RuleOMS: A Rule-Based Online Management System

Mohammad Badiul Islam
NICTA, Queensland Research Laboratory
Brisbane, Australia
badiul.islam@nicta.com.au

Guido Governatori
NICTA, Queensland Research Laboratory
Brisbane, Australia
guido.governatori@nicta.com.au

ABSTRACT

We propose an architecture for a rule-based online management systems (RuleOMS). Typically, many domain areas face the problem that stakeholders maintain databases of their business core information and they have to take decisions or create reports according to guidelines, policies or regulations. To address this issue we propose the integration of databases, in particular relational databases, with a logic reasoner and rule engine. We argue that defeasible logic is an appropriate formalism to model rules, in particular when the rules are meant to model regulations. The resulting RuleOMS provides an efficient and flexible solution to the problem at hand using defeasible inference. A case study of an online child care management system is used to illustrate the proposed architecture.

1. INTRODUCTION

Compliance and regulations are significant components in business processes and typically, governments enforce various regulations and policies for public and private benefit. It is a common scenario that businesses and agencies have to generate reports based on such regulations (either to demonstrate compliance with the regulation, or because mandated to do so by the regulations themselves).

For example, Australia's tax and superannuation system works on the regulation set by Australian Taxation Office for the Commonwealth of Australia and a self-assessment model[1]. Using this system and model, tax payers or their nominated registered tax agent typically claims tax return or activity statement or assess, tax liability. Another example is the online *Child Care Management Systems*, which are operated based on the rules and regulations of the Department of Education for Child Care Management Systems[2] and record child enrolment and attendance information. Typically, these systems are connected with multiple government agencies, sharing information with read/write authority to the government and generate reports for various stakeholders. For instance, Child Care Management systems generate reports based on the data or provide data to the Department of Education, so, the department can calculate possible fee reductions, calculate child care rebates based on child care data and pay services on behalf of eligible families. Child Care Management Systems may also record child abuse and/or neglect data along with the child enrolment and attendance information, and in case of possible risks for abuse or neglect they have to report to appropriate social service or law enforcement agencies.

The most typical scenario today is the organizations may use various record keeping mechanisms/systems and databases, in particular relational databases, to record user and system generated data. By using these databases, the organizations may require to operate their businesses and generate reports for government and other concerned stakeholders according to the guidelines, policies and regulations. This should not come as a surprise, of course, since there is a technical as well as motivational/philosophical concern how to integrate these relational database with rule systems based on plausible assumptions.

In this paper we introduce a new system, called *Rule based Online Management System (RuleOMS)*, that facilitates the integration of (relational) databases with rule based engines meant to implement regulations. In particular, it is developed in the setting of Defeasible Logic (DL). RuleOMS allows users or agents to interact with various stakeholders and generates conclusion based on existing records stored in relational database according to defeasible logic rules encoding relevant norms. Why DL? Indeed, DL is one of the most expressive languages that allows for legal reasoning (and it has been proved that other formalisms successful in legal reasoning corresponds to variants of DL [5]). Recent research has also shown that DL is suitable for dealing incomplete and potentially contradictory information and for modelling situations where conflicting rules may appear simultaneously in real-world applications. Its use has been advocated in various application domains, such as Business rules and regulations [1], firewall verification and reconfiguration [16], agent modeling and agent negotiations [9], Semantic Web [3] and business process compliance [17, 7]. Thus, reasonably, this research has employed DL to detect, in particular, risk involving online management system.

There are many applications of RuleOMS including, but not limited to data management, decision, and domain specific analysis. In this paper we use an online *Child Care Management Systems* as an example to illustrate the RuleOMS system architecture and its internal components. In general, we present an automation system, based on the RuleOMS, using the child care data for the early identification of risk involving into the business processes oriented to Child Care Management Systems.

This paper is organised into five sections. The next section introduces Rule based Online Management System. In the third section we present the Defeasible Logic. We include a sample case

[1] http://www.ato.gov.au/General/How-we-check-compliance/Self-assessment/ accessed on 15 April 2015.

[2] http://education.gov.au/child-care-management-system accessed on 18 April 2015.

ICAIL '15, June 08–12, 2015, San Diego, CA, USA
ACM 978-1-4503-3522-5/15/06.
http://dx.doi.org/10.1145/2746090.2746120.

study: Online Child Care Management System in the fourth section. Finally, we present some review of related works and conclusions.

2. RULEOMS ARCHITECTURE

In this section, we will introduce the *RuleOMS* design architecture and technical details (Refer to Figure 1). Overall, RuleOMS consists of four main system components. In particular, the key system components of RuleOMS are as: 1) JSON (JavaScript Object Notation); 2) I/O Interface; 3) Formal Rules; and 4) SPINdle Reasoner. In the rest of this section we give a short outline of the RuleOMS internal components and their functions.

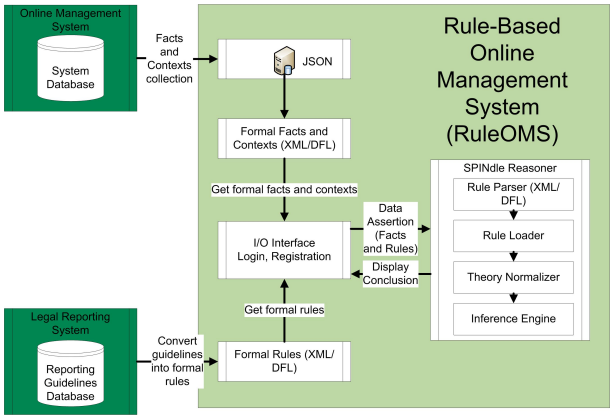

Figure 1: Rule-Based Online Management System (RuleOMS)

2.1 JSON

As one of the initial steps, we collect the facts and contexts from the relational database of the *Online management System* and convert them into lightweight data-interchange JSON (JavaScript Object Notation) syntax[3]. We use JSON syntax to create a bridge between the data stored in the database and the terms passed as facts (input case) to the rule engine. JSON is easy for humans to read and write, easy to generate, and easy to parse for machines. The JSON file contains all of the incidents along with all of the predicates for each of the incidents. Incident ID and relevant details of the incidents are also included for each of the predicates. We use SQL statements to fetch each of the incidents and predicates from the relational database and convert into JSON format using the I/O interface to make the process dynamic. In the next step the records for which there is a match in the relational database are transformed into facts and rules to be used by the SPINdle rule engine [12] and forwarded to it for further processing. The following snippet illustrates JSON syntax adopted by RuleOMS.

```
{"riskForIncident":{ "incidentID":"XXXXXX",
                     "IncidentDetails":"ABC",
                     "IncidentDetails1":null,
                     "IncidentDetails2":"XYZ"}}
```

2.2 I/O Interface

The I/O Interface is implemented using Java programming Language to bridge RulOMS system components and interacting between each other. The I/O interface is used to compile JSON file and to generate facts and contexts in formal notation in DFL syntax which are passed to SPINdle reasoner. The I/O interface also displays the final remarks or comments for each of the incidents and predicates.

[3] `http://jason.org`.

2.3 Formal Rules

One of the features of RuleOMS is its ability to perform reasoning based on (legal) requirements. As we alluded to in the introduction such regulatory requirements are represented as formal rules in Defeasible Logic [2, 15]; to enable their use with the rule engine used by RuleOMS (SPINdle, see the next section) the rules are stored in the DFL format [12]. At this stage the rules are manually created by legal knowledge engineers. For example, the formal representation for "If there is a risk for an incident then report to Chief Operating Officer (CEO)" is an Non-persistent, pre-emptive obligation [OANP] (Refer to [7, 4, 11] for other types of obligation) which can be met by actions performed before the obligation is in force, and if it is violated then the obligation is no longer in force. The requirement to "report to the CEO" if there is risk for an incident is represented by the following formal representation (in DFL SPINdle syntax): $riskForIncident \Rightarrow$ [OANP]$reportToCEO$.

2.4 SPINdle Reasoner

SPINdle Reasoner [12] is a Java based implementation of DL [2, 15] that computes the extension of a defeasible theory (see Section 3 below). SPINdle supports Modal DL, all types of DL rule, such as facts, strict rules, defeasible rules, defeaters and superiority. Due to the limited space, interested readers are referred to [12] for details. In summary, SPINdle is a powerful tool which accepts rules, facts, monotonic and non-monotonic (modal) rules for reasoning with inconsistent and incomplete information. In RuleOMS, SPINdle Reasoner receives the formal facts, contexts and predicates from JSON file generated for data stored in the associated relational databases and computes definite or defeasible inferences which are then displayed by the I/O interface.

3. DEFEASIBLE LOGIC

Defeasible Logic was originally proposed by Donald Nute [15]. Since then it has been significantly used in the legal domain or closely related areas, such as modelling regulations [1], e-contracting [6, 10], business processes compliance [17, 7] and automatic negotiation system [18]. Logic Modelling also supports the analysis of regulations through "Decision support", "Explanation", "Anomaly detection", "Hypothetical reasoning" and "Debugging". Decision support is used to infer a correct answer from given rules and regulations. DL is one of the possible solutions since regulations may contradict one another using the defeasible rules which do not necessarily fire; instead they may be blocked by other rules with contrary conclusions [1].

3.1 Basics of Defeasible Logic

A *defeasible theory* D is a triple $(F, R, >)$ where F and R are finite sets of facts and rules respectively, and $>$ is a superiority relation on R. Here SPINdle only considers rules that are essentially propositional. Rules containing free variables are interpreted as the set of their ground instances.

Facts are indisputable statements, represented either in form of states of affairs (literal and modal literal) or actions that have been performed. Facts are represented by predicates. For example, the fact "John is a human" is represented by $human(John)$.

A *rule*, on the other hand, describes the relation between a set of literals (premises) and a literal (conclusion). We can specify the strength of the rule relation using the three kinds of rules supported by DL, namely: *strict rules*, *defeasible rules* and *defeaters*; in addition, we can specify the mode the rules use to connect the antecedent and the conclusion. However, in such situations, the conclusions derived will be *modal literals*. *Strict rules* are rules in the classical sense: whenever the premises are indisputable (e.g.

facts) then so is the conclusion. An example of a strict rule is: "human are mammal", written formally: $human(X) \rightarrow mammal(X)$. *Defeasible rules* are rules that can be defeated by contrary evidence. An example of such a rule is "mammal cannot flies"; written formally:$mammal(X) \Rightarrow \neg flies(X)$. The idea is that if we know that X is a mammal then we may conclude that it cannot fly *unless there is other, not inferior, evidence suggesting that it may fly* (for example that mammal is a bat).

DL is a "skeptical" nonmonotonic logic, meaning that it does not support contradictory conclusions. Instead DL seeks to resolve conflicts. In cases where there is some support for concluding A but also support for concluding $\neg A$, DL does not conclude neither of them. However, if the support for A has priority over the support for $\neg A$ then A is concluded.

As we have alluded to above, no conclusion can be drawn from conflicting rules in DL unless these rules are prioritised. The *superiority relation* is used to define priorities among rules, that is, where one rule may override the conclusion of another rule. For example, given facts: $\rightarrow bird$ and $\rightarrow brokenWing$ and the defeasible rules: r: $bird \Rightarrow fly$ and r': $brokenWing \Rightarrow \neg fly$ which contradict one another, no conclusion can be made about whether a bird with a broken wing can fly. But if we introduce a superiority relation $>$ with $r' > r$, then we can indeed conclude that the bird cannot fly.

We now give a short informal presentation of how conclusions are drawn in DL. Let D be a theory in DL (as described above). A *conclusion* of D is a tagged literal and can have one of the following four forms: $+\Delta q$ meaning that q is definitely provable in D (i.e. using only facts and strict rules); $-\Delta q$ meaning that we have proved that q is not definitely provable in D; $+\partial q$ meaning that q is defeasible provable in D; and $-\partial q$ meaning that we have proved that q is not defeasible provable in D.

Strict derivations are obtained by forward chaining of strict rules while a defeasible conclusion p can be derived if there is a rule whose conclusion is p, whose prerequisites (antecedent) have either already been proved or given in the case at hand (i.e. facts), and any stronger rule whose conclusion is $\neg p$ has prerequisites that fail to be derived. In other words, a conclusion p is (defeasibly) derivable when: p is a fact; or there is an applicable strict or defeasible rule for p, and either all the rules for $\neg p$ are discarded (i.e. not applicable) or every rule for $\neg p$ is weaker than an applicable rule for p.

A full definition of the proof theory can be found in [2]. Roughly, the rules with conclusion p form a team that competes with the team consisting of the rules with conclusion $\neg p$. If the former team wins p is defeasibly provable, whereas if the opposing team wins, p is non-provable.

3.2 Modal Defeasible Logic

Modal logics have been put forward to capture many various notions somehow related to the intensional nature of agency as well as many other notions. Usually modal logics are extensions of classical propositional logic with some intentional operators. Thus any modal logic should account for two components: (1) the underlying logical structure of the propositional base and (2) the logical behavior of the modal operators. Alas, as is well-known, classical propositional logic is not well suited to deal with real life scenarios. The main reason is that the descriptions of real-life cases are, very often, partial and somewhat unreliable. In such circumstances, classical propositional logic is doomed to suffer from the same problems.

To this end, RuleOMS follows the semantics proposed by [8] on reasoning with modal defeasible logic. Thus for example, the rule[4] *riskForIncident* $\Rightarrow$ [OANP]*reportToCEO* specifies that the mode of the rule is OANP; this means that this rule produces a conclusion that is an achievement non-preemptive obligation, when the antecedent of the rule holds.

Due to space reasons, readers interested in understand the semantics, modal operator conversions, conflict detections, conflict resolutions, and algorithm implemented in this rule based system are referred to [9, 8, 7] for details.

4. CASE STUDY: ONLINE CHILD CARE MANAGEMENT SYSTEM

In this section we illustrate RuleOMS with the help of a case study to demonstrate possible applications of the RuleOMS architecture. Accordingly, we introduce the **ChildSafe Care Online Management Systems (ChildSafeOMS)** and its relational database **ChildSafeDB**. ChildSafeOMS maintains a relational database to record child enrolment, attendance, abuse and/or neglect data and relevant system information.

Through ChildSafeOMS, child care providers are authorised by the Government to access various types of information on behalf of children's families. Typically, child care providers have to comply with Government's regulations and legislation for reporting to respective authorities such as New South Wales Mandatory Reporter Guidelines [13] on matter of child abuse and neglect. The mandatory reporting guidelines also includes decision trees and various types of reporting authorities and their accountability to report or consult with other relevant authorities (i.e., "immediate report to Community Service (CS)", "report to CS" and "report to Child Wellbeing Unit (CWU)").

The integration of RuleOMS and the underlying ChildSafeDB extend the ChilSafeOMS's functionality. In particular, the ChildSafeOMS introduces new functionality including the automation of early identification of children at risk of abuse and/or neglect. This system is developed by applying a rule reasoner, SPINdle [12] on synthetic and sample relational database created for the ChildSafeOMS.

More specifically, the system contains three sources of information: 1) a child care database ChildSafeOMS with the data described above; 2) a set of defeasible rules encoding the decision trees, definitions and (normative) guidelines of "New South Wales Mandatory Reporter Guidelines" [13]. These rules have been created by manual translation from the source document; and 3) a set of bridging statements relating the terms in the rules and the fields (and data) in the ChildSafeDB. These statements are obtained from cross-analysis of the terms and fields in the two related components.

One of the aim of the system is the early identification of children at risk, in particular risk of abuse and/or neglect. In the rest of the section we describe the component outlined above.

4.1 ChildSafe Database

ChildSafeDB contains a number of tables with information about children in a child care centre, types of abuses and their descriptions, and records for possible instances of abuses. The typical work-flow to populate the information is that educators of child care insert data about child in the database to report day-to-day activities. In case they have concerns for some forms of abuse of neglect, the system offers an interactive questionnaires where the educator answers a set of 'Health and Welfare' emotional and physical questions of the type ***"In your knowledge, do any of the following environments apply***

[4]The DFT syntax used by SPINdle is plain text based, thus, the rule is represented as `riskForIncident => [OANP] reportToCEO`. While the syntax explicitly attaches the modal operator `[OANP]` to the conclusion of the rule, it should be understood that the modality refers to the rule itself.

to the child?". Here the questions pertains to any of the abuse types and ask if they were aware about any child was abused or if they have any concern about a particular child. The systems populates the database according to answers provided by the educators/carers. Additionally, the educator/child carer also fills up five additional information along with the abuses types: i) Child Abuse ID (auto-incremental); ii) Child crn (Child Reference Number); iii) Abuse ID; iv) Abuse type id (Type of child abuse); and Abuse date.

4.2 From Database Records to Facts

As we discussed above to aim of ChildSafeOMS is to draw inferences form a database supplemented by a rule systems encoding legal knowledge. In this setting the information stored in the database is queried to provide the facts for the SPINdle reasoner. After the facts are queried from the ChildSafeDB the SPINdle reasoner draws a set of conclusions using the facts and the defeasible rules encoding the decision trees of the NSW mandatory reporter guidelines and a set of additional rules, eventually provided by domain experts, giving further indicators for risks of neglect or abuse.

For the extraction of facts RuleOMS is equipped with a JSON file containing the definitions of the predicates used by the SPINdle component in terms of either simple attribute names and values or by SQL queries. For example, according to the guideline a person is considered a child if her/his age is less than fifteen years. Several steps in the decision trees depend on whether a person at risk of abuse is a child or not. This is represented by the predicate *beingChild*. ChildSafeOMS contains the following (JSON) definition:

```
{"predicate":"beingChild",
 "SQL":"SELECT * FROM tblchildren WHERE
     tblchildren.child_crn = $id AND
     tblchildren.child_dob >= ${today - 15y}"}
```

where `$id` is a parameter given as input by the end-user, and `${today - 15y}` is a computation using the system variable `$today`.

The I/O interface takes the user input, examines the definitions, transforms the definitions into appropriate SQL queries, and executes the queries to return facts and context for the queries with no empty output. For example, for the predicate *beingChild* and `$id=123456789A` ChildsafeOMS produces the following JSON snippet, encoding the predicate and the context that warrants that the predicate holds.

```
{"beingChild":{ "tblchildren.child_crn":"123456789A",
  "tblchildren.child_first_name":"Simone",
  "tblchildren.child_middle_name":"Ruby,
  "tblchildren.child_last_name":"Shirazi",
  "tblchildren.child_dob":"2013-04-02",
  "tblchildren.child_gender":"Female"}}
```

The predicate is then used in the successive reasoning phase, while the context is used in case of reasoning phase determines that a report has to be submitted to a relevant authority; the context is used to populate the required reports.

4.2.1 Formal Rules

This section includes some sample rules in SPINdle format implementing the NSW mandatory reporter guidelines [13]. The guidelines are formulated as decision trees, and they specify (i) when an educator of child carer has to report an abuse or potential abuse (ii) under what conditions one has to report. In addition it contains definitions of types of abuse or potential abuses and indicators of abuse or potential abuse.

Consider for example the provision (as part of a decision tree) prescribing to report immediately to *CS* if there is a situation where you are aware that a child has been sexually abused even though he/she hasn't told you or the child made a clear, unambiguous statement of sexual abuse [13, pp. 17–18]. We can represent the rule as *If "ChildAbusedSexually" has occurred, then there is an obligation (non-persistent, pre-emptive, achievement) to do "reportToCSImmediately"* (Refer to [7, 4, 11] for other types of obligations). Its formal representation in SPINdle is as follows:

$$\mathit{ChildAbusedSexually} \Rightarrow [\text{OANP}]\mathit{reportToCSImmediately}$$

Furthermore, the rule set contains more rules obtained from the definitions of what counts as a child being sexually abused and possible indicators for different types of abuse. The next example concerns the situation where a child or young person exhibits a (persistent) problematic sexual behaviour versus other. In this case an educator has different options to whom report. In case the parent of carers are aware of the problem and they have responded appropriately, the educator has to report to the Child Well fare Unit (*consultWithCWU*). However, if the initiating child/young person has continuing or imminent contact with the victim and there where some coercion or the victim is in a situation of inferiority, then the situation has to be reported immediately to CS. Otherwise, the normal procedure is to file a normal report to CS [13, p. 19]. In case of a problematic behaviour without further conditions the educator has to continue to monitor the situation. This part of the decision tree can be represented by the following rules:

$$r_1 : \mathit{PersistentSexualBehaviourVsOther}, \mathit{ParentRespondedAppropriately}, \neg[\text{OANP}]\mathit{reportCS}, \neg[\text{OANP}]\mathit{reportCSImmediately} \Rightarrow [\text{OANP}]\mathit{consultWithCWU}.$$

The rule above describe the decision when the parents or carers are aware of the situation and responded appropriately. However, to ensure a single outcome (behaviour) we have to derive the failure of other obligations, namely report to CS or report to CS immediately.

$$r_2 : \mathit{SexualBehaviourVsOther}, \mathit{CoercionOrInferior}, \mathit{ContactWithVictim} \Rightarrow [\text{OANP}]\mathit{reportToCSImmediately}$$

This rule is the most specific and its antecedent subsumes all other antecedents thus there is no need to include the failure of the other obligations.

$$r_3 : \mathit{SexualBehaviourVsOther}, \mathit{CoercionOrInferior}, \neg[\text{OANP}]\mathit{reportToCSImmediately} \Rightarrow [\text{OANP}]\mathit{reportToCS}$$

This is the second most specific rule, thus its antecedent has to include the failure of the rule that is more specific that it, that is the obligation of the conclusion of r_2.

Finally, the following rule, whose conclusion is a maintenance obligation, is triggered when the other obligations do not apply.

$$r_4 : \mathit{SexualBehaviourVsOther}, \neg[\text{OANP}]\mathit{reportCSImmediately}, \neg[\text{OANP}]\mathit{consultCWU}, \neg[\text{OANP}]\mathit{reportCS} \Rightarrow [\text{OM}]\mathit{monitor}$$

5. RELATED WORK

The architecture of RuleOMS is inspired by the PLIS+ application which is a rule-based Personalized Location Information System [19]. Similar to PLIS+, RuleOMS collects and evaluates rules on-the-fly and delivers personalized and contextualized information/inferences according to rule-based policies.

Indeed, another important issue of the RuleOMS architecture is to search and extract relevant information from a relational data base and represent it as defeasible facts. RuleOMS employs JSON syntax

for representing facts and contextualized information as intermediate step to bridge relational database information with a rule reasoner.

In this paper, we also argued that DL is suitable for taking decisions based on regulations, policies and relational database information. Recent research has shown that DL is suitable for modeling and inferring from evidence and reasoning in real-world applications where conflicting situation may appear simultaneously. In the introduction section, we included some of these applications.

6. CONCLUSION

Our preliminary step was to describe an ongoing work on integration of existing *Online Management system* with compliance and regulations, which are combined with Defeasible Logic and SPINdle rule reasoner. In particular, we extend the SPINdle rule reasoner to generate reports for possible consequence of the risk involving business processes. Classically, compliance and regulations perform a significant roles in society and business processes have to comply with such policies to avoid future risks. It turned out that this type of motivations are defeasibly captured by a rule-based approach to static decision trees for Online Mandatory Reporter Guide [14].

As a next step, a rule-based architecture, RuleOMS has been introduced. It collects information from relational database of an online management system and represents the information into an intermediate file format using JSON syntax which is further processed by SPINdle rule reasoner.

Thirdly, we demonstrated that external systems such as legal reporting can be integrated with the rule reasoner, SPINdle. Typically, a legal reporting system contains compliance and regulatory requirements which were converted into a rule knowledge-base. We used a pre-defined DL format which can also be represented using predefined plain text or XML file format. We argued that using this technique external stakeholders are can also be connected with the RuleOMS.

Finally, on account of motivations of the RuleOMS, we demonstrated a case study based on an Online Child Care Management, ChildSafeOMS to automate the early identification of children at risk, in particular of abuse and/or neglect. Preliminary results appear to be very promising, showing a close correspondence between compliance, regulations and online management system. Last but not least, RuleOMS allows users or agents to produce conclusions and consequences automatically, based on a given compliance and regulatory knowledge base or generated on the fly by other applications and existing relational database information of an online management system.

ACKNOWLEDGEMENTS

NICTA is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program.

References

[1] G. Antoniou, D. Billington, G. Governatori, and M. J. Maher. On the modeling and analysis of regulations. In *Australian Conference on Information Systems*, 1999.

[2] G. Antoniou, D. Billington, G. Governatori, and M. J. Maher. Representation results for defeasible logic. *ACM Transactions on Computational Logic*, 22:255–287, 2001.

[3] N. Bassiliades, G. Antoniou, and I. Vlahavas. A defeasible logic reasoner for the semantic web. *International Journal on Semantic Web and Information Systems*, 21:1–41, 2006.

[4] G. Governatori. Business Process Compliance: An Abstract Normative Framework. *IT – Information Technology*, 556:231–238, 2013.

[5] G. Governatori. On the relationship between Carneades and defeasible logic. In *Proceedings of the 13th International Conference on Artificial Intelligence and Law (ICAIL 2011)*, pp. 31–40. ACM, 2011.

[6] G. Governatori. Representing Business Contracts in RuleML. *International Journal of Cooperative Information Systems*, 142-3:181–216, 2005.

[7] G. Governatori and A. Rotolo. A Conceptually Rich Model of Business Process Compliance. In *7th Asia-Pacific Conference on Conceptual Modelling (APCCM 2010)*. CRPIT 110, pp. 3–12. ACS, 2010.

[8] G. Governatori and A. Rotolo. BIO logical agents: Norms, beliefs, intentions in defeasible logic. *Autonomous Agents and Multi-Agent Systems*, 171:36–69, 2008.

[9] G. Governatori and A. Rotolo. Defeasible logic: Agency, intention and obligation. In *Deontic logic in computer science (DEON 2004)*. LNCS 3065, pp. 114–128. Springer, 2004.

[10] B. N. Grosof. Representing e-commerce rules via situated courteous logic programs in RuleML. *Electronic Commerce Research and Applications*, 31:2–20, 2004.

[11] M. Hashmi, G. Governatori, and M. T. Wynn. Normative Requirements for Regulatory Compliance: An Abstract Formal Framework. *Information Systems Frontiers*, 2015. DOI: 10.1007/s10796-015-9558-1.

[12] H.-P. Lam and G. Governatori. The making of SPINdle. In *International Symposium on Rule Interchange and Applications (RuleML 2009)*. LNCS 5858, pp. 315–322. Springer, 2009.

[13] National Council on Crime and Delinquency. The NSW Mandatory Reporter Guide. 2006. URL: http://www.community.nsw.gov.au/kts/guidelines/reporting/mrg2.htm.

[14] NSW Government. Online Mandatory Reporter Guide. Accessed on: 2014.09.30. 2013. URL: http://sdm.community.nsw.gov.au/mrg/screen/DoCS/en-GB/summary?user=guest.

[15] D. Nute. Defeasible logic. In D. M. Gabbay, C. H. Hogger, and J. Robinson, editors, *Handbook of logic in artificial intelligence and logic programming*. Vol. 3, pp. 353–395. Oxford University Press, 1994.

[16] P. Rajkhowa, S. M. Hazarika, and G. R. Simari. An Application of Defeasible Logic Programming for Firewall Verification and Reconfiguration. In *Quality, Reliability, Security and Robustness in Heterogeneous Networks (QShine 2013)*, pp. 527–542. Springer, 2013.

[17] S. Sadiq and G. Governatori. Managing Regulatory Compliance in Business Processes. In J. vom Brocke and M. Rosemann, editors, *Handbook of Business Process Management 2nd edition*. Vol. 2, pp. 265–288. Springer, 2nd ed., 2015.

[18] T. Skylogiannis, G. Antoniou, N. Bassiliades, G. Governatori, and A. Bikakis. DR-NEGOTIATE— A System for Automated Agent Negotiation with Defeasible Logic-Based Strategies. *Data & Knowledge Engineering*, 63:362–380, 2007.

[19] I. Viktoratos, A. Tsadiras, and N. Bassiliades. PLIS+: A Rule-Based Personalized Location Information System. In *Proceedings of the RuleML2012@ECAI Challenge*. CEUR Workshop Proceedings 874, 2012.