

Rule Based Business Process Compliance

Guido Governatori, Sidney Shek

NICTA, Australia

Abstract. In this paper we report on the development and evaluation of a business process compliance checker, based on the compliance-by-design methodology proposed by Governatori and Sadiq [9].

For a screencast see http://www.youtube.com/watch?v=gFmDQJNai_4

1 Introduction

Regulatory compliance is the set of activities an enterprise does to ensure that its core business does not violate relevant regulations, in the jurisdictions in which the business is situated, governing the (industry) sectors where the enterprise operates.

The activities an organisation does to achieve its business objectives can be understood as business processes, and consequently they can be represented by business process models. On the other hand a normative document (e.g., a code, a bill, an act) can be understood as a set of clauses, and these clauses can be represented in an appropriate formal language. Based on this [5] proposed that *business process compliance* is a relationship between the formal representation of a process model and the formal representation of a relevant regulation.

To gain compliance different strategies can be devised. [16] classifies approaches to compliance as *detective*, *corrective* and *preventative*.

Detective measures are intended to identify “after-the-fact” un-compliant situations. There are two main approaches: (a) *retrospective reporting* through manual audits by consultants or through IT forensics and Business Intelligence tools; (b) *automated detections* generating audit reports against hard-coded checks performed on the requisite system. Unlike the first approach, automated detection reduces the assessment time and consequently also the time of un-compliance remediation/mitigation.

Corrective measures are intended to limit the extent of any consequence caused by un-compliant situations. For example, situations that can arise from the introduction of a new norm impacting upon the business, to the organisation coming under surveillance and scrutiny by a control authority or to an enforceable undertaking.

The two approaches above suffer from lack of *sustainability*, caused by the extreme interest of companies in continuous improvements of the quality of services, and for changing legislations and compliance requirements. Indeed, even with automated detection means, the hard coded checking of repositories can quickly grow to a very large scale making it extremely difficult to evolve and maintain. To obviate these problem [17, 13] propose a *preventative focus* based on the idea of *compliance-by-design*.

The key aspect of the compliance-by-design methodology is to supplement business process models with additional information to ensure that a business process is compliant with relevant normative frameworks before the deployment of the process itself.

2 BPCC Architecture

In this section we first introduce the architecture of BPCC, a business process compliance checker based on the business process compliance methodology proposed by Governatori and Sadiq [9].

As we have already discussed to check whether a business process is compliant with a relevant regulation, we need an annotated business process model and the formal representation of the regulation. The annotations are attached to the tasks of the process, and it can be used to record the data, resources and other information related to the single tasks in a process.

For the formal representation of the regulation we use FCL [4, 8]. FCL is a simple, efficient, flexible rule based logic. FCL has been obtained from the combination of defeasible logic (for the efficient and natural treatment of exceptions, which are a common feature in normative reasoning) [1] and a deontic logic of violations [6]. In FCL a norm is represented by a rule

$$a_1, \dots, a_n \Rightarrow c$$

Where a_1, \dots, a_n are the conditions of applicability of the norm/rule and c is the *normative effect* of the norm/rule. FCL distinguishes two normative effects: the first is that of introducing a definition for a new term. For example the rule

$$customer(x), spending(x) > 1000 \Rightarrow premium_customer(x)$$

specifies that, typically, a premium customer is a customer who has spent over 1000 dollars. The second normative effect is that of triggering obligations and other deontic notions. The deontic notions covered by FCL are obligations¹, permissions, and reparation chains. For obligations FCL supports both maintenance obligations and achievement obligations, and for achievement obligations both pre-emptive and non-pre-emptive obligations (see [8] for full details). A reparation chain is an expression $O_1c_1 \otimes O_2c_2 \otimes \dots \otimes O_nc_n$, where each O_i is an obligation, and each c_i is the content of the obligation (modelled by a literal). The meaning of a reparation chain is that we have that c_1 is obligatory, but if the obligation of c_1 is violated, i.e., we have $\neg c_1$, then the violation is compensated by c_2 (which is then obligatory). But if even O_2c_2 is violated, then this violation is compensated by c_3 which, after the violation of c_2 , becomes obligatory, and so on.

It is worth noticing that FCL allows deontic expression (but not reparation chains) to appear in the body of rules, thus we can have rules like:

$$restaurant, [P]sell_alcohol \Rightarrow [OM]show_license \otimes [OAPNP]pay_fine.$$

The rule above means that if a restaurant has a license to sell alcohol (i.e, it is permitted to sell it, $[P]sell_alcohol$), then it has a maintenance obligation to expose the license ($[OM]show_license$), if it does not then it has to pay the fine ($[OAPNP]pay_fine$). The obligation to pay the fine is non-pre-emptive (this means it cannot be paid before the violation). For full description of FCL and its feature see [4, 8].

¹ Note the obligations allow us to capture prohibitions; a prohibition is an obligation plus negation, for example the prohibition to smoke can be understood as the obligation not to smoke.

Finally, FCL is agnostic about the nature of the literals it uses. They can represent tasks (activities executed in a process) or propositions representing state variables.

Compliance is not just about the tasks to be executed in a process but also on what the tasks do, the way they change the data and the state of artifacts related to the process, and the resources linked to the process. Accordingly, process models must be enriched with such information. [17] proposes to enrich process models with semantic annotations. Each task in a process model can have attached to it a set of semantic annotations. In our approach the semantic annotations are literals in the language of FCL, representing the effects of the tasks. The approach can be used to model business process data compliance [10]

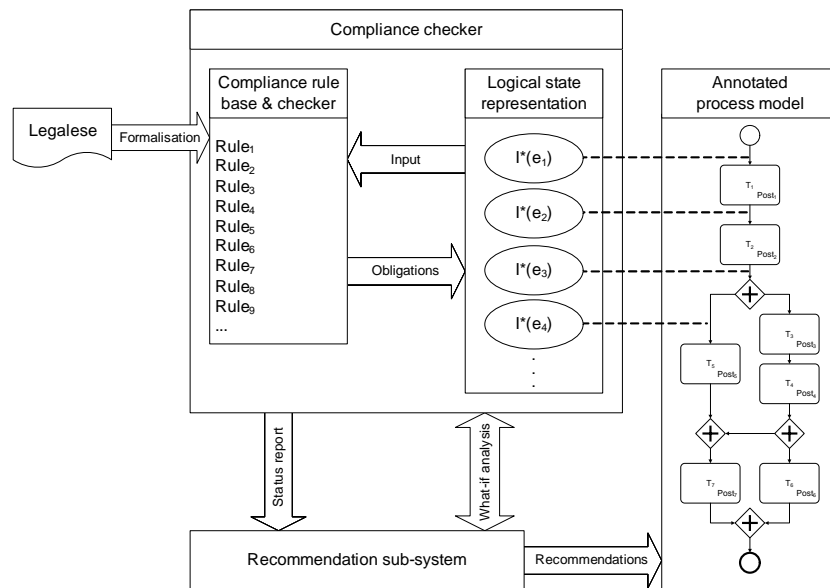


Fig. 1. Architecture of BPCC

Figure 1 depicts the architecture of BPCC. Given an annotated process and the formalisation of the relevant regulation, we can use the algorithm propose in [7, 8] to determine whether the annotated process model is compliant. The process runs as follows:

- Generate an execution trace of the process.
- Traverse the trace:
 - for each task in the trace, cumulate the effects of the task using an update semantics (i.e., if an effect in the current task conflicts with previous annotation, update using the effects of the current tasks).
 - use the set of cumulated effects to determine which obligations enter into force at the current tasks. This is done by a call to an FCL reasoner.
 - add the obligations obtained from the previous step to the set of obligations carried over from the previous task.

- determine which obligations have been fulfilled, violated, or are pending; and
 - if there are violated obligation check whether they have been compensated.
 - repeat for all traces.

A process is compliant if and only if all traces are compliant (all obligations have been fulfilled or if violated they have been compensated). A process is weakly compliant if there is at least one trace that is compliant.

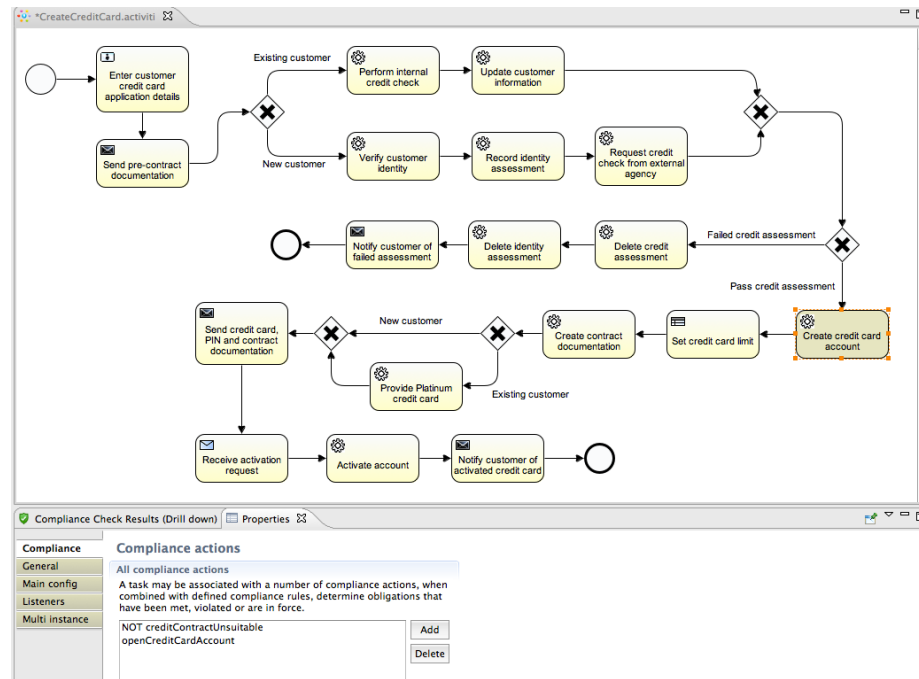


Fig. 2. An Opening Credit Card Account Process with Annotations in BPCC

3 Implementation and Evaluation

BPCC is implemented on top of Eclipse. For the representation of process models, it uses the Eclipse Activiti BPMN 2.0 plugin, extended with features to allow users to add semantic annotations to the tasks in the process model. BPCC is process model agnostic, this means that while the current implementation is based on BPMN all BPCC needs is to have a description of the process and the annotations for each task. A module of BPCC take the description of the process and generates the execution traces corresponding to the process. After the traces are generated, it implements the algorithm outlined in the previous section, where it uses the SPINdle rule engine [12] for the evaluation of the FCL rules. In case a process is not compliant (or if it is only weakly compliant) BPCC

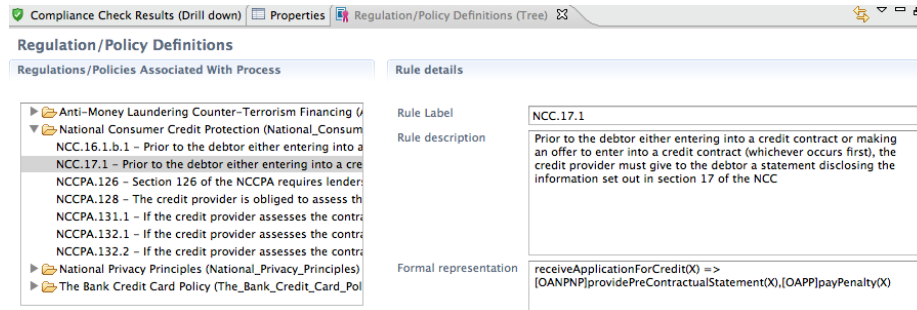


Fig. 3. Regulations Relevant to the Opening Credit Card Process

reports the traces, tasks, rules and obligations involved in the non compliance issues (see Figure 4).

BPCC was tested against an 2012 Australian Telecommunications Customers Protection Code (C628-2012). The code is effective from September 1st 2012. The code requires telecommunication operators to provide annual attestation of compliance with the code starting from April 1st 2013. The evaluation was carried out in May-June 2012. Specifically, the section of the code on complaint handling has been manually mapped to FCL. The section of the code contains approximately 100 commas, in addition to approximately 120 terms given in the Definitions and Interpretation section of the code. The mapping resulted in 176 FCL rules, containing 223 FCL (atomic) propositions, and 7 instances of the superiority relation. Of the 176 rules 33 were used to capture definitions of terms used in the remaining rules. Mapping the section of the code required all features of FCL: all types of obligations apart punctual obligations were used, reparation chains, permissions, defeasibility to easily capture exceptions, and obligations and permissions in the body of rules.

The evaluation was carried over in cooperation with an industry partner subject to the code. The industry partner did not have formalised business processes. Thus, we worked with domain experts from the industry partner (who had not been previously exposed to BPM technology, but who were familiar with the industry code) to draw process models for the activities covered by the code. The evaluation was carried out in two steps. In the first part we modelled the processes they were. BPCC was able to identify several areas where the existing processes were not compliant with the new code. In some cases the industry partner was already aware of some of the areas requiring modifications of the existing processes. However, some of the compliance issues discovered by the tools were novel to the business analysts and were identified as genuine non-compliance issues that need to be resolved. In the second part of the experiment, the existing processes were modified to comply with the code based on the issues identified in the first phase. In addition a few new business process models required by the new code were designed. As result we generated and annotated 6 process models. 5 of the 6 models are limited in size and they can be checked for compliance in seconds. The largest process contains 41 tasks, 12 decision points, xor splits, (11 binary, 1 ternary). The shortest path in the model has 6 tasks, while the longest path consists of 33 tasks (with 2 loops), and the

longest path without loop is 22 task long. The time taken to verify compliance for this process amounts approximately to 40 seconds on a MacBook Pro 2.2Ghz Intel Core i7 processor with 8GB of RAM (limited to 4GB in Eclipse).

4 Conclusions

We reported on the development of a tool, BPCC, for checking the compliance of business processes with relevant regulations. The BPCC was successfully tested for real industry scale compliance problems. In the recent years, a few other compliance prototypes have been proposed: MoBuCom [15], Compass [2] and SeaFlows [14]. MoBuCom and Compass are based on Linear Temporal Logic (LTL) and mostly they just address “structural compliance” (i.e., that the tasks are executed in the relative order defined by a constraint model). The use of LTL implies that the model on which these tools are based on is not conceptual relative to the legal domain, and it fails to capture nuances of reasoning with normative constrains such as violations, different types of obligations, violations and their compensation. For example, obligations are represented by temporal operators. This raises the problem of how to represent the distinction between achievement and maintenance obligations. A possible solution is to use always for maintenance and sometimes for achievement, but this leaves no room for the concept of permission (the permission is dual of obligation, and always and sometimes are the dual of each other). In addition using temporal operators to model obligations makes hard to capture data compliance [10], i.e., obligations that refer to literals in the same task. SeaFlow is based on first-order logic, and it is well know that first order logic is not suitable to capture normative reasoning [11]. On the other hand FCL complies with the guidelines set up in [3] for a rule languages for the representation of legal knowledge and legal reasoning.

Acknowledgment

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

References

1. Antoniou, G., Billington, D., Governatori, G., Maher, M.J.: Representation Results for De-feasible Logic. *ACM Trans. Comput. Log.* 2(2), 255–287 (2001)
2. Elgammal, A., Türetken, O., van den Heuvel, W.-J.: Using Patterns for the Analysis and Resolution of Compliance Violations. *Int. J. Cooperative Inf. Syst.* 21(1), 31–54 (2012)
3. Gordon, T.F., Governatori, G., Rotolo, A.: Rules and Norms: Requirements for Rule Inter-change Languages in the Legal Domain. In: Governatori, G., Hall, J., Paschke, A. (eds.) *RuleML 2009*pp. 282–296. Springer, Berlin (2009)
4. Governatori, G.: Representing Business Contracts in RuleML. *Int. J. Cooperative Inf. Sys.* 14(2-3), 181–216 (2005)

5. Governatori, G., Milosevic, Z., Sadiq, S.: Compliance checking between business processes and business contracts. In: Hung, P.C.K. (ed.) Pp. 221–232. IEEE Computing Society (2006)
6. Governatori, G., Rotolo, A.: Logic of Violations: A Gentzen System for Reasoning with Contrary-To-Duty Obligations. *Australasian Journal of Logic* 4, 193–215 (2006)
7. Governatori, G., Rotolo, A.: An Algorithm for Business Process Compliance. In: Francesconi, E., Sartor, G., Tiscornia, D. (eds.) *Legal Knowledge and Information Systems*pp. 186–191. IOS Press (2008)
8. Governatori, G., Rotolo, A.: A Conceptually Rich Model of Business Process Compliance. In: Link, S., Ghose, A. (eds.) *7th Asia-Pacific Conference on Conceptual Modelling (APCCM 2010)*pp. 3–12. ACS (2010)
9. Governatori, G., Sadiq, S.: “The Journey to Business Process Compliance”. In: *Handbook of Research on BPM*. Ed. by J. Cardoso and W. van der Aalst. IGI Global, 26 October 2008 2009. Chap. 20, pp. 426–454. eprint: <http://www.governatori.net/papers/2009/handbook09journey.pdf>.
10. Hashmi, M., Governatori, G., Wynn, M.T.: Business Process Data Compliance. In: *RuleML 2012* Springer (2012)
11. Herrestad, H.: Norms and formalization. In: pp. 175–184. ACM (1991)
12. Lam, H.-P., Governatori, G.: The Making of SPINdle. In: Governatori, G., Hall, J., Paschke, A. (eds.) *Rule Representation, Interchange and Reasoning on the Web*pp. 315–322. Springer, Berlin (2009)
13. Lu, R., Sadiq, S., Governatori, G.: Compliance Aware Business Process Design. In: ter Hofstede, A.H.M., Benatallah, B., Paik, H.-Y. (eds.) *3rd International Workshop on Business Process Design (BPD’07)*pp. 120–131. Springer, Berlin (2007)
14. Ly, L.T., Rinderle-Ma, S., Göser, K., Dadam, P.: On enabling integrated process compliance with semantic constraints in process management systems - Requirements, challenges, solutions. *Information Systems Frontiers* 14(2), 195–219 (2012)
15. Maggi, F., Montali, M., Westergaard, M., van der Aalst, W.: “Monitoring Business Constraints with Linear Temporal Logic: An Approach Based on Colored Automata”. In: *BPM 2011*. Vol. 6896. LNCS. Springer-Verlag, 2011, pp. 132–147.
16. Sadiq, S., Governatori, G.: “Managing Regulatory Compliance in Business Processes”. In: *Handbook of Business Process Management*. Ed. by J. van Brocke and M. Rosemann. Vol. 2. Berlin: Springer, 13 October 2010. Chap. 8, pp. 157–173. DOI: 10.1007/978-3-642-01982-1_8. eprint: <http://www.governatori.net/papers/2009/handbook09methodology.pdf>.
17. Sadiq, S., Governatori, G., Naimiri, K.: Modelling of Control Objectives for Business Process Compliance. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) *BPM 2007*pp. 149–164. Springer, Berlin (2007)