaLa in Figure 4 may be specified using the CompDSL model.

```
Compliance Scope: SampleService
Message Definitions: M1, M2
Messages Conditions:
     M1.id = M2.relatesTo
Messages Requirements:
     Ts(M2) - Ts(M1) <= 1 min
```

Figure 5: Processing Time Requirement in CompDSL.

In the example it is assumed that request and response messages can be correlated using the ”id” and ”relatesTo” values. It is also assumed, that the Ts function returns the timestamp of a message.

4.3 Generating Compliance Policies

The third phase of the life cycle is generating CompPL policies from CompDSL requirements. CompPL policies specify how requirements are to be monitored and enforced by a middleware-based platform enhanced with policy-based mechanisms.

In order to support the automatic generation of CompPL policies from CompDSL requirements, transformations have to be defined. The generated policies may be then modified or extended (e.g. specifying additional means for sending notifications when requirements are not met).

The CompPL language allows specifying policies similarly to XACML (i.e. policy sets, policies, rules) but in terms of the elements handled by integration platforms (i.e. messages, services). Furthermore, decisions may take values such as ”compliant” and ”not compliant”. In addition to obligations, sanctions may be also specified as decisions. Figure 6 presents policies specified in a syntax variant of the Formal Access Control Policy Language (FACPL) (Margheri et al., 2016), which may be generated to monitor and enforce the processing time requirement presented in Figure 5.

```
PolicySet processingTimeRequest {compliant-unless-other
      destination: equal("SampleService", service)
      policies:
            Rule Rule1 (compliant)
obligations:
      [compliant store("timestamp")]
}

PolicySet processingTimeResponse {compliant-unless-other
      destination: equal("SampleServiceResponse", service)
      policies:
            Rule Rule2
             (not-compliant
                  greater-than (Ts(response) - Ts(request),1)
            obligations:
             [not-compliant notify("sms")]
            )
}
```

Figure 6: Processing Time Compliance Policies.

The first policy specifies that all messages sent to SampleService are compliant. It also specifies an obligation which states that a timestamp for each message has to be stored. The second policy specifies that if the difference between the timestamps (Ts) of response and request messages is greater that one minute, then the response message is not compliant. In this case, an obligation is set so that an SMS is sent.

4.4 Enforcing Policies at Runtime

The last phase of the life cycle is the runtime enforcement of policies. Figure 7 shows how the compliance policies presented in Figure 6 are enforced at runtime by the proposed middleware-based platform.

First, a request for invoking SampleService is receive by the platform (1). The request is processed by the CompPEP which builds a compliance decision request and sends it to the CompPDP (2). The CompPDP obtains policies from the PR (3 and 4) and based on them generates a compliance decision response which is sent to the CompPEP (5). As specified in the policies, messages for invoking SampleService are compliant, so the CompPEP let them pass trough. Also, the CompPEP delegates (6) the task of storing timestamps for these messages to the CompOS (7) which, after that, returns control to the CompPEP (8). Then, the CompPEP routes the message in order to complete the invocation of the service (9).

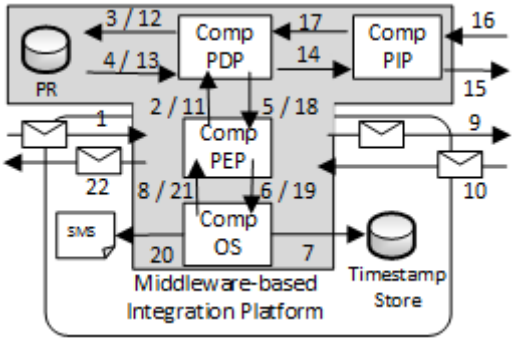

Figure 7: Runtime Operation.

Afterwards, the response message is received and it is also processed by the CompPEP (10) which sends a compliance decision request to the CompPDP (11). As previously, the CompPDP gets policies from the PR (12 and 13) but additional information is required in this case to take the decision (i.e. the timestamp of the request message). So, the CompPDP communicates with the CompPIP to get this additional information (14 and 17). The CompPIP queries a Timestamp API (15 and 16) which gets the required timestamp. The CompPDP is now able to take the decision and sends it to the CompPEP (17). As specified in the policies, if the difference between the timestamps of the response and request messages is greater that one minute, the message is not compliant and an SMS has to be sent. Assuming that this is the case, the CompPEP delegates this action to the CompOS (19), which sends an SMS (20) and returns control to the CompPEP (21). Finally, the response message is returned to the client (22).

5 CONCLUSIONS

This paper presents a middleware and policy-based approach for compliance management in the context of interactions between collaborative organizations.

The here presented proposal focuses on compliance requirements to be satisfied by inter-organizational message-based interactions. This approach enables to cover compliance scenarios involving exchanges between multiple organizations by means of non-invasive enforcement mechanisms external to them. Such scenarios are increasingly relevant as, according to the UN e-Government Survey 2016, the implementation of e-government systems based on cross-organizational coordination is increasingly high in the UN's and governments' agendas. These trends would very likely foster the development of loosely-coupled systems (e.g. message-based) which would require compliance management.

The solution approach includes runtime and design time activities. While the compliance enforcement approach consists of extending an integration platform with runtime policy-based mechanisms, compliance policies are generated at design time by using a domain-specific language, a policy language and transformations.

This work also includes partial feasibility evaluations on the runtime approach using extended ESB components, which is critical for the implementation of this type of solutions. The overall conclusion is that the runtime approach does not introduce relevant overhead in service invocations.

ACKNOWLEDGEMENTS

This work was partially funded by CSIC, Universidad de la República, Uruguay.

REFERENCES

Chappell, D. (2004). *Enterprise Service Bus: Theory in Practice*. O'Reilly Media.

Di Nitto, E., Ghezzi, C., Metzger, A., Papazoglou, M., and Pohl, K. (2008). A journey to highly dynamic, self-adaptive service-based applications. *Automated Software Engineering*, 15(3):313–341.

El Kharbili, M. (2012). Business process regulatory compliance management solution frameworks: A comparative evaluation. In *Proceedings of the Eighth Asia-Pacific Conference on Conceptual Modelling*, volume 130, pages 23–32. Australian Computer Society, Inc.

González, L., Laborde, J. L., Galnares, M., Fenoglio, M., and Ruggia, R. (2013). *An Adaptive Enterprise Service Bus Infrastructure for Service Based Systems*, pages 480–491. Springer International Publishing.

González, L. and Ruggia, R. (2011). Addressing QoS issues in service based systems through an adaptive esb infrastructure. In *Proceedings of the 6th Workshop on Middleware for Service Oriented Computing*, pages 4:1–4:7, New York, NY, USA. ACM.

González, L. and Ruggia, R. (2015). A reference architecture for integration platforms supporting cross-organizational collaboration. In *Proceedings of the 17th International Conference on Information Integration and Web-based Applications & Services*, pages 92:1–92:4, New York, NY, USA. ACM.

González, L., Echevarría, A., Morales, D., and Ruggia, R. (2016). An E-government Interoperability Platform Supporting Personal Data Protection Regulations. *CLEI electronic journal*, 19(2):7:1–7:24.

González, L., Ruggia, R., Abin, J., Llambías, G., Sosa, R., Rienzi, B., Bello, D., and Álvarez, F. (2012). A Service-Oriented Integration Platform to Support a Joined-Up E-Government Approach: The Uruguayan Experience. In *Advancing Democracy, Government and Governance*, LNCS, pages 140–154, Vienna, Austria. Springer Berlin Heidelberg.

Han, W. and Lei, C. (2012). A survey on policy languages in network and security management. *Computer Networks*, 56(1):477 – 489.

Hashmi, M., Governatori, G., and Wynn, M. T. (2016). Normative requirements for regulatory compliance: An abstract formal framework. *Information Systems Frontiers*, 18(3):429–455.

Hohpe, G. and Woolf, B. (2003). *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. Addison-Wesley Professional.

Kai Wähner (2013). Choosing the Right ESB for Your Integration Needs. InfoQ. http://www.infoq.com/.

Knuplesch, D., Reichert, M., Pryss, R., Fdhila, W., and Rinderle-Ma, S. (2013). Ensuring compliance of distributed and collaborative workflows. In *9th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing*.

Margheri, A., Masi, M., Pugliese, R., and Tiezzi, F. (2016). A rigorous framework for specification, analysis and enforcement of access control policies. *CoRR*.

Mecella, M., Scannapieco, M., Virgillito, A., Baldoni, R., Catarci, T., and Batini, C. (2002). Managing data quality in cooperative information systems. In Meersman, R. and Tari, Z., editors, *On the Move to Meaningful Internet Systems 2002 Proceedings*, pages 486–502, Berlin, Heidelberg. Springer Berlin Heidelberg.

Moore, B., Ellesson, E., Strassner, J., and Westerinen, A. (2001). Policy Core Information Model–Version 1 Specification. IETF.

OASIS (2013). eXtensible Access Control Markup Language (XACML) version 3.0.

Papazoglou, M. P. (2011). Making business processes compliant to standards and regulations. In *2011 IEEE 15th International Enterprise Distributed Object Computing Conference*, pages 3–13.

Pezzini, M. and Lheureux, B. J. (2011). Integration Platform as a Service: Moving Integration to the Cloud. Technical Report G00210747, Gartner.

Reichert, M. and Weber, B. (2012). Business Process Compliance. In *Enabling Flexibility in Process-Aware Information Systems*, pages 297–320. Springer.

Sackmann, S., Kähmer, M., Gilliot, M., and Lowis, L. (2008). A classification model for automating compliance. In *10th IEEE Conference on E-Commerce Technology and the Fifth IEEE Conference on Enterprise Computing, E-Commerce and E-Services*.

Tran, H., Zdun, U., Holmes, T., Oberortner, E., Mulo, E., and Dustdar, S. (2012). Compliance in service-oriented architectures: A model-driven and view-based approach. *Information and Software Technology*, 54(6):531 – 552.

Westerinen, A., Schnizlein, J., Strassner, J., Scherling, M., Quinn, B., Herzog, S., Huynh, A., Carlson, M., Perry, J., and Waldbusser, S. (2001). Terminology for policy-based management. IETF.