1-1-2010

# Analyst-mediated contextualization of regulatory policies

George Koliadis
*University of Wollongong*, gk56@uowmail.edu.au

Nirmit V. Desai
*IBM Research India*

Nanjangud C. Nerandra
*IBM Research India*

Aditya K. Ghose
*University of Wollongong*, aditya@uow.edu.au

# Analyst-mediated contextualization of regulatory policies


## Abstract

Increasing legislative and regulatory concerns have fueled an interest in effective and efficient tools for managing business process compliance within organizations. In particular, the key challenge is to understand high-level compliance policies in natural language, and interpret them for a particular usage context. These interpreted policies can then be represented in a formal language, and used to (for example) automatically verify compliance of business process executions against these policies. In this paper, we focus on the first part of this problem: interpreting regulatory policies — called contextualization. We employ a natural language parser to extract key phrases from the natural language statements and generate possible interpretations from predefined templates. An analyst chooses interpretations according to the organizational context. These interpretations are then grounded further and represented in a formal language. Via a prototype, we demonstrate our approach on real-life security compliance obligations used within IBM's IT service delivery units.



## Keywords

Analyst, mediated, contextualization, regulatory, policies


## Disciplines

Physical Sciences and Mathematics


## Publication Details

Koliadis, G., Desai, N. V., Nerandra, N. C. & Ghose, A. K. (2010). Analyst-mediated contextualization of regulatory policies. Proceedings: 2010 IEEE International Conference on Services Computing (SCC) (pp. 281-288). Piscataway, New Jersey, USA: IEEE.

# Analyst-Mediated Contextualization of Regulatory Policies

George Koliadis*, Nirmit V. Desai+, Nanjangud C. Narendra+, Aditya K. Ghose*

*University of Wollongong, Australia; +IBM Research India, Bangalore, India
{gk56@uow.edu.au, nirmit123@in.ibm.com, narendra@in.ibm.com, aditya.ghose@gmail.com}

## Abstract

*Increasing legislative and regulatory concerns have fueled an interest in effective and efficient tools for managing business process compliance within organizations. In particular, the key challenge is to understand high-level compliance policies in natural language, and interpret them for a particular usage context. These interpreted policies can then be represented in a formal language, and used to (for example) automatically verify compliance of business process executions against these policies. In this paper, we focus on the first part of this problem: interpreting regulatory policies — called* contextualization*. We employ a natural language parser to extract key phrases from the natural language statements and generate possible interpretations from predefined templates. An analyst chooses interpretations according to the organizational context. These interpretations are then grounded further and represented in a formal language. Via a prototype, we demonstrate our approach on real-life security compliance obligations used within IBM's IT service delivery units.*

**Keywords:** *Business Process, Compliance, Policies, Contextualization*

## I. Introduction

Business process compliance [9], [10] is becoming an important issue in business operations due to two trends. First, a trend towards strategic outsourcing across organizational and regulatory boundaries has increased focus on business controls [2]. Second, increasing government and institutional regulations (such as Sarbanes-Oxley Act[1], Basel-II regulations [1], and the Gramm-Leach-Bliley Act[2]) aim to control the instances of corporate anomalies and unethical practices. Global service delivery organizations, which see a confluence of these two trends, have a particularly strong need for compliance solutions.

Such regulations are meant to be general for a broader applicability across organizations and geographies. For example, consider this clause from an IBM security policy meant to apply to all divisions of IBM: *"If all members of a group have a need to know about an IBM Confidential object, that group may be granted access to the object"*. What constitutes a "need to know" is deliberately kept undefined as it could vary across divisions. A regulation cannot describe all possible justifications for a *need to know* for all geographies and organizations. Other examples include (underlines phrases are under-specified):

- *specific types of secure private information*
- *if resources are exposed due to insufficient protection*
- *acceptable resolution*
- *relevant advisory sources*

Hence, the lack of precision in such phrases is deliberate and well-intentioned rather than fallacious. Of course, there may be real fallacies in regulations due to under-specification [5] but not all instances of under-specification are fallacious. Recognizing under-specified regulations as well-intentioned and providing a framework for interpreting them according to the the context is a major contribution of this paper.

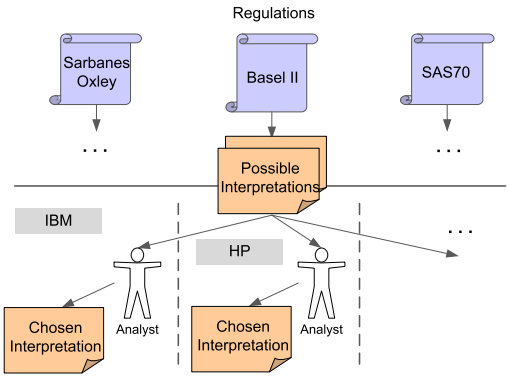


**Figure 1. Contextualization illustration**

[1]http://en.wikipedia.org/wiki/Sarbanes-Oxley_Act
[2]http://en.wikipedia.org/wiki/Gramm-Leach-Bliley_Act

978-0-7695-4126-6/10 $26.00 © 2010 IEEE
DOI 10.1109/SCC.2010.12

However, due to a well-intentioned lack of precision, the policies in the original form are not amenable to rigorous audits. Every time an audit is performed, the auditor needs to first map as well as flesh out such policies according to the organizational context. Only then can the organizational event data be rigorously checked against the regulation. This paper claims that the regulations need to be *contextualized* for an organization to enable a rigorous audit. This paper is a first step towards providing a method and a tool for contextualization.

Several works recognize the importance of a method and a tool for analyzing regulatory requirements [5], [9]. A common theme in these approaches is a manual method for annotating regulations, construction of a semantic model from annotated text, and the model's transformation to rights and obligations. However, these works do not recognize the legitimate need for a lack of precision in regulatory policies. Hence, instead of supporting a plurality of meanings for a clause, only a singular meaning is considered. This problem is not a matter of repeated application of the above methods for each context. Before a clause can be transformed into a formal or a restricted natural language, possible interpretations of it need to be considered systematically. Thus, analysts should be aided in choosing a meaning appropriate to their context from a plurality of meanings. This paper proposes and substantiates with examples, the claim that a contextualization phase enables preservation of the regulation intent during the application of the existing methods.

Figure 1 shows a high-level view of our approach. The essence of our approach is a method and a supporting tool that semi-automatically contextualizes policies. The tool is targeted for audit analysts and it processes policy regulations one clause at a time. In this paper, clauses representing only obligations, rights, and permissions are considered. For each clause, interpretations for the phrases in the clause are generated based on predefined natural language templates and previously chosen interpretations in that context. The analyst may pick one interpretation for every phrase. Based on the chosen interpretations, a set of questions are posed to the analyst. Answers to these questions would address the under-specification in the clauses. Analyst choices and answers can be stored and later used for suggesting interpretations of similar clauses in the same or even different contexts.

With these answers, the clauses can be represented formally in one of many proposed languages [12], [21], [4], [23]. For the sake of completeness and demonstration we choose commitments with deadlines as our language of representation [16]. Commitments can be verified against event traces via an application of standard model checking techniques. This paper supports the contextualization phase via a prototype tool and demonstrates its applicability on a real-life security compliance policy used within IBM's IT service delivery units.

The rest of this paper is organized as follows. Section II describes an example extracted from existing IBM controls documents, as well as an applicable scheme for representing this example. Section III outlines the core steps in contextualization. Section IV describes a prototype we have built to illustrate and test our approach. Section V discusses related work and summarizes our contributions. We conclude in Section VI.

## II. Example and Representation

For demonstration and as a basis for testing and evaluation, we have reviewed three IBM internal regulations that relate to: internal security control; customer security control; and, control practices; choosing one as a core focus. Our long-term aim is to develop a toolkit for assisting analysts and auditors in performing contextualization, audit, and reporting. For example, verifying whether an event log satisfies a set of temporal commitments extracted in the format described in Section II-B below.

### A. An Example

Consider the following obligation taken from IBM controls documentation.

> *"Existing applications and applications deployed prior to year end 2007 must encrypt specific types of SPI which are highly regulated by year end 2008."*

This obligation statement could be deemed ambiguous for a variety of reasons, classified into three broad categories [3]. Firstly, the term "SPI" may have a variety of meanings; an instance of *lexical ambiguity* [3]. For example: a collection of private user attributes — Secure Private Information (the most appropriate meaning, as identified by understanding the context of the sentence); or, a type of communication link - Serial Peripheral Interface. Another example is the references to "applications" as either: software applications; or, instances of electronic (e.g. financial) applications (although these would not typically be "deployed").

Secondly, the prepositional phrase "by year end 2008" may be attached (in the parse tree of the sentence) to the verb phrase containing "regulated" or the verb phrase containing "encrypt"; an instance of *syntactic ambiguity* [3]. That is, either: the encryption of the SPI; or, the regulation of the SPI; must occur "by year end 2008". Approaches to automatically making distinctions like these are an active area of natural language processing (NLP) research.

Finally, the reference to "specific types of SPI" and "highly regulated" require the introduction of additional

knowledge sources in order to adequately interpret the obligation; both instances of *semantic ambiguity* [3]. That is: what "specific types" are we actually interested in; and, what does it mean for a type to be "highly regulated"? The answer to these queries would be dependent on the organizational (e.g. national) context.

In order to approach the formalization of compliance documents, the above ambiguities must be resolved. Clearly, before transforming a clause annotated with syntactic information [5], it is necessary to systematically consider a plurality of interpretations that may resolve the ambiguities. Our method introduces this step which is the essence of the contextualization process and a novelty that sets this paper apart from the state-of-the-art [5], [17], [19], [3], [24]. Also, instead of aiming for automatic resolution of the above ambiguities, this paper proposes a semi-automatic approach. Our claim is that for contextualization, a semi-automatic method supported by a tool is a practical and a viable alternative to the pure machine translation-based approaches.

### B. Representation Scheme

The approach described in this paper can be used with a variety of schemes (see Section V) for representing compliance obligations. The scheme we have chosen to use is based on the language of commitments [22] extended with interval-based temporal quantifiers [16]. Commitments encode the statements in a temporal format similar to many accepted requirements engineering frameworks such as KAOS [6]. Statements in this scheme take the form $CC(x, y, e[p_l, p_u]p, e'[q_l, q_u]q)$ indicating a commitment from the debtor $x$ to a creditor $y$ to $Achieve$ or $Maintain$ (quantifiers $e$, $e'$) the condition predicate $q$ during the time interval $[q_l, q_u]$) if the precondition $p$ is met between the time interval $[p_l, p_u]$). While a commitment represents an obligation, the negation of it represents permission as shown by Singh [21]. Hence, commitments can represent commonly encountered normative modalities of obligation, permission, and prohibition (negation of the condition). The details of the semantics and related results on this language are provided by Mallya and Singh [16], and are skipped here.

For example, the following formula could be one interpretation of the example described in Section II.

$$\begin{aligned} &CC(ApplicationOwner,\\ &\quad IBMGlobalTechnologyServices,\\ &\quad True,\\ &\quad Achieve[31.12.2007, 31.12.2008]\\ &\quad PerformedApplicationEncryptSPI) \end{aligned}$$

## III. Contextualizing Compliance Documents

Here, details of the contextualization process are presented. Figure 2 shows the various steps and the artifacts produced and consumed. Rounded rectangles are automated tools, and cornered rectangles are artifacts.

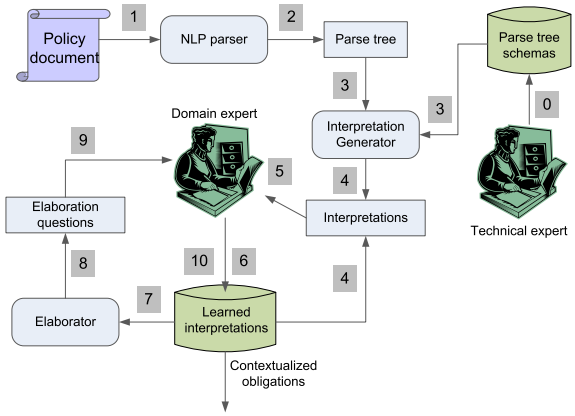


**Figure 2. A method for contextualization of regulatory policies**

A language expert creates parse tree schemas that signify language features of interest, e.g., phrases with a lack of precision or deontic modalities (Step 0). This is currently a manual task, but can be supported with learning techniques [13]. Then, an NLP parser is fed policy clauses one at a time from a policy document (Step 1). The parser generates a parse tree from a clause input from a policy document (Step 2). An interpretation generator generates interpretations from predefined templates for each of the parse tree schemas having a match in the generated parse tree (Steps 3 and 4). These interpretations are then shown to an audit analyst grouped by the text fragment that matched a parse tree schema (Step 5). The analyst chooses one interpretation for each text fragment. These choices are recorded in a repository of chosen interpretations (Step 6). Now that the lack of precision is addressed by way of chosen interpretations, under-specification is addressed by an elaboration phase (Step 7). Here, the elaborator generates questions, answers to which would enable a mapping of the chosen interpretation as a commitment formula (Step 8). These questions are posed to the analyst who provides answers according to a predefined template (Step 9). These answers are stored in the interpretations repository. This repository can be a basis for automatic checking of compliance against event traces. It can also leverage reinforced learning techniques for automatically suggesting interpretations and answers that were provided by the same or other analysts in the same or different contexts.

Below, we first describe how extraction rules are de-

veloped. This is a common pre-processing task. We then describe the main steps in contextualization with illustrative examples.

## A. Developing Extraction Rules

The contextualization process is driven by a parser that supports the matching of grammatical rules with (parsed) statements in compliance documents. Prior to analyst involvement, a language expert who understands the grammatical structure of natural language interpretations of statements in the target language must "bootstrap" the contextualization process with instances of these rules. These rules are stated as regular expressions with tree-related operators [15]. Trees recognized by these expressions can also be manipulated using another general language described in [15]. The parser we have used in our prototype [14] uses these forms of expressions.

*1) Identifying Rules:* The process of identifying extraction rules used can be manual or assisted using Information Extraction (IE)-related methods.

Our current prototype relies on a language expert prespecifying these extraction rules. The language expert must understand how statements in the target language[s] are typically structured in natural, or semi-structured, compliance documents. The literals within rules in our prototype can refer to either: part-of-speech (e.g. "NN" or noun); structural (e.g. "NP" or noun phrase); or semantic (e.g. "RESOURCE-NP" or a resource type); tags annotated to nodes in the parse tree of a sentence. The semantic tags are annotated to the parse tree during interpretation and elaboration (see Section III-B). This aims to simplify the task of creating extraction rules.

To aid in this task, the structure of constrained natural languages, such as the Semantics of Business Vocabulary and Business Rules (SBVR) [18] (providing an interface for certain logics), illustrate how some common compliance-related statements can be structured. Even with this support, we have found the task of developing these rules difficult given the large number of structures that must be considered. However, the rules and schemas (discussed in Section III-B) developed for a compliance language are highly reusable.

As future work, we aim to enhance our prototype with rule discovery techniques [13]. These techniques can be either: semi-automated, with analysts vetting extracted patterns incrementally; or automated, by using supervised, semi-supervised, or unsupervised learning techniques [13].

*2) Rule Types:* During contextualization, extraction rules are used to extract: sentences that map to statements in the target compliance language; relevant terms and conditions from compliance (and other) statements in a compliance document; and, vague and ambiguous statements, terms and conditions for analyst resolution. For example:

$$S << (VP <, (MD < may) < VP)$$

matches sentences ($S$) whose parse tree dominates($<<$) a verb phrase ($VP$) that immediately dominates ($<,$) the "may" modality ($MD$) on its left-most branch and a verb phrase on another. This rule would match sentences such as: "If all members of a group have a need to know about a Confidential object, that group may be granted access to the object". A sentence matching this rule could be interpreted as a statement of *permission* in a normative language.

In addition to rules that are used to extract statements, rules are also defined for extracting terms related to (and within) statements, such as:

$$NP\ !> NP\ !<< S$$

that matches noun phrases ($NP$) within the parse trees of sentences, not immediately dominated ($!>$) by another noun phrase and not dominating ($!<<$) a sentence ($S$). This rule would match terms in sentences such as: "Alarms must operate on emergency power".

The third type of rule in our approach to contextualization aims to identify statements, terms and conditions matching patterns of ambiguity [20] [24], such as:

$$NP <- (.* < (process, (.* > JJ|DT)))$$

that matches noun phrases ($NP$) within the parse trees of sentences, ending ($<-$) with the term "process" that is in turn preceded by an adjective ($JJ$) such as "demonstrable" or a determiner ($DT$) such as "a" or "the". This rule would match statements such as: "...a documented, demonstrable process should be established to manage and protect the private key of the self-managed Certificate Authority". In this example, the *actual* name of the process (without a reference) may be needed to qualify the object of an obligation. Although some phrases matching this rule (e.g. among the 30 we could extract from an IBM global compliance document) could be addressed by resolving coreference using IE techniques (in the case of a determiner prefix), others, such as in this example, require additional contextualization.

## B. Interpreting Compliance Statements

From an analyst perspective, the first phase of contextualization involves selecting interpretations for statements, terms and conditions within compliance documents. We will illustrate this phase using a simple statement: *"Providers of Service must set initial default protection options for user resources."*; that is extracted as an obligation. We would like to refine this statement into the

structure: $CC(x, y, e[p_l, p_u]p, e'[q_l, q_u]q)$, as described in Section II-B.

The second type of rules we discussed in Section III-A2 for extracting terms in compliance-related statements are declared in *interpretation schemas*.

An *interpretation schema* is a collection of: *interpretation schema alternatives*; and, *default interpretation rules*. Each alternative represents one means of interpreting a statement within a compliance document, and each default will automatically choose an interpretation if a rule is met. An interpretation schema alternative has (as illustrated in the example below): a unique label; an extraction rule; a sequence containing a named extraction rule and parse tree operations (referring to named patterns in the rule) for pre-processing; and an interpretation sequence containing either strings or rules. These strings and rules are used to generate a natural language statement that is used to clarify attributes pertaining to the interpretation of matching terms and conditions *after* pre-processing operations are applied to the parse tree.

For example:

**Label:** ACTOR-NP
**Rule:** $NP\ !> NP\ !<< S$
**Interpretation:** $\langle$“, (rule above),” is an Actor.$\rangle$

is one interpretation alternative for a term or condition matching the rule above. Other interpretations may, for example, label terms matching the rule as activities or attribute values. In some cases, we also include a specific interpretation schema alternative to allow an analyst to indicate that none of the other interpretation alternatives are applicable.

With respect to our running example, this first step results in the extraction of: *Providers of Service*, *user resources*, *initial default protection options for user resources*, and *set initial default protection options for user resources*; when using a set of predefined rules. These could be interpreted as: an actor; a resource; an attribute; and an action, respectively by an analyst. In the case of the action in our example, a default interpretation rule is used to automatically choose the interpretation of the action as a commitment due to the associated “must” modality:

**Label:** ACTION-COMMITMENT-MD-VP
**Rule:** $VP <, (MD < must) < VP$
**Default Interpretation:** $True$

This can be overridden by an analyst by simply deselecting the default interpretation. Defaults can also be included if: some form of enterprise ontology is available; interpretations for common terms (such as “a process”) are known; or, if a grammatical pattern is likely to conform to a certain interpretation (as in this example). In addition to default interpretations, if any of the terms in the current statement have been previously interpreted, then these are also chosen as default.

Once an interpretation is selected (by the analyst or prototype), the label of the interpretation is added to the set of labels annotated to parse tree nodes (in contrast to the single labels that are available traditionally). The interpretation alternative is also added to the set of *normative* interpretation alternatives (or rules) associated to the current statement. As previously mentioned, these are also made available during the process of interpreting other statements with similar terms and conditions. The process of applying interpretation schema alternatives then iterates, given the new labels that are available, and any new interpretations are presented to the analyst for selection.

Given the scope of languages and information we would like to extract and interpret from statements in a compliance document, the order in which candidate interpretations are presented can help minimize effort and maximize reuse. This is due to: the use of common terms between statements; and, use of semantic tags (dependencies) in the rules associated to interpretation schema alternatives. This information can be used to compute a *maximum hypothetical impact* score for a candidate interpretation alternative. This could be based on the number of interpretation alternatives (options) pruned if an interpretation is selected. When we consider the automatic selection of interpretations based on the semantic tags used in interpretation schema alternative rules, analyst effort could be significantly reduced.

The output of the interpretation phase is a set of interpreted terms, conditions, and statements (including an annotated parse tree of the statement) that are used during the elaboration and alignment phases.

## C. Elaborating on a Partial Interpretation

Completing an interpretation of a statement within a compliance document using the scheme in a compliance language may not be possible with the information available within the statement and sometimes the compliance document itself. During the elaboration phase the terms, conditions and statements that have been interpreted are presented to an analyst using the structure of the statement as per the corresponding compliance language. At the moment, our prototype supports tabular representation (compatible with many compliance schemes and IE in general).

In order to generate a structure for elaboration, *elaboration schemas* (pre-specified by a language expert) are used. An elaboration schema consists of: *elaboration schema assists* corresponding to rows (or slots) associated to statements in a compliance language; and *default elaboration assists*. Each elaboration schema assist has: a set of matching interpretation schema alternative labels (e.g. “ACTION-COMMITMENT-MD-VP”); an assist type (e.g.

"QUESTION-ANSWER"); a query sequence containing strings or expressions; and, an answer type (e.g. and enumeration, string or term matching a rule).

For example, the following assist would be used to elaborate statements labeled as ACTION-COMMITMENT-MD-VP (among others); be in the form of a query and answer; present the query "Who is debtor of this commitment?"; and present any terms interpreted as "ACTOR-NP" for selection in an answer cell (i.e. "Providers of Service" in the previous example).

**Label:** ACTION-COMMITMENT-MD-VP, ...
**Type:** QUESTION-ANSWER
**Query:** ⟨Who is the debtor of this commitment?⟩
**Answer:** RULES{ACTOR-NP}

For other elaboration assists, the analyst selects or provides answers to each query for each extracted statement. The answers to queries involving rules (as in this example) are stored as additional interpretation rules to aid in the interpretation process. In addition, default elaboration assists can be used to pre-populate cells with default values. For example, the system may be configured with a default *creditor* field: "IBM Global Technology Services"; which may be overridden.

### D. Application-level Alignment

Here, completed compliance-related statements are matched with application-level statements for the purpose of reporting. Part of this process (which we have implemented in our prototype) involves the resolution of any ambiguous term and condition candidates. These are based on known patterns of ambiguity [24] [20] and are detected using the third type of rule discussed in Section III-A2. These rules are declared in *alignment schemas*, which in-turn contain *alignment assists*. The query posed to resolve most of these cases is: "What constitutes ⟨term⟩?"; giving an analyst the opportunity to declare the term using terms available in the application (or organizational) context.

For example, "Existing applications and applications deployed prior to year end 2007 must encrypt specific types of SPI which are highly regulated." (i.e. Secure Private Information such as a client address). The current version of our prototype applies each alignment schema to generate a set of alignment assists. For an alignment assist such as:

**Rule:** $NP << , (specific\ .\ (.* > NNS))$
**Type:** QUESTION-ANSWER
**Query:** ⟨What constitutes, PATTERN{(above)}, ?⟩
**Answer:** Text

the query: "What constitutes "specific types of SPI which are highly regulated"?"; would be generated. In other words, a term beginning with the word "specific" followed by a common noun would match this rule, and could be considered a candidate for alignment. An analyst would be provided with a list of terms to select, combine or match to align the term "specific types of SPI which are highly regulated" with application level terms.

As future work, we would also like to support alignment with goal-oriented elaboration strategies [6]. In our experience, many of the terms and statements requiring alignment (esp. actions) fall into this means-end category of mappings. Once a mapping is established, increased reporting coverage can be achieved.

## IV. Prototype: Architecture and Illustration

Our prototype is a desktop application, developed with the aim of testing and illustrating the workflow we propose in the previous section as well as providing a baseline for future development. It has been built in Java on the NetBeans application platform and development environment. In the following, we briefly describe some aspects of the architecture including a typical analyst session.

### A. Architecture Overview

Our prototype must be initialized with a compliance scheme that includes the schemas required to contextualize compliance documents using our approach. Compliance documents are stored, and displayed, as trees that correspond to the structure inherent in the document. Some leaf nodes in these trees are statements extracted using IE technology. Associated to each of these statements are a set of rules that indicate whether terms and conditions within the statement correspond to a certain type of interpretation. These are reused, where appropriate, during the contextualization process.

The first screen displayed to analysts is a hierarchical document view. From this screen, analysts can select statements for contextualization. These are displayed in their current form, and three tabs are available for interpreting, elaborating and aligning the statement to an organizational and application context.

The core component of our prototype is a statistical natural language parser [14]. Parsing and matching methods are called throughout the contextualization process. Currently, the prototype does not persist information between sessions. Persistence will handled by an an object persistence layer. The prototype is initialized with a compliance scheme (the set of schemas we outlined in Section III) at design-time by implementing an appropriate class.

### B. An Analyst Session

Figures 3 and 4 illustrate the interfaces that support the interpretation and elaboration phases of contextualization

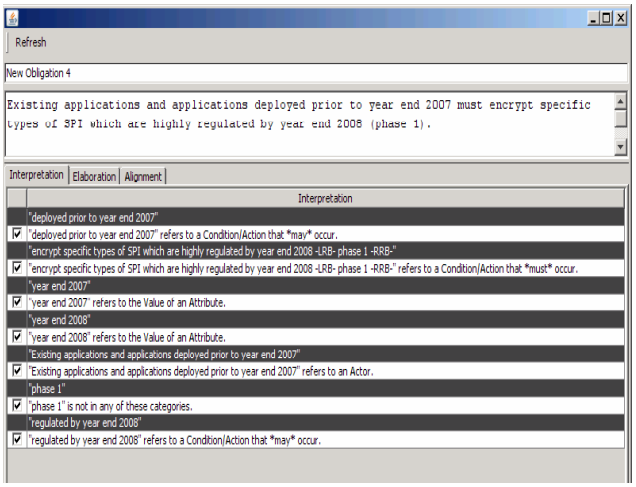


**Figure 3. Prototype Interpretation Phase**

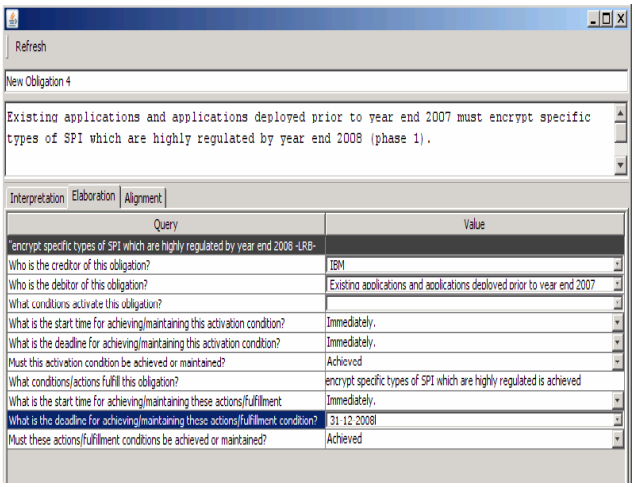


**Figure 4. Prototype Elaboration Phase**

(respectively), as implemented in our current prototype.

An analyst session begins by selecting or importing a compliance document to be contextualized. The document (along with others) is represented using a file and folder hierarchy (a tree) where some leaf nodes are compliance-related statements (screen-shot omitted for brevity).

An analyst selects a statement to contextualize in the hierarchy and is presented with an instance of Figure 3. The statement is displayed along with a table (below the statement) with candidate interpretations for selection. Based on previous sessions and any background knowledge (as discussed in Section III-B) some of these interpretations are automatically selected with the option of deselecting the interpretation in the context of the current statement available.

For example, in Figure 3 the interpretation: ““encrypt specific types of SPI which are highly regulated by year end 2008” refers to a Condition/Action that must occur.”; is automatically selected due to its prefixed modality. The analyst continues selecting interpretations, resulting in what has been presented in Figure 3.

Certain interpretations map to statements in the structure of a compliance-related language that populate the elaboration screen (Figure 4). As illustrated in Figure 4, the aforementioned statement is represented as a commitment in tabular format. The statement is mapped to the effect field in the commitment, with other fields also populated by terms extracted from the statement. These fields, and fields that do not have any values, are available for elaboration.

The analyst can review the alignment screen (not illustrated for brevity), where candidates for clarification are displayed. For example, the rule in Section III-D matches the effect of the commitment, and is displayed for clarification in a format similar to the elaboration screen.

## V. Discussion and Future Work

The approach outlined in this paper employs aspects of an information extraction architecture and can be compared to other information extraction, knowledge acquisition, and markup-centric requirements analysis approaches.

Several compliance-related grammars prescribe a set of schemas (including slots in frame-based representations), which include: nested subject-action-object triples [5]; agent-modality-literal triples sequenced using a reparational operator and prefixed by a conjunction of literal pre-conditions [10]; debtor-creditor-consequent-antecedent tuples [21]; and, actor-object-attribute-value tuples [8]. However, a prerequisite for handling more complex combinations of boolean conditions in some grammars is to identify statements matching the terms and conditions that constitute expressions in the chosen compliance-related grammar(s). In this paper, aspects of an Information Extraction (IE) [13] [11] architecture aid in automatically identifying/generating these aforementioned terms and conditions. In our approach, the source documents are unstructured, requiring the use of Natural Language Processing (NLP) techniques such as part-of-speech tagging (e.g. identifying verbs) and grammatical parsing (e.g. identifying phrases). In addition to tagging and parsing, IE systems also aid in resolving co-reference (e.g. a referent for ‘it’, or the aliases of a name). Once these preliminary processing tasks are completed, extraction patterns are typically applied to extract the values of specific attributes in the tabular grammar being used. In our current approach, we are using the *Stanford Statistical Parser* [14] for tagging, parsing and to aid in extraction, and the structure of *commitments* [7] [16] as a representation in our prototype.

It is commonly acknowledged in the compliance [5]

literature that natural language ambiguities can pose a problem for analysts and other stake holders. Some approaches proposed for dealing with ambiguity include [3]: constraining the language by controlling the words (i.e. single sense) and sentences (i.e. specific schemas) used ; learning and/or assigning conditional preferences to parsing rules; using domain knowledge to prune candidate interpretations; profiling documents against metrics and patterns that assess vague and weak sentence structure [24]; and, providing guidelines for translating natural language statements into precise formal representations [5]. In this paper, we acknowledge that some types of ambiguity [5] should not be prevented from appearing in compliance documents. In contrast, the process of contextualizing (and re-contextualizing) compliance documents should be supported using methods related to this survey - as we illustrate in our approach and prototype.

## VI. Conclusion

Many researchers and practitioners have acknowledged that the task of formalizing natural language requirement [19] and legal [5] [17] documents is a difficult, but important, problem. One aspect of this problem relates to the vagueness and generality of many compliance related statements expressed in compliance documentation. This paper presents our work towards a solution to this aspect.

We have introduced the concept of and a process for "contextualization". This process builds on the current state-of-the-art by filtering and extracting compliance-related statements from compliance documents, providing a natural language interface to (formal) target compliance languages, reducing the interpretations process to a selection and elaboration (editing) exercise, maximizing reuse within and between analyst sessions, and identifying and dealing with abstract and ambiguous terms and phrases within compliance documents.

We also aim to test our approach using a prototype we have developed. We would also like to integrate the prototype with a reporting toolkit to evaluate the performance of a combined solution to the compliance formalization and reporting problem. A reinforced learning approach can improve the tool suggested interpretations based on previously chosen interpretations.

## References


[1] *Basel II: International Convergence of Capital Measurement and Capital Standards: a Revised Framework, Comprehensive Version (BCBS)*, June 2006.

[2] Carl Abrams, Jürg von Känel, Samuel Müller, Birgit Pfitzmann, and Susanne Ruschka-Taylor. Optimized enterprise risk management. *IBM Sys. Jrnl.*, 46(2):219–234, 2007.

[3] Kathryn L. Baker, Alexander M. Franz, and Pamela W. Jordan. Coping with ambiguity in knowledge-based natural language analysis. In *Proc. FLAIRS*, 1994.

[4] Travis D. Breaux, Annie I. Antòn, and Jon Doyle. Semantic parameterization: A process for modeling domain descriptions. *ACM TOSEM* 18(2), 2009.

[5] Travis D. Breaux, Matthew W. Vail, and Annie I. Anton. Towards regulatory compliance: Extracting rights and obligations to align requirements with regulations. In *Proc. RE*, pages 49–58, 2006.

[6] Robert Darimont and Axel van Lamsweerde. Formal refinement patterns for goal-driven requirements elaboration. *SIGSOFT Software Engineering Notes*, 21:179–190, 1996.

[7] Nirmit Desai, Nanjangud C. Narendra, and Munindar P. Singh. Checking correctness of business contracts via commitments. In *Proc. AAMAS*, 2008.

[8] A. K. Ghose and G. Koliadis. Auditing business process compliance. In *Proc. ICSOC*, 2007.

[9] Christopher Giblin, Samuel Müller, and Birgit Pfitzmann. From regulatory policies to event monitoring rules: Towards model-driven compliance automation. TR RZ 3662, IBM Research, 2006.

[10] Guido Governatori, Zoran Milosevic, and Shazia Wasim Sadiq. Compliance checking between business processes and business contracts. In *EDOC*, pages 221–232, 2006.

[11] Ralph Grishman. Information extraction: techniques and challenges. In *Proc. Int. Summer School on Inf. Ext.*, pages 10–27, 1997.

[12] Andrew J. I. Jones and Marek Sergot. Deontic logic in the representation of law: Towards a methodology. *Artificial Intelligence and Law*, 1(1):45–64, 1992.

[13] Katharina Kaiser and Silvia Miksch. Information extraction - a survey. TR, Vienna University of Technology, 2005.

[14] Dan Klein and Christopher Manning. The stanford parser: A statistical parser. *http://nlp.stanford.edu/*, 2008.

[15] Reger Levy and Galen Andrew. Tregex and tsurgeon: Tools for querying and manipulating tree data structures. In *Proc. LREC*, 2006.

[16] Ashok U. Mallya, Pinar Yolum, and Munindar P. Singh. *Advances in Agent Communication*, Resolving Commitments among Autonomous Agents.

[17] Bernard Moulin and Daniel Rousseau. Automated knowledge acquisition from regulatory texts. *IEEE Expert*, 7:27–35, 1992.

[18] OMG. Semantics of business vocabulary and business rules (sbvr) final adopted specification. Tech. Report, OMG, http://www.omg.org/, July 24, 2006.

[19] Howard Reubenstein and Richard Waters. The requirements apprentice: Automated assistance for requirements acquisition. *IEEE Trans. Soft. Eng.*, 17(3):226–240, 1991.

[20] Thomas C. Rover. *Software Testing Management: Life on the Critical Path*. Prentice Hall, 2003.

[21] Munindar P. Singh. An ontology for commitments in multiagent systems: Toward a unification of normative concepts. *Artificial Intelligence and Law*, 7:97–113, 1999.

[22] Munindar P. Singh. Semantical considerations on dialectical and practical commitments. In *AAAI*, pages 176–181, 2008.

[23] Xin Wang. Mpeg-21 rights expression language: Enabling interoperable digital rights management. *IEEE MultiMedia*, 11(4):84–87, 2004.

[24] William M. Wilson, Linda H. Rosenberg, and Lawrence E. Hyatt. Automated analysis of requirement specifications. In *Proc. ICSE*, 1997.