An End-to-End Framework for Business Compliance in Process-Driven SOAs

Huy Tran*, Ta'id Holmes*, Ernst Oberortner*, Emmanuel Mulo*, Agnieszka Betkowska Cavalcante‡,
Jacek Serafinski‡, Marek Tluczek‡, Aliaksandr Birukou§, Florian Daniel§, Patricia Silveira§, Uwe Zdun†, Schahram Dustdar*
*Distributed Systems Group, Institute of Information Systems, Vienna University of Technology, Austria
Email: {htran|tholmes|e.oberortner|e.mulo|dustdar}@infosys.tuwien.ac.at
‡Telcordia Poland
Email: {abetkows|jserafin|mtluczek}@telcordia.com
§University of Trento, Italy
Email: {birukou|daniel|silveira}@disi.unitn.it
†Software Architecture Group, Department of Distributed and Multimedia Systems, University of Vienna, Austria
Email: uwe.zdun@univie.ac.at

***Abstract*—It is significant for companies to ensure their businesses conforming to relevant policies, laws, and regulations as the consequences of infringement can be serious. Unfortunately, the divergence and frequent changes of different compliance sources make it hard to systematically and quickly accommodate new compliance requirements due to the lack of an adequate methodology for system and compliance engineering. In addition, the difference of perception and expertise of multiple stakeholders involving in system and compliance engineering further complicates the analyzing, implementing, and assessing of compliance. For these reasons, in many cases, business compliance today is reached on a per-case basis by using ad hoc, hand-crafted solutions for specific rules to which they must comply. This leads in the long run to problems regarding complexity, understandability, and maintainability of compliance concerns in a SOA. To address the aforementioned challenges, we present in this invited paper a comprehensive SOA business compliance software framework that enables a business to express, implement, monitor, and govern compliance concerns.**

***Keywords*-Business compliance, process-driven SOA, view-based, model-driven development, domain-specific languages, event processing, runtime monitoring, governance dashboard.**

I. Introduction

A service is a distributed object that is accessible via the network and has certain characteristics: The service offers a public interface and is both platform- and protocol-independent. Service-oriented Computing (SOC) is the paradigm in which services are used as the main constructs for composing distributed systems. Service-oriented Architecture (SOA) is the main architectural style for SOC. In this paper we focus on a particular kind of SOAs, which is process-driven. In a *process-driven SOA*, a process engine is used to orchestrate services in order to implement business processes.

This paper deals with issues of compliance in process-driven SOAs. IT compliance means in general complying with laws and regulations applying to an IT system, such as the Basel II Accord[1], the International Financial Reporting Standards[2], the French financial security law[3], and the Sarbanes-Oxley Act[4]. Laws and regulations are, however, just one example of compliance concerns that occur in process-driven SOAs. There are many other rules and constraints in a SOA that have similar characteristics. Some examples are service composition and deployment rules, service execution order rules, security policies, quality of services (QoS) rules, or licenses.

In the ideal case, a software framework for automatically dealing with all kinds of compliance would be provided. Unfortunately, there is the problem that it is almost impossible to formalize all details of a jurisdictional text, as they are usually subject to interpretation by domain experts or judicial experts and typically contain complex references to other (jurisdictional) texts. For this and other reasons, today, in many cases, compliance is reached on a per-case basis using hard-coded solutions that span diverse components of the SOA.

The consequence is that systems containing implementations of compliance concerns are hard to maintain, hard to evolve or change, hard to reuse, and hard to understand. It is difficult to ensure guaranteed compliance to a given set of rules and regulations, as well as to keep up with constant changes in regulations and laws. In many cases, domain experts are not involved enough in the system design, and hence often compliance implementations are missing important domain aspects.

In this paper, we propose an end-to-end approach to business compliance to overcome these issues in the domain of process-driven SOAs. In particular, our approach offers novel techniques for the whole software and compliance life cycle including modeling, implementing, monitoring,

[1]http://bis.org/publ/bcbs107.htm
[2]http://www.ifrs.org/IFRSs
[3]http://senat.fr/leg/pjl02-166.html
[4]http://gpo.gov/fdsys/pkg/PLAW-107publ204/content-detail.html

and governance. At design time, the view-based modeling framework, domain-specific languages tooling, and model repositories shall support the development and modeling of processes and compliance concerns. During run-time, the compliance governance framework shall provide mechanisms for monitoring compliance concerns, detecting compliance violations, and reporting to stakeholders.

The remainder of this paper is organized as follows. In Section II, a Mobile Virtual Network Operator process extracted from a real industrial case study is exemplified to illustrate our approach and the realization of our approach. Section III describes the overall architecture of the proposed end-to-end business compliance framework. The view-based modeling framework, domain-specific languages tooling, and a model-aware service environment are presented in Section IV whilst the compliance governance framework is elaborated in Section V. Section VI discusses the relevant literature. Finally, we conclude the paper in Section VII.

II. Case Study

Modern mobile telecommunication infrastructures are rapidly expanding into SOAs. This is motivated by the need for delivery of advanced multimedia voice and video services to the users through new telecom service delivery platforms. It is crucial to provide real-time monitoring of such services (e.g. the services' QoS parameters and licensing clauses) and adaptability mechanisms to react to any compliance violations.

In this paper, we demonstrate our approach in a Mobile Virtual Network Operator (MVNO) process that provides advanced telecom services such as on-demand aggregated audio and video streaming content (see Figure 1). The MVNO process, which is offered by a fictitious company, namely, WatchMe, shall serve as a proxy between customers and audio and video streaming providers. As the MVNO process executes, a customer can log in and search for the audio and video content of his choice. As the search complete successfully, the customer is provided the requested media streams. In this way, the MVNO process enables the customers to watch, for example, a selected sport event with an audio commentary in the chosen language.

There are several business compliance requirements that the MVNO process must comply with. For instance, the MVNO process must ensure particular quality levels of provided services, the protection of customer personal and payment data, the compliance with the licenses of content providers, and so forth. The terms and conditions of the offered services are regulated by appropriate Service Level Agreements (SLAs) that contain various compliance requirements. In this paper, we illustrate our approach with two compliance concerns that are quality of services (QoS) and licensing. Nevertheless, our approach is applicable to other compliance concerns as well.

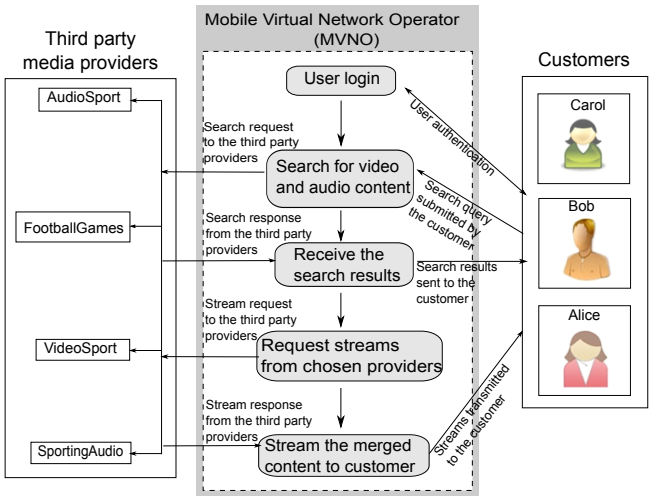

Figure 1. The scenario of the MVNO case study

Compliance Concern	Example Compliance Requirements
QoS: Availability	The availability of the *Login* and *Search* services must be more than 99%.
QoS: Processing-Time	The *Search* service must return useful searching results within less than 2 minutes.
QoS: Minimal-FrameRate	The minimal frame rate that the *Stream* service provides is at least 15 frames per second (fps) .
Licensing: Composition permission	*Only pre-defined combinations of video and audio providers are allowed* due to the licenses specified by the video provider.
Licensing: Pay-per-view plan	The WatchMe enterprise acquires a *limited* number of streams based on *the amount paid* to the media supplier.
Licensing: Time-based plan	The WatchMe enterprise acquires *any* number of times *any* possible streams in a certain period, based on the *amount paid* to the media supplier.

Table I
Excerpt of compliance requirements of the MVNO process

As illustrated in Table I, there are many QoS and licensing compliance requirements associated with the MVNO process and accompanying services to ensure that the process is compliant to the negotiated SLAs. It is crucial to monitor and avoid any potential compliance violations which lowers the services' quality offered to its customers. Monitoring any performance drops of the quality of the third parties' services is also important because these can impact the overall performance of the MVNO process. In addition, some licensing compliance requirements must be satisfied in order to ensure that the data streams are properly delivered to the customers according to the contract established between

the WatchMe company and the content providers.

III. Conceptual Architecture

Our end-to-end approach to business compliance is achieved through a conceptual architecture illustrated in Figure 2. This architecture covers the whole life cycle of business compliance at design time and runtime.

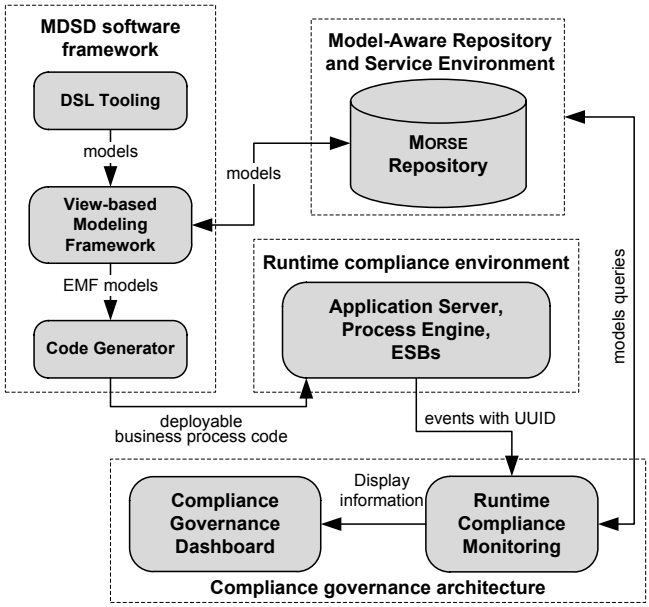

Figure 2. Overview of the Architecture

At design time, the View-based Modeling Framework (VbMF) along with the domain-specific language (DSL) tooling and model repository enable modeling and sharing process-driven SOAs and compliance concerns that we would like to address. The models capture a process-driven SOA in form of architectural views, with each view providing a distinct perspective (concern) of the SOA [8, 11, 20, 21]. The domain-specific languages complement these view models with the specifications of compliance concerns. Finally, the design time components generate code that is deployed to the various platforms to be executed.

At runtime we have the execution environment to which code is deployed, for example, business processes are executed by a process engine. Dynamic verification and validation at run-time are performed by rule-based event-driven monitoring. The runtime compliance monitoring component can query the Model-Aware Repository and Service Environment (MORSE) to leverage models for runtime analysis and reasoning [9]. The monitoring results are assembled for comprehensive reporting on a compliance governance dashboard.

IV. Process-driven SOAs Development and Compliance Modeling

A. View-based Modeling Framework

The View-based Modeling Framework (VbMF) [20, 21] exploits the notion of architectural views [11] to reduce the complexity of software development in process-driven SOAs. In our approach, each view model is a (semi-)formalized representation of a particular process-driven SOA concern such as the control flow, service interactions, data handling, message exchange, or human interaction [8, 14, 20–23]. All VbMF view models are built up around a Core model as shown in Figure 3. The Core model plays an important role in our approach because it provides essential concepts for extending and integrating view models, and establishing and maintaining the dependencies between view models [20–23]. A new concern, for instance, a certain compliance concern, can be integrated into our approach by using a corresponding *New-Concern*-View model that extends the concepts of the Core model and defines additional (and/or domain-specific) concepts of that concern.

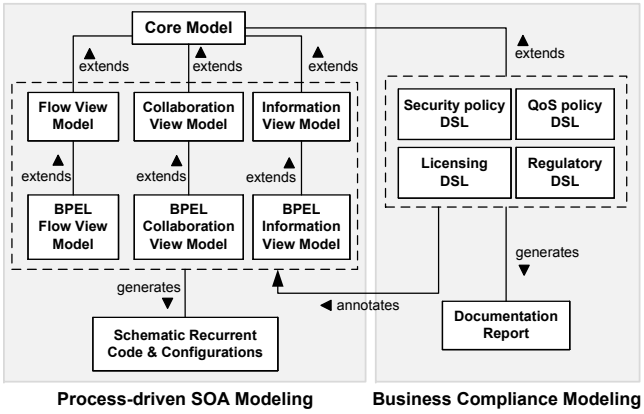

Figure 3. Overview of the View-based Modeling Framework

In addition, VbMF facilitates the model-driven development (MDD) paradigm to offer view models tailored to particular expertise and interests of the involving stakeholders at different abstraction levels. Views belonging to the abstract layer represent high-level or domain concepts that the business and domain experts can understand and manipulate. Then, the IT experts can refine or map these abstract concepts into the platform- and technology-specific views (see Figure 3) that enrich the abstract view models with more technology-specific details [20, 21]. Last but not least, VbMF provides code generations that take these views as inputs and generate process implementations and deployment configurations. In addition, VbMF code generators can also insert appropriate traceability meta-data in the generated process descriptions and/or configurations in order to enable

the tracing from the running process to the corresponding process models. This will be elaborated in Section IV-C.

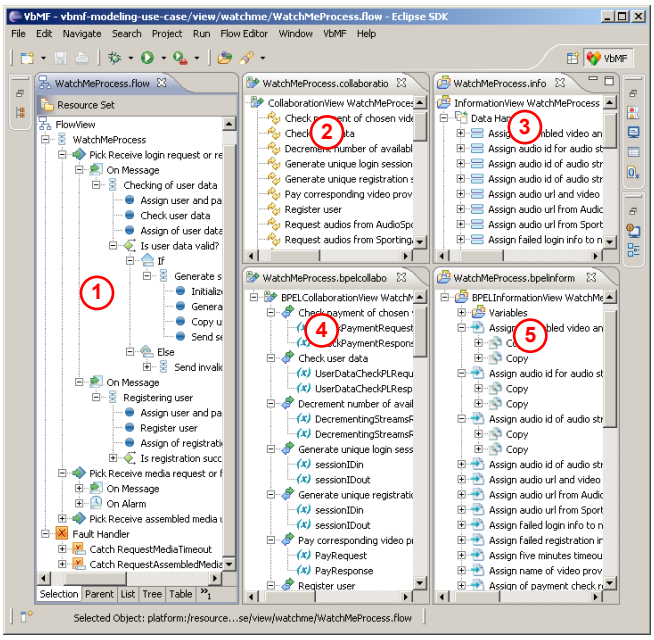

Figure 4. MVNO process development in VbMF: (1) The FlowView, (2-3) The high-level CollaborationView and InformationView, and (4-5) The low-level, technology-specific BpelCollaborationView and BpelInformationView

Figure 4 shows the MVNO process implemented using VbMF. The business and domain experts sketch out the fundamental behavior of the MVNO process to achieve the business goal using the Flow view as well as define high-level business objects such as customer requests using the abstract Information view. The IT experts will refine these concepts in the low-level views that are specific to the BPEL and Web services technology. The process implementation in form of Business Process Execution Language (BPEL)[5] and Web Services Description Language (WSDL)[6] code can be quickly generated out of the aforementioned views for deploying and testing.

B. QuaLa: A DSL for Specifying QoS Compliance Concerns

So far we have presented the development of process-driven SOAs via the view-based modeling framework (i.e., the left part of Figure 3). Next, we elaborate the DSL tooling for modeling compliance requirements relating to process-driven SOA concepts and elements along with a model repository for storing, sharing, and inquiring about process and compliance models.

To offer expressive and convenient languages for the different stakeholders, our approach provides a separation of DSLs into multiple sub-languages, where each sub-language

[5]http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html

[6]http://www.w3.org/TR/wsdl

```
WatchmeSLA {
  Search {
    ProcessingTime<2min
    => mailto "abc@abc.at"
  }
}
```

(a) High-level QuaLa

```
#measuring the ProcessingTime
ProcessingTime chain ServerIn
ProcessingTime phases
     {InPreInvoke InInvoke}

#Location of WSDL file
Search wsdl "./search.wsdl"
```

(b) Low-level QuaLa

Figure 5. Quality of service language (QuaLa)

is tailored for the appropriate stakeholders [17]. We divide our Quality of Service Language (QuaLa) into two sub-languages. The first language, the high-level QuaLa, is tailored for experts of the QoS domain and offers expressive notations for specifying the required QoS compliance concerns. Hence, the high-level QuaLa serves for specifying *which* QoS compliance concerns to monitor. In Figure 5(a) we illustrate an example of specifying one of service's QoS constraints presented in Table I using our high-level QuaLa.

The second language, the low-level QuaLa, extends the high-level QuaLa for specifying the required technological aspects that are needed for monitoring the QoS constraints during the runtime of the system. Hence, the low-level QuaLa serves for specifying *how* to monitor the corresponding QoS compliance concerns. In Figure 5(b) we present our low-level QuaLa and how to use it for specifying the technological aspects.

Based on the high- and low-level specifications, the QuaLa code generator generates interceptors for measuring the services' QoS properties at runtime. The interceptors send events to the runtime compliance governance components, which are responsible for checking the QoS compliance concerns.

C. Model-Aware Repository and Service-Environment

In our approach, all models (i.e., meta-models and conforming models (cf. [4]) are stored in a model repository. For this we have employed MORSE [9, 10] as a central component of the architecture as shown in Figure 2. For the compliance monitoring of the business processes, different services in the SOA reflect on models. Using MORSE they can look-up these models and related models dynamically and in a service-oriented fashion at runtime.

Our approach applies the MDD paradigm to generate service description, process code, deployment artifacts, and monitoring directives out of the VbMF view models. In a typical MDD approach, there are no backward or traceability links in the sense that the generated source code "knows" from which model it has been created. For correlating model instances or code at runtime with source models or model instances, respectively, traceability of model transformations is essential.

To overcome this limitation in order to achieve traceability, models (as the output of the generator) can hold a reference to their source models. As MDD artifacts in

```
<process name="WatchMe">
  <extensions>
    <extension mustUnderstand="yes" namespace=
"http://xml.vitalab.tuwien.ac.at/ns/morse/traceability.xsd"
    />
  </extensions>
  <morse:traceability
     build="a3627172-38bc-4ff3-96d9-dc814d3a7ab2">
    <row query="/process[1]">
      <uuid>b0e5aad1-8aa1-406d-859c-324caf6044db</uuid>
    </row>
    <row query="/process[1]/sequence[1]/receive[1]">
      <uuid>1f56c377-7d12-436b-9760-349a0979df49</uuid>
    </row>
    <row query="/process[1]/sequence[1]/invoke[1]">
      <uuid>16a7749e-9560-4194-b9a8-ab04d6d8f2c9</uuid>
    </row>
    <row query="/process[1]/sequence[1]/invoke[2]">
      <uuid>16a7749e-9560-4194-b9a8-ab04d6d8f2c9</uuid>
    </row>
  </morse:traceability>
  <sequence>
    <!-- ... //-->
  </sequence>
</process>
```

Listing 1. MVNO BPEL process with extensions for MORSE traceability

MORSE repositories are identifiable by Universally Unique Identifiers (UUIDs)[7], MORSE annotates the destination models with the UUIDs of the models. The VbMF code generators automatically insert references to these UUIDs into the generated source code or configuration directives, so that the corresponding models can be identified and accessed from the running system. The code in Listing 1 shows a generated BPEL process description that contains traceability information as a BPEL extension. The BPEL process engine shall emit events for, e.g., the process activities that contain matching UUIDs. Finally, the events will be processed by the monitoring and governance infrastructure.

V. RUN-TIME COMPLIANCE GOVERNANCE

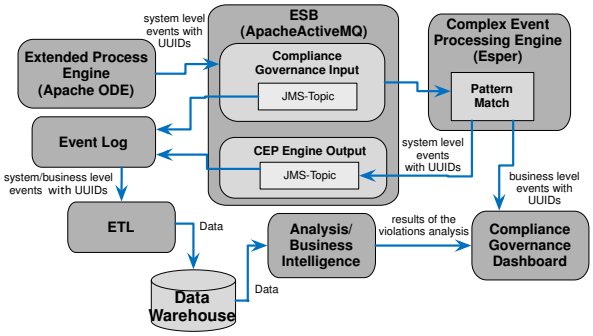

Figure 6. Runtime compliance governance architecture

In this section we present the compliance governance framework for monitoring the compliance of business processes at runtime. It is the bottom part of the architecture depicted in Figure 2, and its detailed view is provided in Figure 6. Runtime governance starts with deploying a BPEL business process to the Apache ODE process engine. The process contains the UUIDs of the process model and the UUIDs of the activities relevant for monitoring compliance requirements. After the deployment a Process Deployed system-level event is emitted in the Apache ActiveMQ Enterprise Service Bus (ESB).

Both the Event Log and the Complex Event Processing (CEP) engine are subscribed to the ESB to receive all system-level events relevant to runtime compliance monitoring. The goal of CEP is to detected violations (complex event patterns) within the low-level streams of events generated by the process engine. The results of CEP will be shown in the Compliance Governance Dashboard, allowing for near real-time detection of violation patterns of events, which could lead to violations of the licenses of the content providers. Hence, it is possible to verify event violations detected on the fly and take actions to avoid such violations in the future. In this way, the dashboard allows business process experts to react quickly and efficiently to violations.

The Event Log stores all low- and high-level events. The Extract, Transform, and Load (ETL) routines extract the data from the Event Log and load them into the Data Warehouse. The Analysis/Business Intelligence component is used to perform advanced off-line analysis of the historical data about compliance violations to raise awareness of the overall compliance status of the company. The MORSE repository may be queried in order to perform drill-down analysis to see in detail what at lower levels led to violations. The results of the off-line compliance monitoring of all compliance requirements (e.g., QoS and Licensing) are shown on the Compliance Governance Dashboard. Section V-B describes the use of the dashboard in the MVNO scenario. In the subsequent sections, we describe the major components of the runtime compliance governance framework including the Rule-based, Event-driven Compliance Monitoring component and the Compliance Governance Dashboard. For further details of the rest of the runtime compliance governance framework, please consult our previous work in [5].

A. Rule-based Event-driven Compliance Monitoring

In order to be able to quickly react to compliance violations, it is important to conduct the online monitoring of process execution. In our approach, we realize the runtime monitoring using complex event processing (CEP) for efficient and fast detection of events that match violation patterns. The process engine generates events during the course of process execution. The CEP engine aggregates the events and evaluates them according to a number of predefined rules. Listing 2 presents an excerpt of the event-based rules for monitoring the violations of the composition permission licensing concern shown in Table I. Those rules are used by the CEP engine to detect patterns of video and

[7]http://itu.int/ITU-T/studygroups/com17/oid/X.667-E.pdf

audio request events that are not compliant with a certain license.

In this case, the pattern includes combinations of *WatchMeGetAudioStream* events from the audio stream of *AudioSport* and from the video stream of *VideoSport* in a certain session. The query has to match only the events related to the same session (matching is done by the "*sessionID*" property of the events). CEP allows for constant, real-time accumulation of events collected to evaluate the event patterns specified within the rules. Thus, any compliance violation specified by the CEP rule is evaluated in real-time, i.e., just in time when the last event required to match the event pattern arrives. When the rule is positively evaluated, the proper notification of compliance violation is immediately sent to run-time compliance dashboard. The notifications are also stored for off-line processing and analysis.

```
select * from pattern [ every (
VidProvVideoSport = Event
(name = 'WatchMeGetVideoStreamEvent'
AND VideoProviderID= 'VideoSport')
AND (AudProvAudioSport = Event
(name = 'WatchMeGetAudioStreamEvent'
AND NOT (AudioProviderID = 'AudioSport'))))]
where AudProvAudioSport.sessionID =
VidProvVideoSport.sessionID
```

Listing 2. An example of event monitoring rules in the MVNO process

B. Compliance Governance Dashboard

The current state of compliance of the organization's business processes is shown in the Compliance Governance Dashboard (CGD) that comprises a number of Key Compliance Indicators (KCIs) widgets and interactive tables. It is possible to quickly check violations in different perspectives (e.g., business vs. compliance) and summarization levels (e.g., compliance source, requirement, or policies, which group related requirements, such as licensing requirements). In our approach, KCIs are defined and their values are displayed in the dashboard. Having both business and compliance perspective and different summarization levels, it is possible to show high-level information (e.g., KCIs of compliance sources) useful for CEOs and CFOs and low-level information (e.g., list of violations events per compliance requirement) useful to technical experts.

Figure 7 shows the monitoring of compliance requirements in the MVNO process. The top left part contains the KCIs of different business process activities in descendant order, where the first widget shows the compliance source or activities with the lowest compliance performance (the worst case). The results of monitoring QoS compliance requirements from Table I are reported in the top right part of the CGD. In addition, the CGD also offers interactive tables (the bottom part) that enable users to inspect the details of KCIs from the highest level information to the lowest level. The values showed by the KCIs are calculated based on the compliance requirements and the data stored into the Data Warehouse. For example, looking at the interactive table the business and domain experts can see the high-level KCIs showing that the Composition permission concern is 100% compliant whilst the monitoring of QoS concerns shows unexpected results that need to be investigated and addressed. Furthermore, the IT experts can click on these high-level KCIs to drill down and investigate the root causes and technical details of the compliance results. For more technical details about the CGD design and implementation, we recommend the readers to consult [18] and the CGD website[8].

VI. Related Work

In this section, we discuss the major related works in the area of an end-to-end framework for business compliance in general as well as model repositories and compliance runtime monitoring and governance. We categorize compliance into two main strategies: "*compliance at design time*", i.e., the implementation of compliance through designing it into a system, and "*compliance at detection runtime*", i.e., the implementation of compliance by monitoring a system's compliance state.

For checking "compliance at design time", most of the existing approaches focus on one single specific compliance domain. For example, the European MASTER project [13] introduces a full life cycle for modeling, assessing, and monitoring security related business compliance concerns. We deal with multiple domains, especially in this work with QoS and licensing. Our approach enables the adaptation to various domains of compliance by extending the conceptual model for compliance governance. Daniel et al. [7] introduce the related components in the compliance governance architecture accordingly.

Namiri et al. [16] support compliance experts to add control patterns to the business process models to make the processes compliant. In our work, we concentrate to support the stakeholders with tailored model-driven DSLs that automatically transform the compliance requirements into rules that are checked during the system's runtime.

AMOR [1], Odyssey-VCS 2 [15], and EMFStore [12] are model repositories that follow a similar approach to our MORSE approach. These repositories have a focus on the versioning aspect of model management (see also [2]). In contrast, MORSE abstracts from modeling technologies and its UUID-based implementation allows for a straightforward identification of models and model elements. None of the mentioned model repositories offers an integration scheme for runtime events, like our approach, or automated model generation and deployment capabilities.

For checking "compliance at runtime", Sriraman et al. [19] focus on business utility and agility provided by

[8]http://compas.disi.unitn.it/CGD/home.html

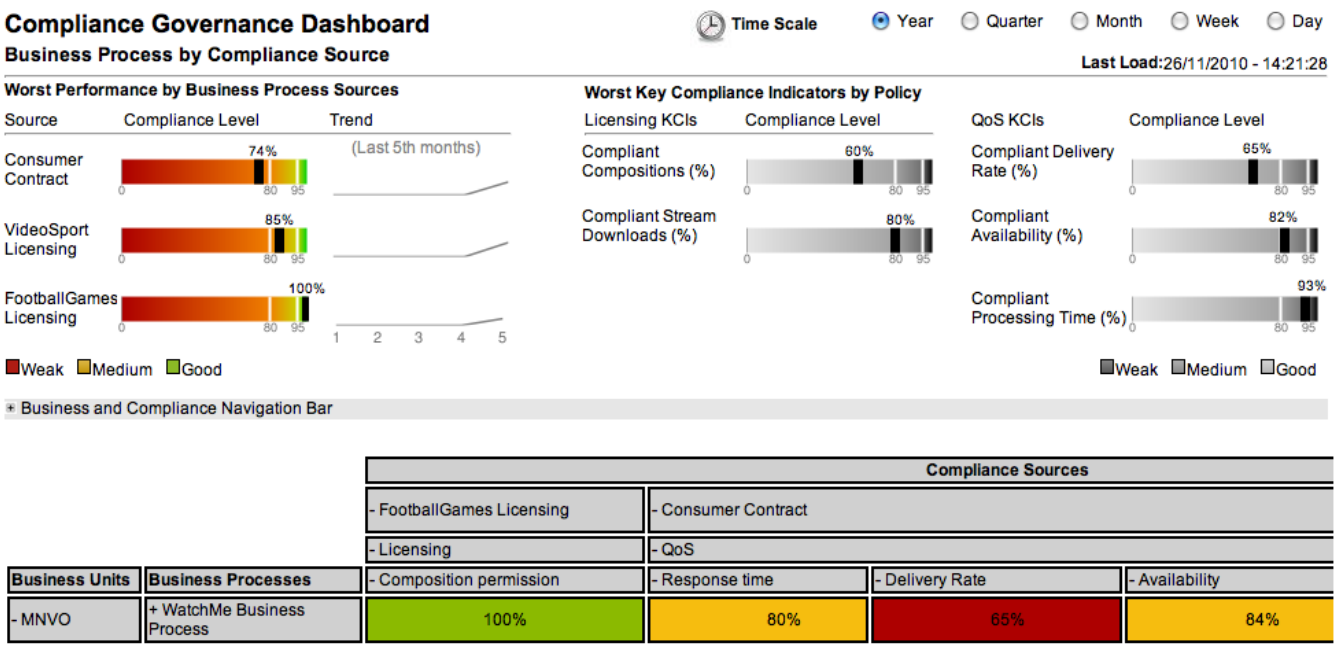

Figure 7. Monitoring compliance using the Compliance Governance Dashboard

the union of SOA, event-driven architecture and model-driven architecture. The approach does not consider monitoring and reasoning business compliance. During the last decade Business Activity Monitoring has gained a lot of attention, and dedicated tools have been proposed to support it (e.g., Oracle Business Activity Monitoring, HP Business Availability Center, Nimbus IBM Tivoli, among others). The compliance management part of these tools, if any, comes in the form of monitoring of SLA violations, which need the SLA formal specifications as one of their inputs. In our research, we adopt a more general view on compliance (beyond SLAs, which are a special case to us) and cover the whole life cycle of compliance governance, including an appropriate dashboard for reporting purposes. Such tools still do not have the capability to process and interpret generic events. They only support the definition of thresholds for parameters or SLAs to be monitored. Also, the ability to compare monitored business process executions or, more in general, business patterns with expected execution behaviors is not supported.

To the best of our knowledge, there are no research approaches that specifically address the issue of visualizing compliance concerns using dashboards. Existing approaches, such as described in [3] or [6], do not provide suitable navigation models supporting different analysis perspectives, summarization levels, and user roles.

VII. Conclusion

In this invited paper, we have presented an end-to-end approach and architecture for dealing with business compliance in process-driven SOAs. In summary, our approach supports stakeholders to deal with the variety of compliance requirements, including, but not limited to, QoS policies or license policies. The presented view-based, model-driven framework lays a solid foundation for modeling process-driven SOAs and *compliance engineering*. DSLs and view models together can be tailored to present compliance concerns to each stakeholder in an understandable form. During the course of process execution, a runtime governance infrastructure enacts the detection of compliance violations and compliance reporting according to the monitoring directives generated from the compliance DSL models. For this, traceability information emitted in process events is used and dynamic model look-up is supported through a model-aware service environment that consolidates design- and runtime use and management of models. Finally, the stakeholders, such as business analysts, IT experts, and end-users, can use the compliance governance dashboard to observe and analyze the current status of software systems' compliance, the root cause of any compliance violations, and so forth.

Acknowledgment

This work was supported by the European Union FP7 project COMPAS, grant no. 215175.

References

[1] K. Altmanninger, G. Kappel, A. Kusel, W. Retschitzegger, W. Schwinger, M. Seidl, and M. Wimmer, "AMOR – towards adaptable model versioning," in *1st International Workshop on Model Co-Evolution and Consis-*

tency Management, in conjunction with MODELS '08, 2008.

[2] K. Altmanninger, M. Seidl, and M. Wimmer, "A survey on model versioning approaches," *IJWIS*, vol. 5, no. 3, pp. 271–304, 2009.

[3] R. Bellamy, T. Erickson, B. Fuller, W. Kellogg, R. Rosenbaum, J. C. Thomas, and T. V. Wolf, "Seeing is believing: Designing visualizations for managing risk and compliance," *IBM Systems Journal*, vol. 46, no. 2, pp. 205–218, 2007.

[4] J. Bézivin, "On the unification power of models," *Software and System Modeling*, vol. 4, no. 2, pp. 171–188, 2005.

[5] A. Birukou, V. D'Andrea, F. Leymann, J. Serafinski, P. Silveira, S. Strauch, and M. Tluczek, "An integrated solution for runtime compliance governance in SOA," in *Proc. of the 8th International Conference on Service-Oriented Computing (ICSOC10)*, 2010.

[6] P. Chowdhary, T. Palpanas, F. Pinel, S.-K. Chen, and F. Y. Wu, "Model-driven dashboards for business performance reporting," in *IEEE Int'l Enterprise Distributed Object Computing Conference*. IEEE Computer Society, 2006, pp. 374–386.

[7] F. Daniel, F. Casati, V. D'Andrea, S. Strauch, D. Schumm, F. Leymann, E. Mulo, U. Zdun, S. Dustdar, S. Sebahi, F. de Marchi, and M.-S. Hacid, "Business compliance governance in service-oriented architectures," in *Proceedings of the IEEE Twenty-Third International Conference on Advanced Information Networking and Applications (AINA'09)*, May 2009.

[8] T. Holmes, H. Tran, U. Zdun, and S. Dustdar, "Modeling human aspects of business processes – a view-based, model-driven approach," in *ECMDA-FA*, ser. Lecture Notes in Computer Science, I. Schieferdecker and A. Hartman, Eds., vol. 5095. Springer, 2008, pp. 246–261.

[9] T. Holmes, U. Zdun, F. Daniel, and S. Dustdar, "Monitoring and Analyzing Service-Based Internet Systems through a Model-Aware Service Environment," in *CAiSE*, ser. Lecture Notes in Computer Science, B. Pernici, Ed., vol. 6051. Springer, Jun. 2010, pp. 98–112.

[10] T. Holmes, U. Zdun, and S. Dustdar, "MORSE: A Model-Aware Service Environment," in *Proceedings of the 4th IEEE Asia-Pacific Services Computing Conference (APSCC)*, M. Kirchberg, P. C. K. Hung, B. Carminati, C.-H. Chi, R. Kanagasabai, E. D. Valle, K.-C. Lan, and L.-J. Chen, Eds. IEEE, Dec. 2009, pp. 470–477.

[11] IEEE, "Recommended Practice for Architectural Description of Software Intensive Systems," IEEE, Tech. Rep. IEEE-std-1471-2000, 2000.

[12] M. Kögel and J. Helming, "EMFStore: a model repository for EMF models," in *ICSE (2)*, J. Kramer, J. Bishop, P. T. Devanbu, and S. Uchitel, Eds. ACM, 2010, pp. 307–308.

[13] V. Lotz, E. Pigout, P. M. Fischer, D. Kossmann, F. Massacci, and A. Pretschner, "Towards systematic achievement of compliance in service-oriented architectures: The MASTER approach," *WIRTSCHAFTSINFORMATIK*, vol. 50, no. 5, pp. 383–391, Oct. 2008. [Online]. Available: http://www.springerlink.com/content/lgr0v556845tt036/

[14] C. Mayr, U. Zdun, and S. Dustdar, "Model-Driven Integration and Management of Data Access Objects in Process-Driven SOAs," in *ServiceWave*, 2008, pp. 62–73.

[15] L. Murta, C. Corrêa, J. G. Prudêncio, and C. Werner, "Towards Odyssey-VCS 2: Improvements over a UML-based version control system," in *CVSM '08: Proceedings of the 2008 international workshop on Comparison and versioning of software models*. New York, NY, USA: ACM, 2008, pp. 25–30.

[16] K. Namiri and N. Stojanovic, "Pattern-Based Design and Validation of Business Process Compliance," in *OTM Conferences (1)*, 2007, pp. 59–76.

[17] E. Oberortner, U. Zdun, and S. Dustdar, "Tailoring a model-driven quality-of-service DSL for various stakeholders," in *MISE '09: Proceedings of the 2009 ICSE Workshop on Modeling in Software Engineering*. Vancouver, BC, Canada: IEEE Computer Society, 2009, pp. 20–25.

[18] P. Silveira, C. Rodriguez, F. Casati, F. Daniel, V. D'Andrea, C. Worledge, and Z. Taheri, "On the design of compliance governance dashboards for effective compliance and audit management," in *Proc. of the 3rd Workshop on Non-Functional Properties and SLA Management in SOC (NFPSLAM-SOC'09)*, 2009.

[19] B. Sriraman and R. Radhakrishnan, "Event driven architecture augmenting service oriented architectures," Report of Unisys and Sun Microsystems, 2005.

[20] H. Tran, T. Holmes, U. Zdun, and S. Dustdar, *Modeling Process-Driven SOAs – a View-Based Approach*, Handbook of Research on Business Process Modeling ed. IGI Global, Apr. 2009, ch. 2.

[21] H. Tran, U. Zdun, and S. Dustdar, "View-based and Model-driven Approach for Reducing the Development Complexity in Process-Driven SOA," in *Int'l Conf. Business Process and Services Computing*, ser. LNI, vol. 116. GI, 2007, pp. 105–124.

[22] ——, "View-based integration of process-driven soa models at various abstraction levels," in *1st Int'l Workshop on Model-Based Software and Data Integration*. Springer, Apr. 2008, pp. 55–66.

[23] ——, "View-based reverse engineering approach for enhancing model interoperability and reusability in process-driven SOAs," in *10th Int'l Conf. Software Reuse (ICSR)*. Springer, 2008, pp. 233–244.