A Case Study on Modeling and Validating Financial Regulations Using (Semi-) Automated Compliance Framework

Suman Roychoudhury(✉), Sagar Sunkle, Namrata Choudhary, Deepali Kholkar, and Vinay Kulkarni

Tata Consultancy Services Research, 54B Hadapsar Industrial Estate, Pune, India
{suman.roychoudhury, sagar.sunkle, namrata.choudhary, deepali.kholkar, vinay.kulkarni}@tcs.com

Abstract. Modern enterprises operate in an unprecedented regulatory environment where increasing regulation and heavy penalties on non-compliance have placed regulatory compliance among the topmost concerns of enterprises worldwide. Previous research in the field of compliance has established that the manual specification/tagging of the regulations not only fails to ensure their proper coverage but also negatively affects the turnaround time both in proving and maintaining the compliance. Our contribution in this paper is a case study using a subset of European Union Regulation in the financial markets, namely, Money Market Statistical Reporting (MMSR) and that we validated it in the context of our model-driven semi-automated compliance framework. The novelty of the framework is the key participation of domain experts to author regulatory rules in a controlled natural language to enable compliance checking. We demonstrate transformation of regulations present in legal natural language text (English) to a model form via authoring of Structured English rules in the context of MMSR regulations for a large European bank. This generated regulatory model is eventually translated to formal logic that enables formal compliance checking contrary to current industry practice, that provides content management-based, document-driven and expert-dependent ways of managing regulatory compliance.

Keywords: Regulatory compliance · Money Market Statistical Reporting Structured English · Compliance checking

1 Introduction

Modern Enterprises are regulated by stricter norms and regulations that are often present in the form of legal documents, compliance process descriptions and audit reports. A major concern for enterprises arise due to heavy penalties imposed on them due to non-compliance. This has compelled enterprises to place regulatory compliance as a top priority among other challenges. Therefore, to avoid unnecessary penalties and remain compliant with respect to newer regulations, enterprises are increasingly looking towards technologies that may assist them in their overall compliance checking process.

As legal documents captures a major chunk of the regulations that should be processed to facilitate compliance checking, enterprises spent considerable effort to

R. A. Buchmann et al. (Eds.): PoEM 2018, LNBIP 335, pp. 288–302, 2018.
https://doi.org/10.1007/978-3-030-02302-7_18

decipher such enormous volume of legal text, and subsequently build software systems that can assist them in compliance checking. However, the cost of building such systems from scratch is both effort intensive and time consuming. Typical solutions that are prevalent in the industry, known as the governance, risk, and compliance (GRC) offerings, rely on taxonomies, which are collection of predefined tags that can be affixed to data [1] pertinent to the regulations. Taxonomy tagging tools used separately or from within the GRC frameworks, enables auto-population of, and in some cases, user definition of taxonomies [6, 9, 14, 15]. However, GRC based offerings do not provide support for formal compliance checking [20].

On the contrary, significant research literature by academia focuses on checking compliance of business processes and/or enterprise data using a formal specification of the regulatory rules [3, 7, 10]. Such formal representation and subsequent checking of legal rules offers significant merit over existing GRC based frameworks [8]. Therefore our approach towards automated regulatory compliance checking emphasizes on using formal methods and rules specified in formal languages like DR-Prolog [8, 22] and/or DROOLs [13]. However, formal languages have their own drawbacks and it is almost impossible for legal or domain experts to write rules using low-level logic based languages. Therefore, a high-level representation of such rules in a domain-specific language is more desirable, where participation of domain experts is of paramount interest [16]. Therefore, we provide a high-level controlled natural language (CNL) as an abstraction layer on top of the formal specifications to hide the underlying complexities and provide a business friendly English like notation to express regulations. This language is adapted from Structured English [16] and compliant to OMG's Semantic of Business Vocabulary and Rules (SBVR) [12]. However, legal text of regulations that exists in plain English is subject to scrutiny and interpretation by legal/domain experts and may require significant effort on their part to extract out the applicable legal clauses from large volume of English language sentences. Therefore to aid domain experts in authoring regulatory rules using our CNL seamlessly, we provide a machine-learning/natural-language processing (ML-NLP) based front-end engine that extracts the domain model and dictionary (i.e., various terms and concepts) from the regulatory text and provide suggestions to domain experts in their authoring process [17].

Figure 1 motivates the above hypothesis and describes our end-to-end semi-automated compliance framework that has specific human touchpoints (i.e., manual intervention) with tool support (M+T) at certain parts, while others being fully automated (T). The framework assumes precise interpretation of regulations already available with enterprise in the form of natural language (NL) legal text. Using machine-learning/distributional semantics techniques a domain model (refer to number [1] in Fig. 1) is first obtained by processing the given text with active participation by the domain expert [19]. The domain model primarily captures the keys concepts, relations and their mentions (i.e., ontology) in the given domain and serves as a core artefact for model authoring. For model authoring, the domain expert expresses the desired regulations in a controlled natural language (refer to number [2] in Fig. 1) using the domain model/dictionary and rule suggestions originating from the ML-NLP engine [16, 17]. In our case, this language was built from scratch using the XTexT language engineering workbench [4] and adapted from OMG's SBVR Structured English (SE) specification [16]. Once regulatory rules are authored in SE, a model of the regulation in SBVR is

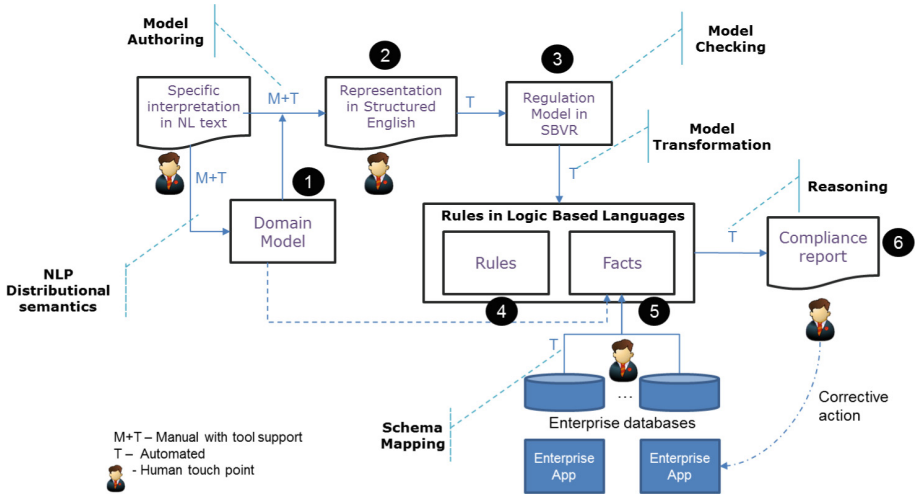

Fig. 1. Overview of automated compliance framework

automatically generated (refer to number [3] in Fig. 1) that serves as an intermediate representation for translating to low-level logical specifications (e.g., DROOLs, refer to number [4] in Fig. 1). The SBVR model is also used for obtaining suitable data facts (refer to number [5] in Fig. 1) from the enterprise databases (DBs), necessary for compliance checking [8]. A DB expert maps the suitable data model obtained from SBVR with the database schema already available with the enterprise. Finally, rules are applied to the populated fact base to generate a compliance report (refer to number [6] in Fig. 1) amenable for human understanding and suitable corrective action, if any.

The remaining parts of the paper is organized as follows. Section II of the paper describes our case study in detail with respect to a regulation from the financial domain, while section III discusses the results from the case study. Section III also provides valuable feedback and comparison of our framework with prevalent industry practice, before we conclude in section IV.

2 Case Study – Financial Regulations

In this section we introduce Money Market Statistical Reporting (MMSR) as our base case and describe in detail each of steps as outlined in Fig. 1. MMSR is a reporting regulation in the European Union involving the money markets and is regulated by the European Central Bank (ECB). All financial institutions, namely banks are mandated to report their daily transactions to ECB as prescribed by the MMSR regulatory document [11]. A leading European bank (one of our customers) was interested in validating our framework using MMSR, as test data[1] for MMSR was already available. In addition, the regulation was well understood by the bank, hence validation would be precise.

[1] Sample data is currently used due to GDPR restrictions.

Therefore, it was mutually agreed to do a pilot case study using MMSR as a litmus test of our framework.

A MMSR regulation typically consists of four different sections pertinent to money market, namely, secured market, unsecured market, FX swaps and overnight index swaps [11]. The structure and nature of regulations in all the four sections are similar, therefore modeling and validating one section will give a fairly good idea about validating other sections as well. For our case study, we chose secured market (Sect. 3) of MMSR, which captures the conceptual and field definitions of various variables that must be reported by individual banks to ECB. Overall, there are 24 such variables defined in Sect. 3 of MMSR [11], with each variable having their own conceptual definition and field definition. Conceptual definition pertains to the underlying semantics of the variable, while field definition pertains to how the variable should be constructed structurally. With this background, we would demonstrate how our auto-compliance framework can be used to model and validate regulations referring to secured market by traversing each of the six steps (numbered 1–6) as described in Fig. 1. We begin with Step 1, domain model construction.

2.1 Domain Model Construction

This is a human assisted step with tool support [19] (refer to number [1] in Fig. 1). Figure 2 shows a snapshot of the domain model generator (DMG). The input to the DMG is the set of MMSR regulations as defined in natural language text. Here, the toolset helps the domain expert to unearth the important concepts, relationships and their mentions in the underlying text. The domain expert provides the seed concept, i.e., entity types and relations immediately available from the definitions section of the legal NL text. The DMG uses two techniques to retrieve other mentions of seed entity types as well as relations between all entity types [21].

The first technique that the generator uses is based on context-based clustering. The idea behind this technique is that the contexts, i.e., spans of texts, around the mentions of various domain entities (e.g., *central bank, maturity date, transaction status, currency*) are important and could be clustered to extract useful information from the text. We cluster the contexts, i.e., **n** characters to the left and right of mentions of each entity type so far known and then cluster these to suggest to the domain expert, what looks like other possible mentions [21].

The second technique that the generator uses is based on open information extraction to discover relations between the known entity types. We input Ollie with sentences and extract relations. We try to match mentions of known entity types in the subject and object of each relation (see top half of Fig. 2). The dictionary is automatically build as shown in the bottom half of Fig. 2, while the domain model captures the complete set of concepts and their relationship (RHS of Fig. 2).For MMSR Sect. 3, the DMG captured 55 core concepts, 201 mentions (i.e., 4 mention on an average for every concept) and 49 relations. Thus, the domain model serves as an ontology for the domain expert and aids her to author rules in Structured English as described in the following step.

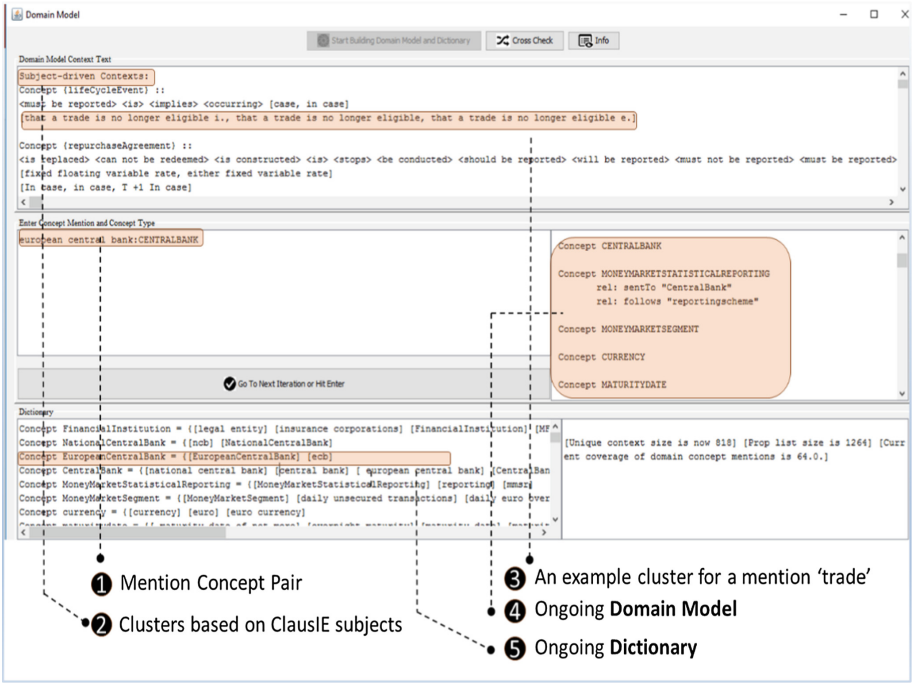

Fig. 2. Domain model generation

2.2 Rule Authoring Phase

This is a human assisted step with appropriate language support [16] (refer to number [2] in Fig. 1). In this phase the domain expert authors regulations using one form of Controlled Natural Language (CNL), which is called Structured English (SE) whose initial description appeared in OMG SBVR specification (Appendix C) [12]. The main motive behind OMG SE is to provide a business vocabulary for capturing business rules in simplified natural English in textual format. We had to adapt the language to remove some of the ambiguities from SE whose detail description is available in [16]. SE contains concepts like *general terms*, *individual terms*, *facts*, *verbs*, *quantification*, *modality*, *quantifiers* and *rules*. The English like semantics of SE makes it amenable for domain experts to specify rules at a level of abstraction appropriate to them (described in Fig. 3).

The SE editor is divided into 4 parts – *vocabulary editor* for capturing concepts, definitions, synonyms etc., the *fact editor* for relating terms or concepts in the underlying domain using verbs [16]. We use two kinds of verbs - *object* verbs for relating binary terms and *data* verbs for relating unary terms (also known as characteristics). Finally, facts are expanded by adding *implications*, *modality*, *quantification* and *qualification* to form rules. Rules are the final product that domain experts author in the *rules editor* from their natural language representations (see Fig. 3). Rules can contain any number of valid facts that are checked and validated by the error handler.

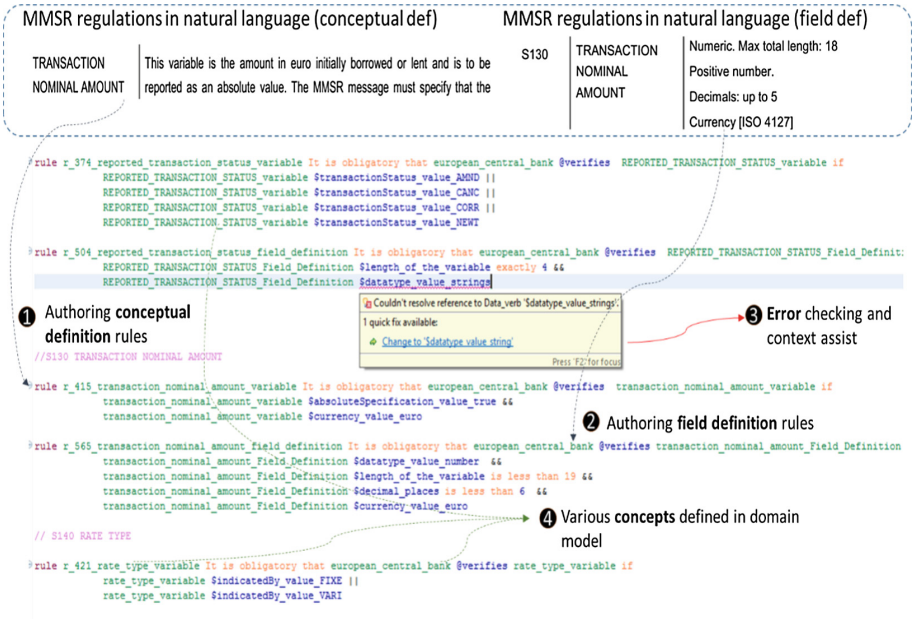

Fig. 3. Rule authoring phase

For our MMSR case study, 48 rules in Structured English were authored for each of the 24 variables belonging to either of the two categories from secured market segment, namely conceptual definition and field definition. A snapshot of the authored rules is shown in Fig. 3. The NL text of regulations for a given variable (e.g., *transaction nominal amount*) as present in MMSR specification [11] is shown the top half of the figure. The SE *rule editor* shows how the given NL text is correspondingly authored (see rule *r_415*, etc.) by the domain expert using the domain model already obtained in step 1. The SE rules are self- explanatory and easy to comprehend and author.

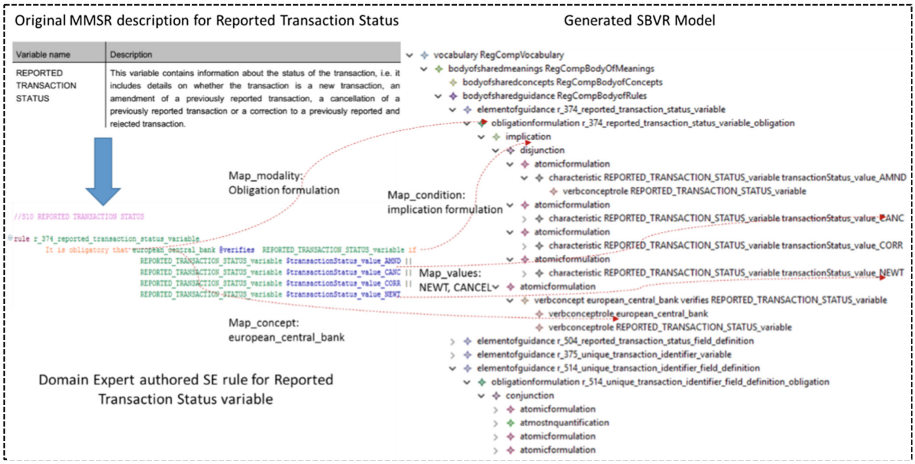

Fig. 4. SBVR model generation

Typically, rules are conjuncted and/or disjuncted by facts in an implication (conditional) statement and contains an antecedent and a consequent. Rules could be further quantified using quantifiers (e.g., *is less than 6*) as shown in Fig. 3. The SE language editor also supports error handling capabilities in the form of syntactic and/or consistency checks that may occur during the authoring phase and provides assistance for context-assist as shown in Fig. 3.

2.3 SBVR Model Generation

This is a complete automated step (refer to number [3] in Fig. 1). As domain experts author regulatory rules in Structured English, a corresponding SBVR model of the textual representation is generated in the background. This model is an instance of SBVR metamodel as given by OMG specification [12, 17]. This intermediate regulatory model is language and platform independent and can be translated to any target formal language (i.e., not tied to a particular one), which provides the execution platform. Further, domain experts are completely oblivious of this intermediate step, which is primarily used for various purposes, like consistency/model checking and also translating SE to DROOLs and/or DR-Prolog executable specifications. We chose SBVR as our desired choice of intermediate modelling language, specifically because SBVR is an "industry" standard

Fig. 5. DROOLs rule generation and population of POJOs

and comes with rich semantics for capturing business vocabularies and rules [11], which is ideally suited for capturing the semantics and vocabulary of regulations.

Figure 4 describes how this translation to SBVR model is realized from SE. The figure shows the original regulation information about a particular variable (i.e., *reported transaction status*) in its natural language form as present in MMSR specification [11]. Corresponding authored rule in SE is shown in bottom right of the figure. Generated SBVR model (snippet) is shown in the RHS of the figure and the dotted arrows depicts how the mapping from SE to SBVR is realized in the translator. For example, *modality* in SE is realized as *obligationformulation* in SBVR, *concepts* are realized as *verbconceptrole*, condition ("if" statement) is mapped to *implication, data verbs* are mapped to *characteristics*, and their *values* are mapped to *characteristic values*. Creating this SBVR model by hand is an enormous effort and statistics revealed for 48 rules authored for 24 secured market segment variables in SE, 1076 model elements were generated. This again proves the usefulness of SE, as domain experts can work at a higher level of abstraction instead of being knowledgeable about the low level details of underlying SBVR model, which is primarily required for translation to executable logical specification, inter-operability with other SBVR based tools and internal consistency or model checking of the authored rules etc.

2.4 Rules and Fact Generation

This is a fully automated step, except for schema mapping which is human assisted (see steps 4 and 5 in Fig. 1). The generated SBVR model serves as an intermediary representation that is translated to low-level logical specifications for reasoning by inferencing engines like DROOLs [13] or DR-Prolog [22]. Our framework currently supports translation to both DROOLs and DR-Prolog but can be easily extended to

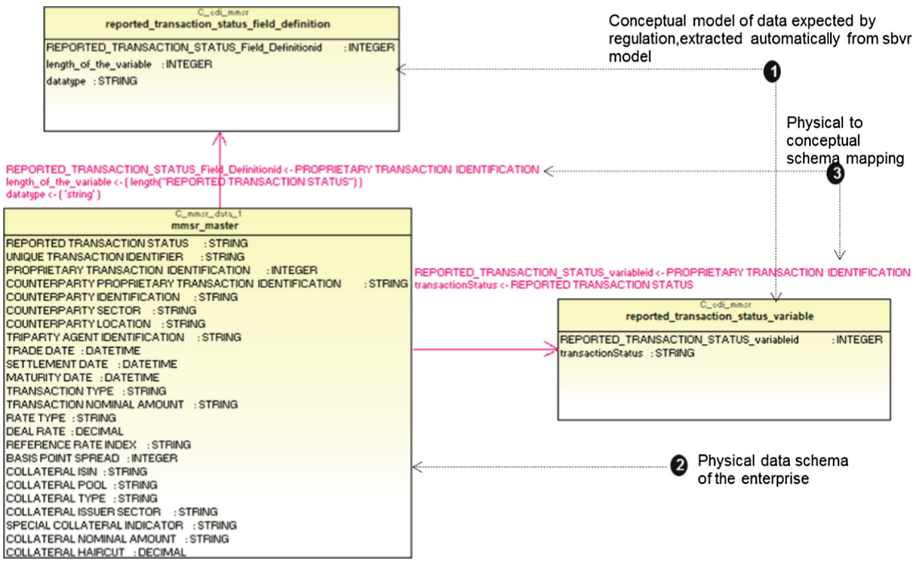

Fig. 6. Conceptual to enterprise schema mapping

support any other reasoning engine. The choice of which execution engine to be used is based on the nature of reasoning/computation required. For example, if defeasible reasoning is required to resolve conflict in modalities among rules (e.g., *obligation* vs *necessity*), then DR-Prolog is a superior choice, while DROOLs provides more fine grained Java API support for regulations that require computation. Since MMSR regulations generally require procedural computation, we chose DROOLs as our primary execution engine. As seen in Fig. 4, each SBVR model rule element (i.e., *elementofguidance*) is formed of an expression of type *implication* having an antecedent and a consequent part. At a generic level, we translate a SBVR rule to a corresponding DROOLs rule specification by mapping each antecedent with the *when* clause and each consequent with the *then* clause in DROOLs. For example, in Fig. 5, one can observe how a generated DROOLs rule for a given MMSR variable (i.e., *reported transaction status*) is translated with suitable evaluation operators. The antecedent for the *reported transaction status* variable is evaluated in the *when* part of the rule while the consequent is evaluated in the *then* part.

In addition to rule generation, one also needs data to check against the generated rules. Data typically resides in multiple physical databases (DBs) or data warehouses in an enterprise. Therefore data required for compliance checking against a particular regulation needs to be extracted from the enterprise DBs. The intermediate SBVR model here also aids the DB expert to derive a conceptual data model (i.e., data required for checking) which is then mapped against the enterprise schema. The source conceptual data model is automatically derived by inferring the leaf level nodes from the SBVR model [8]. For MMSR regulation, 49 tables were derived from the SVBR model as part of the conceptual schema. Each table typically pertains to a MMSR variable definition or field definition as seen in Fig. 6. Note, there are 24 variable definitions and 24 field definitions that corresponds to 48 derived tables and 1 master table containing ECB data.

Using our in-house enterprise data integration tool, DB experts can create relationship and mappings between the source (conceptual) and target (enterprise) schemas. A snippet of such mapping is shown in Fig. 6 for the given MMSR regulation. Once this mapping is established, suitable *select* queries are generated to pull out data from the physical enterprise DBs and populate facts by instantiating POJOs with suitable values as present in each MMSR transactions (sample dataset). An example of such an instantiation of POJOs for *Reported Transaction Status* variable is shown in the bottom right of Fig. 5.

3 Results and Discussion

In this section, we discuss the results of compliance checking with respect to our framework. The sample dataset used for testing the auto generated rules contains 90 records of MMSR transaction with respect to secured market. Each of this 90 records contains values pertaining to 24 variables of secured market segment that needed to be checked for compliance with the regulations that were authored during the earlier phases. An example of a MMSR message can be found in [11] in pg. 63 and 65 of Annex VII. We begin with describing the compliance report that was generated for the given regulation (Sect. 3 of MMSR) and dataset.

3.1 Compliance Report

Figures 7 and 8 shows the output of compliance checking with respect to the given dataset (refer to number [6] in Fig. 1). As explained in Section II, we had already obtained the rules in DROOLs executable specification and data facts in the form of POJOs (Fig. 5). The generated DROOLs rules are stored in the *Production Memory* and the facts that the Reasoning Engine matches against are stored in the *Working Memory*. The process begins by propagating facts from the working memory and asserting their values against all rules in production memory. Therefore each of the 90 facts were asserted against the 48 rules that were generated earlier and their result is shown in the form of a graphical output (snippet) as seen in Figs. 7 and 8. The graph of Fig. 7 shows the number of rule/fact pair that were successfully fired by the DROOLs engine while the graph of Fig. 8 shows the rule/fact pair that were not fired. For example out of the 90 given facts *rule r_440* was successfully fired 85 times, while it failed to fire 5 times. Similarly all the facts relating to variable *deal rate* (rule ID: *r_422*) were successfully fired. For unsuccessful facts (i.e. facts that did not satisfy the regulation), one can navigate through the graph down to the exact cause of error. The rule IDs that are preserved from rule authoring stage (Fig. 3) to SVBR model generation (Fig. 4) to DROOLs code generation (Fig. 5) stage help us to realize traceability

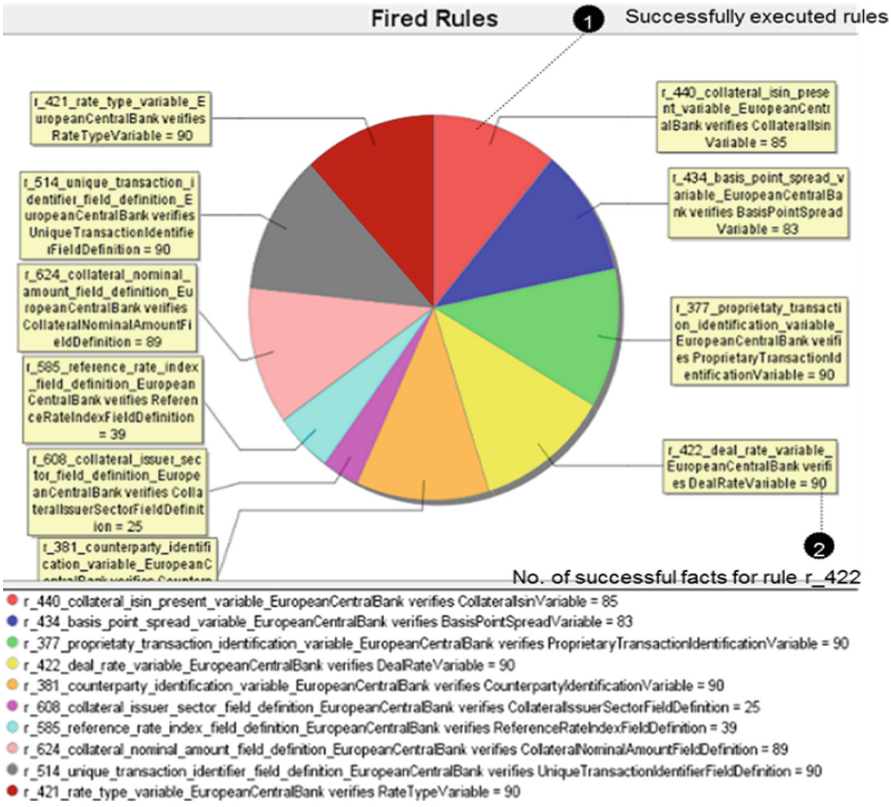

Fig. 7. Compliance report (Success rules)

in case of errors. Therefore whenever a fact is not fired, one can traverse back to SE or to the actual text of regulation in NL and obtain a formal proof of compliance or non-compliance. For our case study with 90 sample facts and 48 rules for secured market segment, the results thus obtained were 100% accurate.

3.2 MMSR Statistics

As shown in Fig. 1 the entire process of compliance checking began by processing NL text of MMSR regulations (*secured market*) [11], thereby obtaining the domain model, followed by rule authoring in Structured English followed by a series of text-to-model/model-to-text transformations along with fact population and compliance report generation. In each of these steps there have been a significant automation involved that raises the level of abstraction from low-level formal rule specification to a high-level Controlled Natural Language based rule authoring without losing traceability or specificity. However the approach is firmly grounded in formal methods and provides accurate, sound and consistent results. The following statistics will highlight some of the benefits of our approach against pure manual or tagging based implementation. In order to model secured market segment regulations in MMSR, we encountered 24 variables that either captured their conceptual or field definitions.

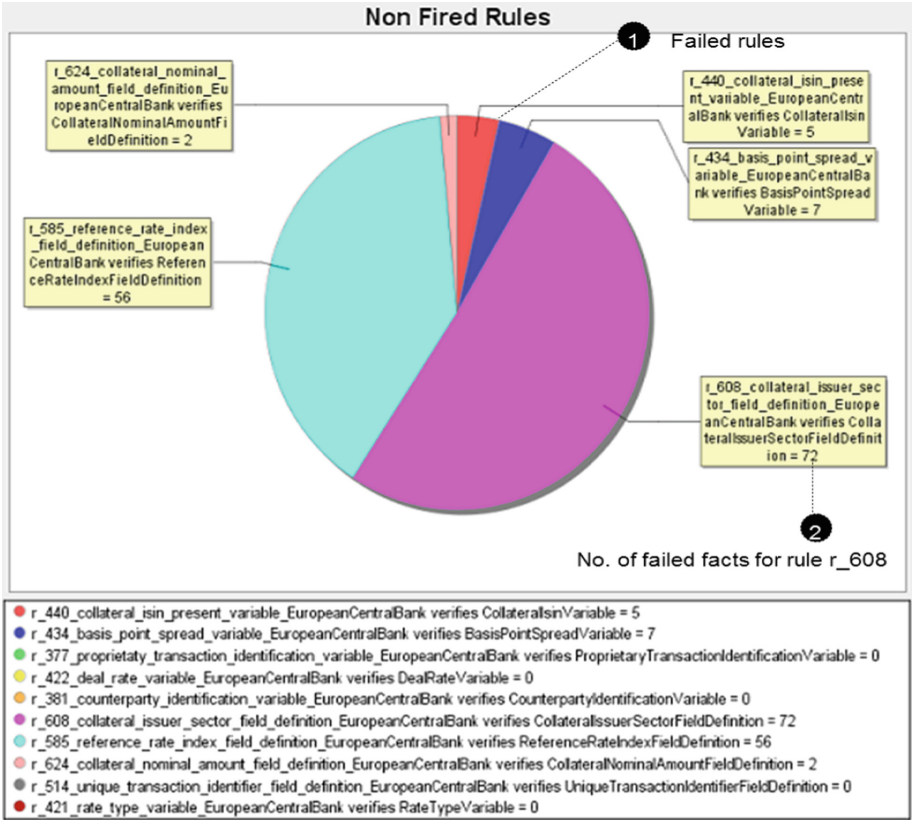

Fig. 8. Compliance report (Failed rules)

We authored 48 rules in Structured English covering all the 24 variables with the help of the domain model that captured mentions and relationships pertaining to these variables. From here on, the next chain of transformations were fully automated. We generated the SBVR model from SE rules that consisted of 582 *Terms*, 112 *Atomic Formulations,* 184 *verb concept roles*, 43 *conjunctions/disjunctions* and 62 *characteristics*. Overall, the SVBR model consisted of more than 1000 model elements which would have taken a considerable amount of time and effort to create manually, yet difficult to comprehend by a domain (human) expert. Thus, abstracting SE over SBVR gave domain experts sufficient gain in comprehensibility, yet they remained oblivious to the underlying modelling details.

Next, we automatically generated the conceptual data model (DDL) from SBVR, that consisted of 49 *tables* with 181 columns and 97 *select* queries. This served as a basis of mapping enterprise schema to the conceptual schema by a DB expert, which is required to populate the 90 data facts into suitable POJOs. Finally, using the SBVR model and mapped schemas, the framework automatically generated the DROOLs code base consisting of 497 LOC of rules specification, 2757 LOC of POJO classes and 25758 LOC of populated java objects. Quite clearly, as DROOLs code base generation was fully automated except schema mapping, it gave an order of magnitude savings in both time and effort w.r.t constructing them manually. Instead, using the compliance framework, all that the domain expert had to do was to author those 48 MMSR rules in Structured English, which was anyway much easier to comprehend and author. Nevertheless, without compromising on accuracy, the framework preserved complete traceability from any part of the tool chain to another to locate exact cause of error or inconsistency in case of non-compliance of data facts. The compliance report thus generated against the sample test dataset were accurate to the extent of 100%.

3.3 Comparison with Current State-of-Practice

The current state-of-practice in compliance checking can be categorized along three dimensions. The first dimension offers solutions in the form of governance, risk and compliance frameworks (GRC) [9]. However GRC based offerings mostly provide content management-based, document-driven and expert-dependent ways of managing regulatory compliance. They are usually semi-formal and are not as rigorous as formal approaches to compliance checking. Such techniques typically rely on tagging important concepts present in the regulations to data available in the enterprise. Such tags are generally incomplete, expert driven and lack in providing formal proof of compliance. On the contrary our approach towards compliance checking is built on the basis of formal compliance checking that offers several analysis benefits as described in [3, 7, 10], thereby reducing the burden on domain experts towards accurately covering all aspects of a regulation. Nevertheless, the domain experts are oblivious of the underlying formal techniques and operates at a level of abstraction that is closer to natural language (i.e., in the form of Structured English).

The second category of industry practice solutions is based on ETL-based queries [18] that are typically data-driven leveraging IT experts to encode SQL queries specific to regulations directly in the enterprise schemas. These queries are then executed on the enterprise data and suitable compliance reports are generated. However, such an

approach fails to leverage the knowledge of domain experts in encoding regulations, instead heavily relies on IT experts to acquire the knowledge from domain experts and fill the gap to accurately encode all parts of the regulation. This gap in knowledge between the domain and IT/DB experts can often lead to incorrectly interpreting or encoding the regulations into undesired queries resulting in inaccurate reporting. Our framework on the other hand is *non-intrusive* and provides human touch points for both domain experts in rule authoring and IT experts in schema mapping, thereby bridging the gap between them but still leveraging their respective knowledge.

The third category of industry based practice do employs NLP/ML techniques to process legal NL text [5], but do not derive SE rules (as in our approach), or an SBVR model or formal logical specification like DROOLs or DR-Prolog. These machine learning based approaches on the other hand have primarily focused on classifying the sentences/paragraphs from the legal texts into different kinds of provisions as in [2, 5], with underlying learning techniques that require training sets labelled by the domain expert, which is a cost and time intensive effort. These approaches stand in contrast to our own use of active learning, a semi-supervised machine learning technique, for rule identification based on the features engineered from the domain model and the dictionary, which we detailed in [19].

4 Conclusion

In this paper, we described a model-driven framework for semi-automatic compliance checking through a series of transformations (i.e., NL → SE → SBVR → DROOLs) involving interactive human touch points. We provided a detailed case study in validating the framework using a subset of European Union Regulation in the financial markets, namely, Money Market Statistical Reporting. The novelty of the framework was the key involvement and participation of domain-experts to author regulations in a Controlled Natural Language at an appropriate level of abstraction, while the generated formal specifications and mapping to enterprise data were managed by IT experts. This allowed better coordination for enacting compliance between domain and IT experts. In comparison to other industry based practices, our framework is built on the foundation of formal compliance checking and considerably reduced time and effort (via automation and as observed in the pilot case study) required by an enterprise to accomplish compliance checking without losing soundness, consistency or accuracy in terms of the desired result. As part of future work, we would be exploring how the framework can cater to rule changes and how to manage scalability issues with respect to validating high volume of regulatory data more effectively.

References

1. AvePoint: AvePoint compliance guardian product brochure (2014). http://www.avepoint.com/assets/pdf/Compliance_Guardian_product_brochure.pdf
2. Awad, A., Decker, G., Weske, M.: Efficient compliance checking using BPMN-Q and temporal logic. In: Dumas, M., Reichert, M., Shan, M.-C. (eds.) BPM 2008. LNCS, vol.

5240, pp. 326–341. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85758-7_24
3. Becker, J., Delfmann, P., Eggert, M., Schwittay, S.: Generalizability and applicability of model based business process compliance-checking approaches—a state-of-the-art analysis and research roadmap. BuR—Bus. Res. **5**(2), 221–247 (2012)
4. Bettini, L.: Implementing Domain-Specific Languages with Xtext and Xtend. Packt Publishing (2013). ISBN: 1782160302 9781782160304
5. Boella, G., Janssen, M., Hulstijn, J., Humphreys, L., van der Torre, L.: Managing legal interpretation in regulatory compliance. In: Francesconi, E., Verheij, B. (eds.) International Conference on Artificial Intelligence and Law, ICAIL 2013, Rome, Italy, June 10–14, 2013, pp. 23–32. ACM (2013). https://doi.org/10.1145/2514601:2514605
6. Cau, D.: Governance, risk and compliance software business needs and market trends (2014)
7. Kharbili, M.E., de Medeiros, A.K.A., Stein, S., van der Aalst, W.M.P.: Business process compliance checking: current state and future challenges. MobIS. LNI, **141**, 107–113. GI (2008)
8. Kholkar, D., Sunkle, S., Kulkarni, V.: Towards Automated Generation of Regulation Rule Bases Using mda. In: MODELSWARD, pp. 617–628 (2017)
9. KPMG: A good offense is the best defense: managing regulatory compliance with GRC whitepaper (2012)
10. Ly, L.T., Maggi, F.M., Montali, M., Rinderle-Ma, S., van der Aalst, W.M.P.: A framework for the systematic comparison and evaluation of compliance monitoring approaches. In: Gasevic, D., Hatala, M., Nezhad, H.R.M., Reichert, M. (eds.) 17th IEEE International Enterprise Distributed Object Computing Conference, EDOC 2013, Vancouver, BC, Canada, September 9–13, pp. 7–16 (2013)
11. Money Market Statistical Reporting - ECB - Europa EU. https://www.ecb.europa.eu/stats/money/mmss/shared/files/MMSR-Reporting_instructions.pdf
12. OMG: Semantics of business vocabulary and business rules (SBVR), v1.3 (2015)
13. Proctor, M.: Drools: a rule engine for complex event processing. In: Schürr, A., Varró, D., Varró, G. (eds.) AGTIVE 2011. LNCS, vol. 7233, p. 2. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34176-2_2
14. Racz, N., Weippl, E., Seufert, A.: Governance, risk & compliance (GRC) software - an exploratory study of software vendor and market research perspectives. In: 44th Hawaii International Conference on System Sciences, pp. 1–10. IEEE Computer Society, Washington, DC, USA (2011)
15. Racz, N., Weippl, E.R., Bonazzi, R.: IT Governance, risk & compliance (GRC) status quo and integration: an explorative industry case study. In: SERVICES 2011, USA, July 4–9, 2011, pp. 429–436. IEEE Computer Society (2011)
16. Roychoudhury, S., Sunkle, S., Kholkar, D., Kulkarni, V.: A domain-specific controlled english language for automated regulatory compliance (Industrial Paper). In: Proceedings of 2017 ACM SIGPLAN International Conference on Software Language Engineering (SLE 2017), Vancouver, BC, Canada. https://doi.org/10.1145/3136014.3136018
17. Roychoudhury, S., Sunkle, S., Kholkar, D., Kulkarni, V.: From natural language to SBVR model authoring using structured english for compliance checking. In: Proceedings IEEE Enterprise Distributed Object Computing (EDOC 2017), Quebec, Canada (2017)
18. Simitsis, A., Vassiliadis, P., Sellis, T.: Optimizing ETL processes in data warehouses. In: 21st International Conference on Data Engineering (ICDE 2005), pp. 564–575 (2005). https://doi.org/10.1109/icde.2005.103
19. Sunkle, S., Kholkar, D., Kulkarni, V.: Informed active learning to aid domain experts in modeling compliance. In: 2017 IEEE Enterprise Distributed Object Computing (EDOC) 2016, pp. 1–10, Vienna, Austria (2016)

20. Sunkle, S., Kholkar, D., Kulkarni, V.: Toward better mapping between regulations and operations of enterprises using vocabularies and semantic similarity. CSIMQ **5**, 39–60 (2015)
21. Sunkle, S., Kholkar, D., Kulkarni, V.: Comparison and synergy between fact-orientation and relation extraction for domain model generation in regulatory compliance. In: Comyn-Wattiau, I., Tanaka, K., Song, I.-Y., Yamamoto, S., Saeki, M. (eds.) ER 2016. LNCS, vol. 9974, pp. 381–395. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46397-1_29
22. Antoniou, G., Bikakis, A.: DR-Prolog: a system for defeasible reasoning with rules and ontologies on the semantic web. IEEE Trans. Knowl. Data Eng. **19**(2) (2007). https://doi.org/10.1109/tkde.2007.29