SHAMROQ: Towards semantic models of regulations

Patrick D. Cook, Susan A. Mengal, Siva Parameswaran
Department of Mechanical Engineering, Department of Computer Science, Department of Mechanical Engineering
Texas Tech University, Lubbock, TX 79409−3104
patrick.d.cook@ttu.edu, susan.mengel@ttu.edu, siva.parameswaran@ttu.edu

***Abstract*— Regulatory documents contain a rich set of provisions that requirement engineers must observe in software requirements. If a requirement engineer fails to accurately interpret or include the provisions in the software requirements, then a right, privilege, or obligation could be omitted or incorrectly applied – resulting in a violation. When a violation occurs, complaints are filed, penalties are imposed, and in some instances, the responsible party goes to prison; thus, this paper introduces SHAMROQ, a methodology to systematically acquire software requirements from regulations, and demonstrates the methodology using a section of the Health Insurance Portability and Accountability Act (HIPAA). SHAMROQ is applied to a case study to show that it is possible to use the basic activity pattern with modality, description logic, and Hohfeldian legal concepts to analyze, classify, and model the legal relationships to ascertain meaning, context, and structure.**

Keywords- Knowledge representation, Semantic Web, OWL, Resource Description Framework, SHAMROQ

I. INTRODUCTION

Regulations contain a rich set of provisions that requirement engineers must observe in software requirements [1]. However, if requirement engineers fail to accurately interpret or include provisions in software specifications, then a right, privilege, or obligation could be omitted or incorrectly applied – resulting in a violation [2]. When a violation occurs, a complaint is filed, a penalty is imposed, and in some instances, the responsible party goes to prison.

In fact, between April of 2003 and the end of January 2020, The U.S. Department of Health and Human Services (HHS) Office for Civil Rights (OCR) received more than 227,866 Health Insurance Portability and Accountability Act (HIPAA complaints about violations of the Privacy Rule. As a result, civil penalties of $116,203,582 were settled or imposed.[1]

Apart from over $116 million in penalties, building regulatory compliant software systems presents several challenges [3]. First, regulations may complement, overlap, or contradict at the federal, state, and local levels. Secondly, regulations are continually changing, plagued with ambiguity, and often accompanied by previous administrative rulings, reference handbooks, and other guidelines published to facilitate interpretation [4]. Third, the influence of case law (the interpretation of the law by the judicial process) over statutory law (the written law passed by the legislature) poses other challenges because the courts could add new interpretations to the statues from court rulings [5]. Additionally, regulations are notable for frequent references to other sections, also known as cross-references [6], and regulations contain domain-specific language or jargon – sometimes called "legalese" [1].

Despite these challenges, researchers offer several approaches to aid requirement engineers in building regulatory compliant software systems. Approaches included logic models [7, 8], extracting formal specifications from regulations [9], goal-oriented approaches [10], production rules [11], machine learning [12] and access control [13] . More recently, researchers are using semantic web technologies to aid requirement engineers in building regulatory compliant software systems [14].

Semantic web technologies are promising because they provide a common framework that facilitates interoperability across applications, organizations, and jurisdictional boundaries. Moreover, the semantic web offers a family of technologies that enable requirement engineers to create data stores, construct vocabularies, and write rules for dealing with data[2]. In this research, we leverage semantic web technologies, in particular, the Web Ontology Language (OWL) to aid requirement engineers in systematically acquiring software requirements from regulations.

The purpose of this descriptive, embedded, single-case study is to develop and validate the SHAMROQ methodology. At this stage in the research, SHAMROQ is generally defined as the systematic process to examine all words and phrases written in a regulatory document, classify patterns that correspond to Hohfeldian legal concepts, and model the regulations using the basic activity pattern with modality.

The remainder of this paper is organized as follows: Section 2 reviews the background and related work; Section 3 outlines the methodology; Section 4 describes the case study; Section 5 presents the findings; Section 6 examines the threats to validity; and Section 7 discusses the conclusion and future work.

[1] https://www.hhs.gov/hipaa/for-professionals/compliance-enforcement/data/enforcement-highlights/index.html?language=en

[2] https://www.w3.org/standards/semanticweb/

DOI reference number: 10.18293/SEKE2020-144

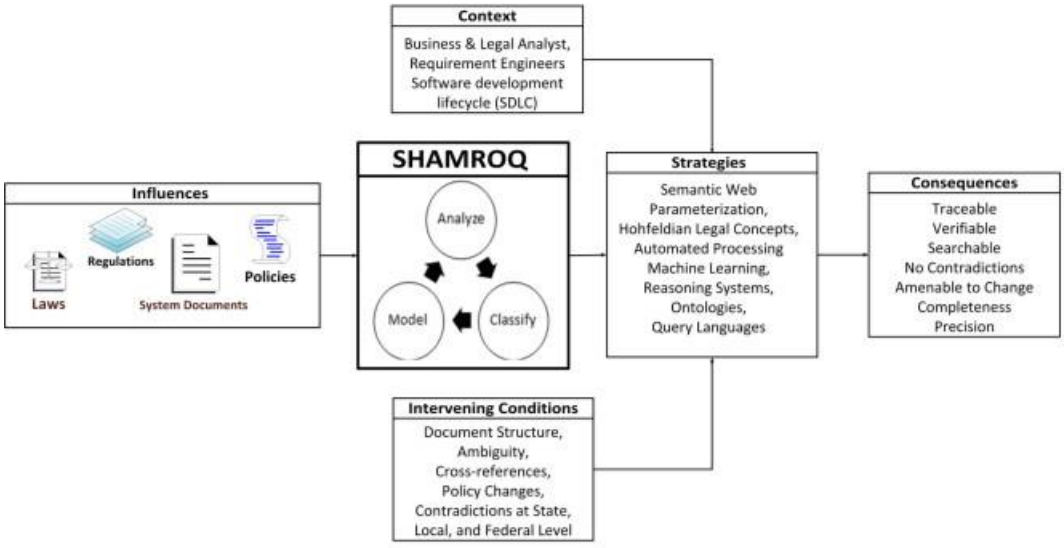

Figure 1. SHAMROQ - Theoretical Framework

II. Background and Related Work

In this section, the background and related work in requirements engineering to extract software requirements from regulations is considered.

A. Background

Our working definition of requirements engineering, borrowed from Pamela Zave and generalized by Phillip A. Laplante, is the following: "Requirements engineering is the branch of engineering concerned with the real-world goals for, functions of, and constraints on systems. It is also concerned with the relationships of these factors to precise specifications of system behavior and to their evolution over time and across families of related systems" [15].

Laplante argues that software systems are bifurcated along functional (i.e., what the system does) and nonfunctional (how well the system does it under observable quality attributes) requirements. SHAMROQ provides a means to address both, however, this paper concentrates on nonfunctional.

Nonfunctional requirements are further broken down into design/implementation constraints, economic constraints, operating constraints, and political/cultural constraints. In this work, we will focus on the political/cultural constraint; i.e., the laws and regulations category, of nonfunctional requirements.

B. Related Work

Researchers use a variety of approaches to extract requirements from regulations and model them for system development. A comprehensive survey of the approaches is outlined by Otto [3]. Here, we concentrate on the related work that directly influences the ideas in our research: Semantic Parameterization [1, 2, 16-18], Frame-based [19, 20], and Production Rules [11, 21, 22].

Semantic Parameterization [1, 2, 16-18] is a process to represent a domain of interest in a structural way using Description Logic [23]. This process happens over three phases. In phase 1, phrase heuristics are applied to natural language features, so that noun phrases, pronouns, intentional and extensional synonyms, and polysemes are differentiated. In phase 2, a dictionary is used to assign meaning to the words so that the domain is grounded. In phase 3, the dictionary is used to identify the tacit relationships to build a meta-model. As a result, Restricted Natural Language Statements (RNLS) are modeled using the basic activity pattern with modality. RNLS are derived from the original text and are restricted to one discrete activity. The RNLS is then represented by the basic activity pattern using one unary relation and two asymmetric, binary relations. The *unary relation* defines the root concept σ, while parameters use *associative relations* α, and values use *declarative relations* δ.

To further illustrate the point, Breaux uses the following RNLS as an example: "***The provider may share information***." Figure 2 depicts the *unary relation* σ ($activity_1$) in the shaded region. The *associated relations* α ($activity_1$, $actor_1$), α ($activity_1$, $action_1$), α ($activity_1$, $object_1$), is captured by the shaded region and oval. The *declarative relations* δ ($actor_1$, provider), δ ($action_1$, share), δ ($object_1$, information) is captured by connecting the directed arrow between the oval in the shaded region with the ovals outside of it.

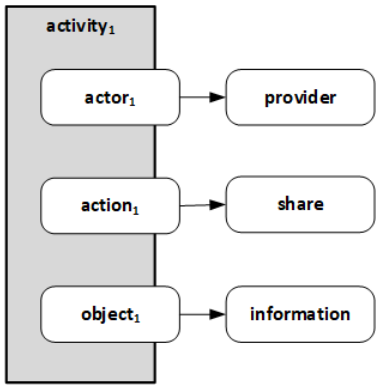

Figure 2. Basic Activity Model

Breaux contends that these three relations represent a complete parameterization process when all words and phrases written in a regulatory document are assigned or subsumed by a parameter or value. In instances where parameter values require concepts with additional parameters, then a second parameterization takes place with an additional associative and declarative relations.

In figure 3, we see the additional *associative relations* α ($activity_1$, $purpose_1$), and the additional declarative relations δ ($purpose_1$, $activity_2$) that represents the RNLS: "***The provider may share information to market services.***" Note, the preposition "to" is indicative of an additional *associative relation*.

This research builds on Semantic Parameterization and extends this work in the following ways. First, we codify the unary relation, as a root node, with a Hohfeldian legal concept [24, 25]. For example, the statement, "***The provider may share information to market services,"*** uses the modal verb "may," to establish the root node "Privilege Activity."

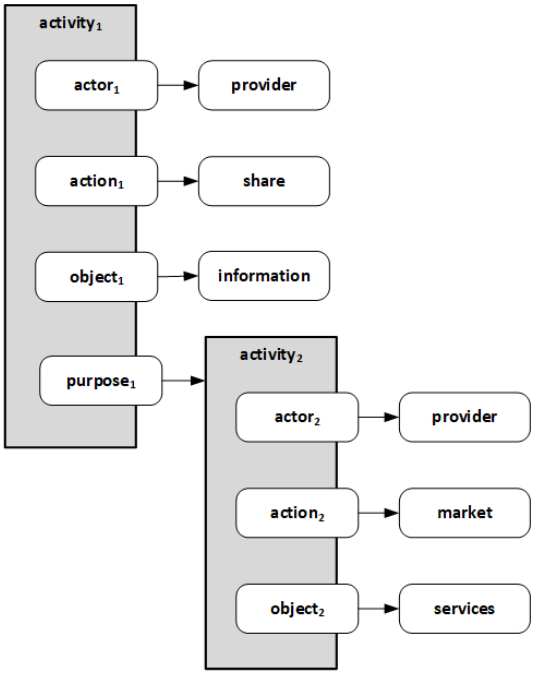

Figure 3. Basic Activity Model with Purpose

Secondly, we reduce the steps presented by Breaux [16]from UNLS, RNLS, Activity Model (3 steps) to UNLS, Activity model (2 steps). Third, we combine the associate and declarative relations and explicitly represent the activity as Resource Description Framework (RDF) *triples*. Next, we add a meta-data-model to the basic activity pattern that consists of the following attributes: a unique identifier, category, title, priority, and degree of necessity. We capture the triples in Figure 4.

Figure 4 shows the root node, PrivilegeActivity, and the associate relations as predicates (i.e., hasActor, hasAction, hasObject, and hasPurpose). Also depicted are the declarative relations as objects (i.e., provider, share, information, and PurposeActivity), and the meta-data-model as predicates.

The Frame-Based Requirements Analysis Method (FBRAM) [19, 20], is another means of extracting requirements from regulations. Breaux uses FBRAM to annotate the regulatory document manually in order for a tool to parse the annotations to extract the requirements. From this extraction, three artifacts are produced: an upper ontology, a context-free markup, and a document model.

The upper ontology is used to classify regulatory statements and consists of three concepts: a statement-level used to categorize individual regulatory statements, a phrase-level used to categorize individual regulatory phrases, and an abstract placeholder. The context-free markup describes the structure using concepts and logical connectives. The analyst uses the context-free markup to make some interpretation about the text and aligns the upper ontology in a manner that removes ambiguity.

The document model describes how the document is organized using a hierarchical representation. Moreover, the document model enables traceability between the requirements and the section, subsections, and paragraphs of the original regulations. The requirements are represented as HTML, in a table format, and contain the frame type; i.e., the type of requirement, the pattern, and the traceability information

Similarly, to FBRAM, we examine the natural language features of regulatory documents and map concepts to an ontology. However, our approach differs from FBRAM in that we extract software requirements directly from the regulations with natural language processing techniques and use Web Ontology Language Description Logics (OWL-DL) to express requirements as opposed to a document model to formalize the legal syntax. Moreover, our work focuses on all eight Hohfeldian legal concepts – not just rights and obligations.

The Production Rule Methodology [11, 21, 22] codifies four sections of the HIPAA Privacy Rule (§164.520, §164.522, §164.524 and §164.526) SWI-Prolog software application [21]. A production rule is a knowledge representation technique that is stated using horn clauses connected by logical operators [22]. Each rule consists of a two-part structure: an antecedent and a consequent. If the antecedent set of conditions resolves to true, then the consequent set of actions takes place. A collection of rules creates a knowledge base. The interaction with this knowledge base requires the top-level query using an inference engine; for example, backward chaining, as a reasoning strategy to execute on the rules base [22].

Prior to getting started, the production rule methodology requires an ontology and some legal text as input. Then, a preparatory step (Create Rule Patterns of Ontological Concepts) followed by two activities (Specify Production rules and Refactoring, respectively) takes place. In the preparatory step, production rule patterns are created from the ontology. The first activity, specify production rules, requires five steps.

In step 1, normative phrase analysis is used to classify rules based on the words and phrases used in the legislation. In step 2, identify rule parameters, the objective is to identify the subject of the statement, the relation the actor can change, the action the actor has the right or obligation to perform, and the source of the rule. In step 3, identify preconditions, the legal preconditions

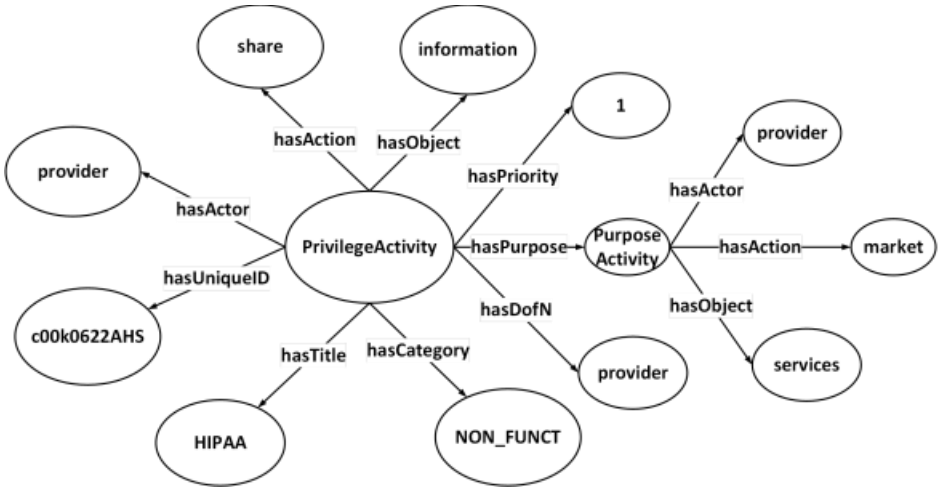

Figure 4. Semantic Web Parameterization

that enable the rule to be true are captured. In step 4, remove the rule, disjunctions, the statements in the legislation that are separated by an "or," are split into separate statements. Finally, in step 5, identify rules implied by the ontology, the software engineer may deduce other facts. After the completion of the first activity, a complete production rule model exists. However, the second activity, which refactors the rules base to remove duplicates, provides an opportunity to improve the design.

Like the production rule methodology, we use a multi-step process to extract requirements from regulations. We evaluate the natural language phrases and classify patterns that correspond to Hohfeldian legal concepts. Unlike the production rule methodology, we read the regulations directly from a file, segment the sentences from the regulations, tokenize the strings, tag the words with parts of speech, and chunk the sentences in a manner that can be modeled with OWL-DL.

III. METHODOLOGY

The research methodology is borne out of a constructivist worldview [26]. The philosophical idea around constructivism is to seek understanding of the world in its real-world context and is typically associated with qualitative research. Moreover, constructivists interpret meanings others have about the world or generate theory inductively as opposed to starting with theory. This induction is directly in contrast to the postpositivist worldview. The philosophical ideas of the postpositivist start with a theory, collect data to support or refute the theory, then revise, and are typically associated with quantitative research.

The constructivist worldview is necessary in this research because we seek to understand a phenomenon in its real-world context. Therefore, we adhere to the qualitative research design and the case study strategy of inquiry. Yin [27] describes a case study as an empirical method that takes an in-depth analysis of a contemporary phenomenon within a real-world context. Case study design includes four types: single-case embedded, single-case holistic, multi-case embedded, and multi-case holistic. The choice to use a single vs. multi-case study design is based on the number of cases in a study.

A case is a centralized phenomenon that exists within a real-life context. Within the context of software engineering, a case may range from a software development project to a process, product, team, technology, specific role, or policy [28]. Consequently, if only one case exists, then it is best to select a single case study. However, if two or more cases exist, then it is best to select the multi-case study design.

The choice to use an embedded vs. holistic is based on whether the case study has multiple units of analysis; i.e., subunits, or the case study examines the global nature of a phenomenon. Yin defines a unit of analysis as the actual source of information (e.g., a person, organizational document, or an artifact.)

Runeson [28] elaborates on the unit of analysis for software engineering as a project, group, or a decision. In short, a unit of analysis is the phenomenon within a case that is examined. On the other hand, to examine the global nature of a phenomenon means a holistic view of the case is assessed, and there are no subunits. Therefore, if the case study has multiple units of analysis, then one selects an embedded case study. If the case study looks at the nature of the whole phenomenon, then one selects a holistic approach.

This research employs an embedded, single-case study research design as defined by Yin and as recommended for software engineering by Runeson. This design is intentionally chosen with a long-range strategy in mind. We intend to leverage the results of this descriptive study to support future studies that will be prescriptive in nature. In the next section, we outline the case, units of analysis, research questions, theoretical framework, and strategy for mitigating threats.

IV. CASE STUDY

1) Case Selection

This study purposely selects HIPAA regulation §164.510 (a)(1), as illustrated in figure 2, because this specific provision of the regulation provides normative phrases, continuations, exceptions, and parameter values that are concepts with other parameters.

2) Units of Analysis

Yin defines the unit of analysis as the element within the case study for which the data is collected [27]. For software engineering research, Runeson stated that the unit of analysis might be some element of the project, the methodology, or some aspect of the ongoing development or maintenance [28]. Here, the unit of analysis consists of the natural language features (i.e., the keywords, sentences, phrases, and clauses) that form the parameters applied to the basic activity model.

3) Research Questions

Creswell declares that qualitative research questions are central with associated sub-questions [26]. A central research question takes a broad view and explores a central phenomenon. In this study, the following main central question outlines a broader view of the purpose statement to describe and explain the SHAMROQ methodology. To what extent can SHAMROQ be used to build a knowledge base? What is the SHAMROQ framework? How does the SHAMROQ methodology work in practice to extract requirements from regulatory documents?

4) Quality Assurance – Mitigating Threats to Validity

The quality of a case study is evaluated based on its ability to identify and mitigate threats to validity. Yin outlines four tests to assess the threats to validity. The four tests are construct, internal, external, and reliability. This study addresses three out of the four threats to validity. Internal validity applies to explanatory or causal studies and does not apply to descriptive or exploratory studies [27].

5) Theoretical Framework

SHAMROQ represents a contribution to the body of knowledge formalized by a systematic literature review (SLR) as outlined by Barbara Kitchenham [29] and meta-ethnography synthesis as outlined by Noblit and Hare [30]. A meta-ethnography synthesis (MES) uniquely and systematically defines a qualitative process for generating theory, which involves induction and interpretation. Meta-ethnography places emphasis on maintaining alignment with the original research articles and encourages researchers to extend beyond the original ideas of the research [30].

A clear finding of the SLR and MES was that several strategies are required to analyze, classify, and model regulations. SHAMROQ is a manifestation of those strategies. In the next section, we provide an overview of the SHAMROQ and will answer the central research question, to what extent can SHAMROQ be used to extract requirements from regulations.

V. Findings and Discussion

This section presents the evolution of SHAMROQ and answers the research questions.

A. What is the SHAMROQ Framework?

SHAMROQ is an acronym that embodies the strategies used to build a knowledge base: semantic web parameterization, Hohfeldian legal concepts, Artificial Intelligence, Metadata Enrichment, Reasoning System, Ontologies, and Query language. Collectively, these seven core strategies provide requirement engineers a means to analyze, classify, and model functional and nonfunctional requirements using the semantic web. As illustrated in figure 1, there are four main artifacts that influence SHAMROQ.

First, depicted in figure 1, are the laws that are established by Congress. Secondly, are the regulations (i.e., rules) that implement a statue or act as a guide. Third are the system documents that represent the stakeholder's needs, goals, deliverables, constraints, limitations, security, and performance criteria. Some examples of system documents are a Statement of Work (SOW), Software Requirement Specification (SRS), and Concept of Operations (CONOPS). Finally, are the policies that are an assortment of legal artifacts to include executive orders and presidential actions.

The context depicted in figure 1 shows the people involved and contends that they must be a part of the software development lifecycle. The intervening conditions include the characteristics that laws, regulations, and policy artifacts exhibit that make them both beneficial and problematic; in particular, the legal document structure, ambiguity, cross-references, and frequent changes.

The next construct of the framework, strategies, are reflective of the techniques to carry out the analysis, classification, and modeling of the artifacts that influence the framework – given the context and intervening conditions. As a result, the strategies yield a set of consequences that make legislative documents traceable, verifiable, searchable, absent of contradictions, complete, precise, and amenable to change.

Given the number of strategies to unpack, we narrow the scope of this paper to semantic web parameterization, Hohfeldian legal concepts, and ontologies. In the next section, we describe how the SHAMROQ methodology works in practice to extract requirements from regulations.

§ 164.510 Uses and disclosures requiring an opportunity for the individual to agree or to object.

(a) *Standard: Use and disclosure for facility directories—*

(1) *Permitted uses and disclosure.* Except when an objection is expressed in accordance with paragraphs (a)(2) or (3) of this section, a covered health care provider may:

(i) Use the following protected health information to maintain a directory of individuals in its facility:

(A) The individual's name;

(B) The individual's location in the covered health care provider's facility;

(C) The individual's condition described in general terms that does not communicate specific medical information about the individual; and

(D) The individual's religious affiliation; and

(ii) Use or disclose for directory purposes such information:

(A) To members of the clergy; or

(B) Except for religious affiliation, to other persons who ask for the individual by name.

Figure 5. §164.510 (a)(1) of HIPAA

B. How does the SHAMROQ methodology work in practice to build a knowledge base of regulatory documents.

In this section, we describe the methodology that supports the framework. The methodology aids practitioners in providing a formal means to represent legal provisions, minimize

ambiguity, trace a requirement from its basic activity model back to its origin, and enrich the data model with metadata. What follows is a description of the automated process to analyze and classify and the manual process to model. To illustrate, we use the example of §164.510 (a)(1)(i)(A) of the HIPAA Privacy Rule, to describe each phase.

1) Analysis

In the analysis phase, we download the XML version of the regulations from the govinfo.gov website [3] and perform the following automated processing on the text using the Python programming language and Natural Language Toolkit (NLTK) [31]. First, the XML file is scanned to obtain the root node of the regulation. Secondly, preprocessing is performed on the document by traversing the root node and extracting the information associated with the node.tag, the node.attrib, and node.text. The analysis phase concludes when the results are stored in a python dictionary that contains the metadata, header, and body structure.

2) Classify

In the classification phase, the python dictionary is taken from the analysis phase and NLTK aids in performing sentence segmentation on the body of the python dictionary which contains the regulatory text. Next, NLTK helps to segment the regulatory text into sentences, to tokenize each sentence into words, and to tag each word with a part of speech. Finally, a grammar, illustrated in figure 6, helps to chunk the tagged words into eight categories: section, topic, noun phrase, exception, modality, conjunction, continuance, and action.

```
SECTION: {<\(><DT|CD><\)>?}
Topic: {<JJ>+<NNS><\.>}
EXCEPTION: {<IN><WRB><DT><NN.*>+}
NOUN_PHRASE: {<IN>?<DT>?<JJ.*>*<NN.*>+}
MODALITY: {<MD>}
CONJ: {<CC>}
CONTINUANCE: {<\:>}
ACTION: {<RB.*>*<VBZ>?<VB|VBN|VBZ>*<RB|RBR>?}
```

Figure 6. Grammar

Chunks, with a focus on noun phrases and verbs, are inspected manually in the output to identify an actor, action, object, target, exception, or constraint. A search for the modal verb is performed to assign to the root activity. Table 1 outlines the normative phrases [1] that align with Hohfeldian legal concepts[24, 25].

Table 2 captures the predicates that are aligned with the subject, verb, and object along with other attributes to support building a model that represent the regulation. The classification phase ends when all words and phrases written in a regulatory document are mapped to a category.

TABLE I. HOHFELDIAN CLASSIFICATION

Serial No.	Modality and Normative Phrase Correlation	
	Normative Phrase	***Concept***
1	has a/the right to, retains the right to	Right
2	must, is required to, shall, may not, is prohibited, is subject to	Obligation
3	may, may elect not to, is not required to, requirement does not apply, is permitted to, at the election of, is not subject to	Privilege
4	does not have a right to	No-Right
5	authorize termination of, must obtain an authorization, may revoke, may terminate	Power
6	provide that <actor> will/must, obtain assurance	Liability
7	None	Immunity
8	may not authorize	Disability

The sentences are further examined manually to ascertain continuances [32]. Continuances are clauses that break into multiple constituent parts. The constituent parts are appended to the base clause and must be classified and modeled separately in the following manner [21].

TABLE II. ACTIVITY CLASSIFICATIONS

Serial No.	Classification Scheme	
	Predicate	***Description***
1	hasActor	The subject of the clause and answer the ICM question who
2	hasAction	The verb of the clause and answers the question what
3	hasModality	The auxiliary verb that corresponds to a Hohfeldian legal concept
4	hasObject	The verb of the clause and answers the question what
5	hasTarget	The person, place, or thing receiving an action
6	hasPurpose	The goal or objective of the clause
7	hasException	Contains keywords that express an exception
8	hasConstraint	Contains keywords that express a constraint
9	hasSource	Contains the legislation source section

a) § 164.510 (a)(1)(i)(B)

Except when an objection is expressed in accordance with paragraphs (a) (2) or (3) of this section, a covered health care provider may: (i) Use the following protected health information to maintain a directory of individuals in its facility: *The individual's location in the covered health care provider's facility;*

a) 164.510 (a)(1)(i)(C)

Except when an objection is expressed in accordance with paragraphs (a) (2) or (3) of this section, a covered health care provider may: (i) Use the following protected health information to maintain a directory of individuals in its facility: *The individual's condition described in general terms that does not communicate specific medical information about the individual; and*

[3] https://www.govinfo.gov/content/pkg/CFR-2019-title45-vol2/xml/CFR-2019-title45-vol2-sec164-510.xml

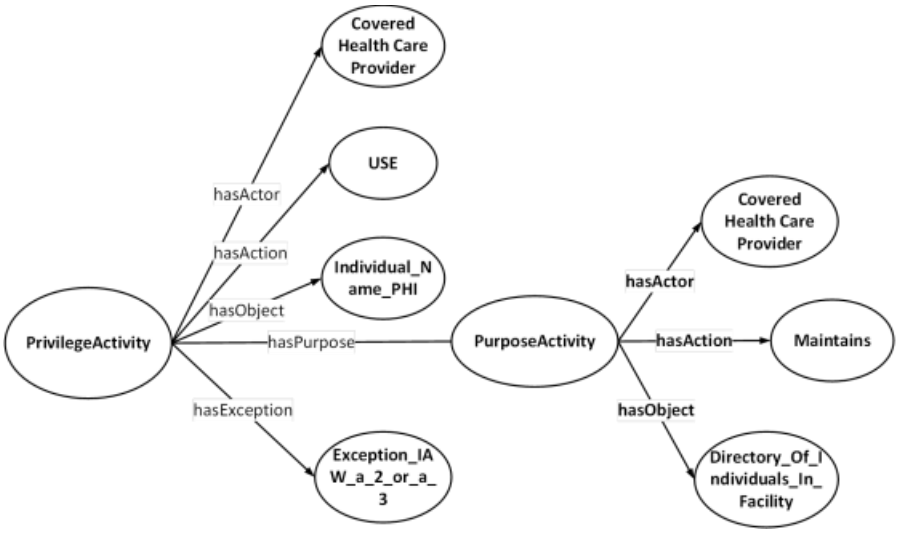

Figure 7. Semantic Web Parameterization

b) § 164.510 (a)(1)(i)(D)

Except when an objection is expressed in accordance with paragraphs (a) (2) or (3) of this section, a covered health care provider may: (i) Use the following protected health information to maintain a directory of individuals in its facility: *The individual's religious affiliation; and*

3) Model

In this phase, the output of the classification phase is inspected manually and protégé [33] is used to represent an organized, logical representation of concepts and categories using semantic web parameterization. The root activity contains the assignment based on the modal verb in the normative phrase. To provide more clarity, we continue with the example of § 164.510 (a)(1)(i)(A).

In figure 7, the illustration shows the results of the Semantic Web Parameterization process, in particular, the assignment of the root activity $\sigma'(\text{PrivelegeActivity}_0)$. The root activity is a privilege according to the phrase, "a covered health care provider may:" A look in Table 1, serial 3 shows the modal verb "may" maps to a privilege – the root activity.

The combined associate and declarative relations become the following triples: $\alpha'(\text{privelegeActivity}_0, \text{hasActor}_0, \text{some Health care Provider})$, $\alpha'(\text{privelegeActivity}_0, \text{hasAction}_0, \text{some Use})$, $\alpha'(\text{privelegeActivity}_0, \text{hasObject}_0, \text{some Name Individual Activity})$, $\alpha'(\text{privelegeActivity}_0, \text{hasPurpose}_0, \text{some Purpose Activity})$, $\alpha'(\text{privelegeActivity}_0, \text{hasException}_0, \text{some CFR 164 (a) (2)\_Activity})$, and $\alpha'(\text{privelegeActivity}_0, \text{hasException}_0, \text{some Exception\_IAW\_a\_2\_or\_a\_4})$

VI. Threats to Validity

In this paper, we used section §164.510 of HIPAA as a descriptive, embedded, single-case study to develop and validate the SHAMROQ methodology. To assess the quality of a case study, Yin describes four criteria: construct validity, internal validity, external validity, and reliability. Internal validity is used for explanatory or causal case studies and not for descriptive or exploratory studies [27]. Therefore, internal validity is not tested here. In this section, we discuss construct validity, external validity, and reliability.

Construct validity assesses the correctness of operational measures by evaluating the means in which the researcher collects data, builds, or validates theory, and reports results [27]. Yin outlines three case study tactics to mitigate the threats to construct validity: use multiple sources of evidence, establish a chain of evidence, and use key informants to review the draft case study [27].

Although the case study in this paper uses one section of HIPAA, the basis of the SHAMROQ methodology is grounded in the literature and based on prior theories of semantic parameterization [16], description logic [23], and Hohfeldian legal concepts [24, 25]. We establish a chain of evidence by following our methodology and retaining copies of all artifacts. Lastly, the authors listed here reviewed the draft case study report.

External validity assesses whether the results are specific to the phenomenon under investigation or are applicable more generally [27]. We acknowledge several threats to external validity in our case study. First, we only examine one legal text within one regulatory domain - HIPAA. However, we purposely selected §164.510 because this provision provides normative phrases, continuations, exceptions, and parameter values that are indicative of legal text. Further studies modeling more legal texts across multiple domains will serve to validate and refine the methodology.

Reliability assesses whether the research can be independently verified, using the same methodology, to yield the same results [27]. Researchers independently verifying our case study are likely to use different grammar rules, identify a different combination of noun phrases, and identify a different ontology. Therefore, a small probability exists that the exact results of our case study could be replicated.

However, reliability is improved by evaluating reliability against our documented process and case-study database.

VII. Conclusion and Future Work

In this paper, we introduced SHAMROQ, a methodology to examine all the natural language features in a regulatory document, group the features according to their shared characteristic, and model the features using the semantic web parameterization. We analyzed HIPAA regulation §164.510, which contained provisions with normative phrases, continuations, exceptions, and cross-references that are indicative of legal text. Our findings show that it is possible to use the basic activity pattern with modality, description logic, and Hohfeldian legal concepts to analyze, classify, and model the legal relationships to ascertain meaning, context, and structure.

Future work will include automating the manual steps to generate a semantic model from the noun phrases produced by the chunked grammar. Moreover, we will refine and validate SHAMROQ with a more extensive legal corpus across multiple regulatory domains and evaluate to what extent a multi-class classification machine learning algorithm can classify Hohfeldian legal concepts for semantic modeling.

References

[1] T. D. Breaux, M. W. Vail, and A. I. Anton, "Towards regulatory compliance: Extracting rights and obligations to align requirements with regulations," in *Requirements Engineering, 14th IEEE International Conference*, 2006: IEEE, pp. 49-58.

[2] T. D. Breaux and A. I. Antón, "Analyzing regulatory rules for privacy and security requirements," *Software Engineering, IEEE Transactions on*, vol. 34, no. 1, pp. 5-20, 2008.

[3] P. N. Otto, A. Antón, and others, "Addressing legal requirements in requirements engineering," 2007 2007: IEEE, pp. 5-14. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4384161[Online]. Available: files/272/abs_all.html

[4] E. Kamsties, "Understanding ambiguity in requirements engineering," in *Engineering and Managing Software Requirements*: Springer, 2005, pp. 245-266.

[5] M. J. Sergot, F. Sadri, R. A. Kowalski, F. Kriwaczek, P. Hammond, and H. T. Cory, "The British Nationality Act as a logic program," *Communications of the ACM*, vol. 29, no. 5, pp. 370-386, 1986.

[6] J. C. Maxwell, A. I. Anton, and J. B. Earp, "An empirical investigation of software engineers' ability to classify legal cross-references," in *Requirements Engineering Conference (RE), 2013 21st IEEE International*, 2013: IEEE, pp. 24-31.

[7] L. E. Allen, "Symbolic logic: A razor-edged tool for drafting and interpreting legal documents," *Yale LJ*, vol. 66, p. 833, 1956.

[8] T. J. Bench-Capon, G. O. Robinson, T. W. Routen, and M. J. Sergot, "Logic programming for large scale applications in law: A formalisation of supplementary benefit legislation," in *Proceedings of the 1st international conference on Artificial intelligence and law*, 1987: ACM, pp. 190-198.

[9] T. D. Breaux, A. I. Ant, and #243, "Mining rule semantics to understand legislative compliance," presented at the Proceedings of the 2005 ACM workshop on Privacy in the electronic society, Alexandria, VA, USA, 2005.

[10] A. Antón, J. B. Earp, A. Reese, and others, "Analyzing website privacy requirements using a privacy goal taxonomy," 2002 2002: IEEE, pp. 23-31. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1048502[Online]. Available: files/279/abs_all.html

[11] J. C. Maxwell, A. I. Ant, and #243, "The production rule framework: developing a canonical set of software requirements for compliance with law," presented at the Proceedings of the 1st ACM International Health Informatics Symposium, Arlington, Virginia, USA, 2010.

[12] J. Cleland-Huang, A. Czauderna, M. Gibiec, and J. Emenecker, "A machine learning approach for tracing regulatory codes to product specific requirements," presented at the Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1, Cape Town, South Africa, 2010.

[13] M. J. May, C. A. Gunter, and I. Lee, "Privacy APIs: Access control techniques to analyze and verify legal privacy policies," 2006: IEEE, pp. 13-pp.

[14] P. Slootweg, L. Rutledge, L. Wedemeijer, and S. Joosten, "The Implementation of Hohfeldian Legal Concepts with Semantic Web Technologies," *AI4J–Artificial Intelligence for Justice*, p. 65, 2016.

[15] P. A. Laplante, *Requirements engineering for software and systems*. CRC Press, 2017.

[16] T. D. Breaux and A. I. Antón, "Analyzing goal semantics for rights, permissions, and obligations," 2005: IEEE, pp. 177-186.

[17] T. D. Breaux, A. I. Antón, and J. Doyle, "Semantic parameterization: A process for modeling domain descriptions," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 18, no. 2, p. 5, 2008.

[18] T. D. Breaux and A. Anton, "Deriving semantic models from privacy policies," in *Policies for Distributed Systems and Networks, 2005. Sixth IEEE International Workshop on*, 2005: IEEE, pp. 67-76.

[19] T. D. Breaux and A. I. Antón, "A systematic method for acquiring regulatory requirements: A frame-based approach," *RHAS-6), Delhi, India*, 2007 2007.

[20] T. D. Breaux, *Legal requirements acquisition for the specification of legally compliant information systems*. ProQuest, 2009.

[21] J. C. Maxwell and A. I. Anton, "A refined production rule model for aiding in regulatory compliance," North Carolina State University. Dept. of Computer Science, 2010.

[22] J. C. Maxwell and A. I. Anton, "Developing production rule models to aid in acquiring requirements from legal texts," in *Requirements Engineering Conference, 2009. RE'09. 17th IEEE International*, 2009: IEEE, pp. 101-110.

[23] F. Baader, I. Horrocks, C. Lutz, and U. Sattler, *Introduction to Description Logic*. Cambridge University Press, 2017.

[24] W. N. Hohfeld, "Fundamental legal conceptions as applied in judicial reasoning," *The Yale Law Journal*, vol. 26, no. 8, pp. 710-770, 1917.

[25] W. N. Hohfeld, "Some fundamental legal conceptions as applied in judicial reasoning," *Yale Law Journal*, pp. 16-59, 1913.

[26] J. W. Creswell, *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage, 2013.

[27] R. K. Yin, *Case study research and applications: Design and methods*. Sage publications, 2017.

[28] P. Runeson, M. Host, A. Rainer, and B. Regnell, *Case study research in software engineering: Guidelines and examples*. John Wiley & Sons, 2012.

[29] B. Kitchenham and P. Brereton, "A systematic review of systematic review process research in software engineering," *Information and Software Technology*, vol. 55, no. 12, pp. 2049-2075, 12// 2013, doi: http://dx.doi.org/10.1016/j.infsof.2013.07.010.

[30] G. W. Noblit and R. D. Hare, *Meta-ethnography: Synthesizing qualitative studies*. Sage, 1988.

[31] E. Loper and S. Bird, "NLTK: the natural language toolkit," *arXiv preprint cs/0205028*, 2002.

[32] W. M. Wilson, L. H. Rosenberg, and L. E. Hyatt, "Automated analysis of requirement specifications," 1997, pp. 161-171.

[33] M. A. Musen, "The protégé project: a look back and a look forward," *AI matters*, vol. 1, no. 4, pp. 4-12, 2015.