Regulatory Requirements as Open Systems: Structures, Patterns and Metrics for the Design of Formal Requirements Specifications

Travis D. Breaux **David G. Gordon†**

February 25, 2011

CMU-ISR-11-100

Institute for Software Research
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, 15213

Abstract

Increasingly, information systems are becoming distributed and pervasive, enabling organizations to deliver services remotely to individuals and to share and store personal information, worldwide. However, system developers face significant challenges in identifying and managing the many laws that govern their services and products in this new multi-jurisdictional environment. To address this challenge, we explore the concept of a computational requirements document expressible using a formal requirements specification language (RSL). The purpose of this document is to make requirements open and available to policy makers, business analysts and software developers, alike. We show how document authors can codify policy and law using the RSL and design, debug, analyze, trace, and visualize relationships among requirements from different policies and regulations. The RSL provides new model-based constructs for expressing multi-jurisdictional, distributed constraints and navigating a regulatory narrative and conditional surface structure. In addition, the RSL makes regulatory specification patterns visually salient and enables metrics to quantitatively measure different compositional styles for writing legal and policy documents. We discovered and validated the RSL using nine U.S. state data breach notification laws that govern transactions of financial and health information of residents of these nine states.

This research was supported by the U.S. Department of Homeland Security under Grant Award Number 2006-CS-001-000001, under the auspices of the Institute for Information Infrastructure Protection (I3P) research program.

† Engineering and Public Policy

Keywords: requirements specification; traceability; formal languages; legal requirements

1 Introduction

The Internet and wireless computing are enabling increasingly distributed and pervasive systems. In distributed systems, computer state, data and functions can be stored in multiple remote servers independent of the client's geographic location; in pervasive systems, the client is physically unbounded and can connect to the network from multiple locations. In both cases, software developers must respond to government and industry regulations that affect their product and service requirements. Because these regulations are bound to the geography in different ways (such as the area where the data is stored, where users access remote services, or where users are citizens), developers must contend with a multi-jurisdictional environment. In addition, new laws are enacted each year to improve information privacy and security, often in response to unanticipated and innovative uses of computer technology. This changing environment puts requirements in motion and necessitates new tools to manage this change.

In the United States, a prominent example includes the recent surge in state data breach notification laws, which have been empirically observed to reduce identity theft [21]. Collectively, these laws combine the act of notification with technical security controls targeted at different information types, business practices and consumers. The challenge for developers, especially in small businesses, is to distill these regulations into actionable requirements that are traceable across their business practices. In addition, these requirements must be integrated with industry standards to further articulate how businesses comply with government laws [22].

We believe that existing approaches to governance, which consists of independently published, paper-based laws and policies, can no longer scale with rate of technology innovation. Furthermore, if an honest expectation of compliance is to be preserved in this new environment, regulations must be made accessible to policy makers, business analysts and software developers, alike. We propose that regulators and industry can reach a coordinated solution wherein regulations are computational artifacts, dynamically linked across jurisdictions. These computational artifacts can integrate with industry standards to become more easily comparable and addressable in a manner that reflects the jurisdiction of the computer state, users' location, and the rate of technological change. To this end, we report our efforts to study a requirements specification language (RSL), derived from grounded analysis of conflicting regulations from multiple jurisdictions. By compiling specifications in the RSL, document authors can design and debug their requirements documents in an open system using formal structures, patterns and metrics that we discuss in this paper.

The remainder of the paper is organized as follows: in Section 2, we discuss related work; in Section 3, we introduce the RSL by example; in Section 4, we present our research methodology to discover the RSL; in Section 5, we discuss our research findings, including techniques for navigating and cross-linking legal requirements; and in Section 6, we conclude with discussion and future work.

2 Related Work

Related work discussed in this section includes research to develop formal requirements languages, extract requirements from laws, prioritize requirements, and model legal documents and their legal effects.

Requirements specification languages (RSLs), including requirements modeling languages (RMLs), have a rich history in requirements and software engineering [14]. RSLs include informal, natural language descriptions to provide readers with context and elaboration, and formal descriptions, such as mathematical logic, to test assumptions across requirements using logical implications [7]. Goal-oriented languages, such as i* [25] and KAOS [6], and object-oriented notations, such as Adora [11], include graphical notations to view relationships between entities, such as actors, actions and objects. Because of computational intractability and undecidability of using highly expressive logics [GMB94], RSLs often formalize only a select class of requirements phenomena, e.g., using various temporal logics, including interval [MBK90], real-time [6] or linear [8] temporal logic, or description logic [2]. Consequently, RSLs and RMLs may struggle with the balance between expressability and readability [7].

Unlike i*, KAOS and Adora, the RSL proposed herein is designed for the policy domain by integrating formal expressions of document structure with semi-formal expressions of rights, permissions and obligations. The RSL prioritizes readability by requiring limited formalization of: actor roles, constraints on those roles and Boolean logic to express pre-conditions; definitions and their scope of applicability; and cross-references as typed relations between requirements. Unlike frame-based approaches that seek to classify phrases by logical roles [4], our RSL simulates how polices are written using limited formalization to enable analysis using Description Logic and graph theory.

Approaches to formalize laws in requirements engineering have focused on prescriptions, called rights, permissions and obligations [3], ownership and delegation [9], and production rule systems [17]. Cross-references within and among laws have been shown to coordinate definitions, exceptions and refinement and must be addressed in a comprehensive legal requirements management strategy [5]. Recent analysis of external cross-references emanating from the Health Information Portability and Accountability Act (HIPAA) shows the potential for conflicts between laws [18].

The ability to prioritize requirements has long been a research challenge in requirements engineering. Laurent et al. use automated clustering of requirements into orthogonal categories and allow analysts to assign weights to clusters to prioritize requirements [12]. Liaskos et al. assign preferences to goal hierarchies complemented by pre-condition relations to infer priorities for goals and tasks using planning specification languages [15]. Herein, prioritization means the ability to identify requirements that apply to a business and specifically the order in which requirements must be evaluated. We infer these priorities from logical implications expressed in a narrative in the RSL-generated model, as discussed in Section 5.

In artificial intelligence and law, document modeling is an important topic to manage legal complexity. Document modeling includes the entire regulatory drafting lifecycle (e.g., enactment, repeal, enforcement) and related commentary, including legal opinions [13, 19, 23]. In this paper, a document model consists of the section and paragraph structure of a document, the titling, and the ability to unambiguously cross-reference definitions and requirements across documents. Our present focus on requirements modeling limits our work to the as-is state of regulations and their inter-relationships, excluding important opinions and legal evolution, which we defer to future work.

Within this limited scope, Bourcier and Mazzega propose a vision to model legal documents using networks, wherein legal articles are nodes connected by edges that represent either "legal influences" or quotations, called "legal selection" [1]. They advocate for content-based measures that account for legal effects produced by normative statements [1]. Massey and Anton propose several metrics for measuring regulation dependency and complexity [16]. Our RSL addresses these needs in three respective ways: a) by codifying legal influences in typed, priority-based relations (including exemptions, pre-emptions and waivers) that cross-link between portions of regulatory documents; b) by assigning types to cross-references between individual requirements (a much finer level of detail) that encodes certain legal effects, such as refinement, exception and pre- and post-conditions; and c) by demonstrating how measuring these relations quantifies dependency and complexity exhibited in legal writing styles.

The need for software tools that can reference and itemize subtexts has been recognized [13, 19], with criticism that modern tools often only link at the document level [13]. There is also a need to type cross-references [23]. While some believe references are syntactically unambiguous [23], we measured semantic ambiguity in cross-references that occur when linking to specific regulatory rules (see Section 5) and we propose a technical solution in Section 4.

Requirements Specification Language

The requirements specification language (RSL) makes several assumptions about the domain of requirements. These assumptions were first observed in our case study and thus incorporated into the RSL syntax and semantics described here. As we discuss later, they support what we believe are good requirements specification practices. In the discussion that follows, we use the following excerpt from Arkansas Title 4, §110.105 to present the RSL:

4-110-105. Disclosure of security breaches.

(a)(1) Any person or business that acquires, owns, or licenses computerized data that includes personal information shall disclose any breach of the security of the system… to any resident of Arkansas…

(2) The disclosure shall be made in the most expedient time and manner possible and without unreasonable delay, consistent with the legitimate needs of law enforcement as provided in subsection (c) of this section

Figure 1. Excerpt from the Arkansas (AR) Title 4, §110.105 of the Personal Information Protection Act

2.1. Document Model

The RSL is applied directly to original text, converting statements and phrases from the text into expressions in the RSL. Figure 2 shows the excerpt from Figure 1 expressed in the RSL: reserved keywords, special operators, and line numbers (found along the left side) appear in bold. The `DOCUMENT` keyword (see line 1) is used to assign a unique index to the specification. The `SCHEMA` keyword (see line 2) is followed by an expression consisting of components in curly brackets. Each component corresponds to a different reference level within the document model, beginning with the outermost component, in this case the title and chapter. References within the specification will be parsed using this schema. Line comments are indicated by the "//" operator. We use the ellipsis "…" to indicate omissions from the specification to simplify presentation in this paper.

```
1       DOCUMENT US-AR-4-110
2       SCHEMA {title:4}-{chapter:110}-{section:\d+}{par:\([a-z]\)}
        {par:\(\d+\)} //...
3       TITLE 4-110 Personal Information Protection Act
4
5       SECTION 4-110-105 Disclosure of security breaches
6       PAR (a)
7       PAR (1)
8       person!
9          | business!
10         & acquires, owns, or licenses computerized data that includes
             personal information
11         : shall disclose a breach of the security of the system to any
             resident
12      PAR (2)
13      disclosure!
14         : shall be made in the most expedient time and manner possible
             and without unreasonable delay
15         ANNOTATE timing requirements
16         REFINES (1)
17         EXCEPT (c)(1) #1
```

Figure 2. Excerpt from Arkansas 4-110-105 expressed in the RSL

The document model consists of sections and nested paragraphs, expressed in the RSL by the `SECTION` and `PAR` keywords, respectively. These keywords are followed by a reference and an optional title. For example, line 5 shows the section reference 4-110-105 followed by the section title from §105 in the excerpt in Figure 1; sub-paragraphs (a) and (1) follow on lines 6-7. The parser validates the references against the previously declared document schema and constructs an internal document model that is used with cross-references to lookup definitions and requirements.

2.2. Roles, Constraints and Requirements

Requirements consist of roles and constraints on a role, organized into first-order logical expression using operators "|" for logical-or (see line 9), and "&" for logical-and (see line 10). Associativity in logical expressions is inferred from the number of tabs before the logical operator: one less tab than the previous line is right associative; otherwise the logic is left associative. Roles are noun phrases that describe the actors or objects to whom the requirements apply and are followed by "!" (see lines 8-9); constraints on a role are phrases that begin with a verb (see line 10). For a role R and constraint C, we always assume the sentence "R who C" is valid and grammatically correct for the purpose of generating natural language from this formalization. Roles and constraints are part of the pre-conditions in a requirement. Next follows the requirement clause, preceded by a ":" and modal verb, such as "shall" to indicate an obligation (see lines 11 and 14). We identify these modal verbs using established phrase heuristics [3]. Finally, the analyst can write commands in the RSL to instruct the parser to perform special operations on rules. In Figure 2, the command keyword `ANNOTATE` (see line 15) indicates that the following text contains comma-separated annotations that should be linked to the requirement. Annotations can be used to group requirements by shared themes.

2.3. Relations and Cross-References

Requirements are related to each other through relations and cross-references. The RSL includes several commands by default for expressing relations and can accommodate more as needed. The default commands are:

- `REFINES`, with the inverse `REFINED-BY`, indicates that this requirement is a sub-process or quality attribute that describes how another requirement is fulfilled.
- `EXCEPT`, with the inverse `EXCEPT-TO`, indicates that this requirement has an exception (another requirement). If the pre-conditions of the exception are satisfied, then this requirement does not apply (it becomes an exclusion, e.g., is not required).
- `FOLLOWS`, with the inverse `PRECEDES`, indicates that this requirement is a post-condition to another requirement, e.g., this requirement is permitted, required, or prohibited after the other requirement is fulfilled.

In Figure 2, the command keyword `REFINES` (see line 16) indicates that this requirement refines the requirements in paragraph (1). This example is a quality attribute, because the refinement on line 13 elaborates the act on line 11 (to disclose), elaborating when the act must occur (expediently, without delay). Generally, quality attributes describe the act or an object in the act of another requirement. The command keyword `EXCEPT` (see line 17) indicates this requirement has one exception in paragraph (c)(1): the first requirement.

Relative references in these commands are expanded by the parser using a simple algorithm: in Figure 2 for example, starting from the source paragraph (2), the parser ascends the document model checking the document schema for a descending match rooted at the current paragraph. Thus, the first check is for a matching sibling paragraph: in this case, the index (a)(1) is a match. References may be either: an index to a singular paragraph; a paragraph range separated by the "--" operator; the ".." operator, which matches the parent paragraph; or the "." operator, which matches the current paragraph. References followed by a "*" operator refer to the paragraph and all sub-paragraphs (i.e., transitive closure). Rule selection is done in three ways: a) by default, references select all rules within the referenced paragraph(s); b) singular paragraph references followed by the ordinality operator "#" and a number n will identify the nth rule in that paragraph (see line 17); and c) references followed by a comma-separated list of annotations will find rules that share those annotations (e.g., all "permissions" or all "timing requirements"). Using the last mechanism, document authors can organize requirements around aspects or themes shared across a system and index requirements accordingly.

2.4. Definitions

Definitions describe the actors and objects in the environment of the system. They can be used to organize roles and constraints into a single term-of-art, which allows document authors to substitute the term for repeated logic across requirements. For requirements with complex pre-conditions, we found this simplification to make reading the specification much easier. Because regulations govern multiple industries and systems, it is also important to coordinate and reuse definitions across separate regulations. Consider the following RSL specification in Figure 3, acquired from Nevada Chapter 603A, Security of Personal Information, §215(5), which describes definitions related to security measures for businesses who collect payment cards.

In Figure 3, paragraph (a) on lines 3-9 contains a definition for data storage device, indicated by the "=" operator. Definitions are expressed similar to pre-conditions and can use the logical operators for logical-and and logical-or, in addition to the operator "<", which means "includes" and precedes examples or sub-classes (see line 7), and the operator "~", which means "excludes" (see line 13). The parser assumes definitions apply to the paragraph in which they occur, unless instructed using the `INCLUDE` keyword, followed by two references: the source location of the definitions, and the target section or paragraph to which the definitions will apply. The instruction in Figure 3, line 2 tells the parser to apply all the definitions from paragraph (5) and all sub-paragraphs (indicated by the "*") to §215. In contrast, the `INCLUDE EXTERNAL` instruction on line 15 instructs the parser to lookup the definition "payment card" by finding a regulatory document indexed by NV-205.602, and to apply this definition to §215. This second usage enables reuse of definitions from and across multiple regulations.

```
  PAR 5.
  INCLUDE 603A.215.5* 603A.215*
  PAR (a)
  data storage device
     = device
        &  stores information or data from any electronic or
           optical medium
        <  computers
           | cellular telephones
  // ...
  PAR (c)
  facsimile
     =  electronic transmission between two dedicated fax machines
        using Group 3 or Group 4 digital formats...
     ~  onward transmission to a third device after protocol
        conversion, including, but not limited to, any data storage
        device
PAR (d)
INCLUDE EXTERNAL NV-205.602 603A.215* payment card
```

Figure 3. Excerpt from Nevada 603A.215(5)(c) expressed in the RSL

2.5. Tool Support and Generated Artifacts

The RSL is complemented by an automated parsing tool, which checks the language for syntax errors, such as malformed or unassociated logical expressions, and semantic errors, such as incorrect references, empty relations that refer to no rules, unreferenced definitions, and cycles among relations of the same type, e.g., REFINES, EXCEPT, FOLLOWS. The parser also handles pre- and post-clause continuations [3], wherein one or more roles and constraints apply to rules in sub- or parent paragraphs, respectively. Lastly, the parser annotates the requirements using phrase heuristics that indicate the modality of the clause [3], for example, "may" indicates a "permissions" annotation, or "shall" indicates an "obligations" annotation. Annotations are used to sort and reference requirements.

The parser constructs a model from the RSL, which is exported to other formats, such as the eXtensible Markup Language1 (XML), HyperText Markup Language2 (HTML) and the Graph Markup Language3 (GraphML). Each format offers a different perspective: the HTML allows users to browse the specification by clicking hyperlinks, viewing definitions and referenced rules in context of a single rule; the GraphML allows users to visualize relationships across multiple requirements; and the XML enables data inter-operability and exchange with other tools. Figure 4 shows a graph generated from the RSL example in Figure 1: text labels include a unique requirement identifier (e.g., AR-7), followed by the roles in parentheses and the requirement clause (abbreviated in this figure). Nodes are colored by whether they are permissions (green), obligations (yellow), prohibitions (red) and exclusions (blue) based on annotations. Directed edges represent relations and point to referenced rules as follows: solid edges are REFINES, dashed edges are EXCEPT, and dotted edges are FOLLOWS relations.

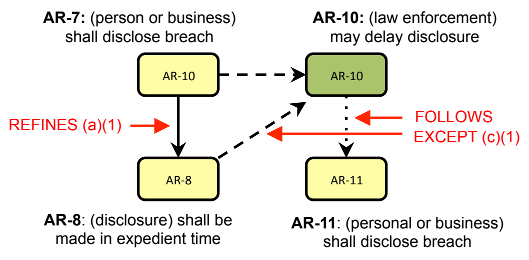
AR-7: (person or business) shall disclose breach
AR-10: (law enforcement) may delay disclosure
AR-10
AR-10
REFINES (a)(1)
FOLLOWS
EXCEPT (c)(1)
AR-8
AR-11
AR-8: (disclosure) shall be made in expedient time
AR-11: (personal or business) shall disclose breach

Figure 4. Excerpt from Arkansas §110.105 expressed in GraphML

[1] http://www.w3.org/XML/
[2] http://www.w3.org/html/
[3] http://graphml.graphdrawing.org/

We foresee importing our models into other tools that support open requirements exchange formats, such as the Requirements Interchange Format4 (RIF) and the User Requirements Notation (URN).

3 Research Methodology

Science includes the codification of knowledge from repeatable observations. Whereas science often invokes experimentalism, which includes observing human behavior under established environmental controls, computer science provides formal languages as a basis for describing world knowledge. We now describe our case study research method [24] used to develop a formal language from repeated observations of natural language expressions in regulatory documents. The method includes our selection criteria, the translation process, and units of analysis.

While the overarching goal of our study is to observe variation in regulations across multiple jurisdictions with the aim of understanding how regulations introduce complexity into system requirements, this paper only presents preliminary results to this end. To observe this variation, we selected a single theme (data breach notification) to illustrate dependencies between two sets of phenomena: functional requirements of systems, and personnel responsibilities of the environment. In the United States, this theme represents the recent enactment of 46 state and territorial laws from 2002-2009, each governing personal information about state residents. For distributed and pervasive systems, variations in these laws require businesses to reconcile different legally required practices for customers of different states. The laws we selected are as follows:

1. **AK**: Personal Information Protection Act, Alaska Chapter 45.48, enacted 2009.
2. **AR**: Personal Information Protection Act, Arkansas Chapter 14.110, enacted 2005.
3. **MA**: Security Breaches, Massachusetts Chapter 93H, enacted 2007.
4. **MA-S**: Standards for the Protection of Personal Information of Residents of the Commonwealth, Massachusetts Chapter 17, enacted Sep. 19, 2008.
5. **MD**: Personal Information Protection Act, Maryland Subtitle 14-35, enacted 2008.
6. **NV**: Security of Personal Information, Nevada Chapter 603A, enacted 2006.
7. **OR**: Oregon Consumer Identity Theft Protection Act, Oregon Chapter 646A, enacted 2008.
8. **VT**: Protection of Personal Information, Vermont Chapter 26, enacted 2007.
9. **WI**: Notice of Unauthorized Access to Personal Information, Wisconsin Chapter 134.98, enacted 2006.

We down selected from 46 laws to 9 laws using two criteria: first, we invited suggestions from a legal expert with seven years of privacy and security law expertise to highlight industrial challenges, resulting in AR, MA-S, MA, MD, NV; second, we selected three laws with the largest number of pages, resulting in AK, OR, VT. Finally, we included the State of Wisconsin, because it uniquely covers biometric information, including fingerprints, voice prints, retinal images and unique physical characteristics.

Our translation process consisted of two investigators (the authors) separately translating each statement within each law using the RSL. The process includes a general classification of each statement, as a definition, requirement, exemption, etc., and writing an expression in the language to characterize the statement. Definitions were identified by common phrases, such as "*x* means *y*", where a term *x* has the logical definition *y*. Requirements were identified using the phrase heuristics identified by Breaux et al. [3]. Comments were used in the translation to capture questions, issues and other discrepancies. We maintained a caveats list of translation strategies that reflect unusual cases and how the parser should treat such cases, and a proposed changes list of requirements with examples for new language constructs. As a new construct was introduced, we reviewed each law to update the translation to reflect the new construct to ensure consistency across the entire dataset.

The units of analysis correspond to the language constructs: definitions, requirements, exemptions, and operational statements for mapping definitions to sections and paragraphs. The RSL acts as a natural filter, capturing only what it can express, which is a threat to validity discussed in Section 6.

4 Research Findings

The translation of the nine laws by two investigators (the authors) required an average of 2.60 minutes per statement with the longest document consisting of 142 statements and requiring an average of 5.5 hours. Each investigator spent an average total of 26 hours to encode the nine laws. Figures 5 and 6 present summary statistics for the units of

[4] http://www.omg.org/spec/ReqIF/

analysis encoded in the RSL. Recall these laws cover the same theme (data breach notification). We observed the number of definitions did not vary greatly and that the number of exemptions was a matter of writing style; neither definitions nor exemptions are proportional to the number of requirements in this dataset.

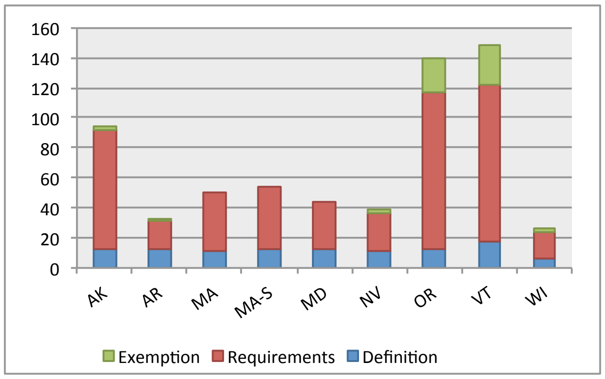

Figure 5. Summary Units of Analysis– Statements

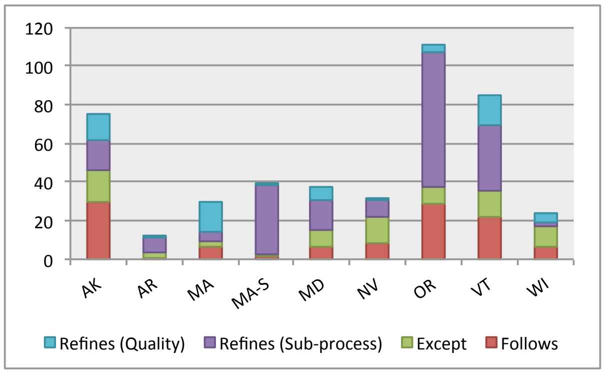

Figure 6. Summary Units of Analysis – References

The references reported in Figure 6 originate from multiple origins, including: anaphora, which is indicated by determiners (such) and pronouns (this); case-splitting, which is indicated by English conjunctions (and, or) separating verb clauses that are headed by a modal phrase (must, may, shall); and direct references to sections and paragraph that may be anaphoric (this section, this paragraph) or indexed by paragraph number, such as "paragraph (a)." Table I presents summary statistics for each origin. For direct references, we present the number of corresponding rules identified by the original reference for each regulation, called direct literal (dL), and the number of corresponding rules indexed by the operationalized reference using the RSL language construct, called direct indexed (dI). Because the operationalized references are more precise, we can calculate the ambiguity loss, which is (dL – dI) / dL. As shown in Table I, the operationalized references reduce ambiguity by up to 93%.

TABLE I. CROSS-REFERENCE ORIGINS AND AMBIGUITY

State Law	Anaphora	Case Split	Direct	Direct Literal	Direct Indexed	Ambiguity Loss
AR	2	4	5	24	7	0.708
AK	16	19	35	143	36	0.748
MA	20	1	3	45	3	0.933
MA-S	4	34	1	2	1	0.500
MD	4	12	21	62	23	0.629
NV	7	3	13	83	14	0.831
OR	29	15	24	190	24	0.874
VT	36	10	25	269	32	0.881
WI	6	0	18	78	20	0.744

We now discuss four sets of observations enabled by the RSL: (A) how definitions and exemptions can be used to shape conditionality, or when and how businesses must implement the law; (B) the existence of a narrative and conditional surface structure evidenced by the typed relations; (C) three prominent regulatory structures made salient by the RSL-generated model; and (D) metrics for comparing regulatory specification writing styles.

4.1. Shaping Conditionality and Coverage

Conditionality means the extent to which a legal requirement is conditioned by who stakeholders are and what events have occurred. Definitions and exemptions scope conditionality by relaxing or tightening the meaning of terms and thus scaling the number of possible situations those terms cover. We discuss two ways in which these conditional effects are observed through the RSL: the cross-linking of terms-of-art to paragraphs and their presence in pre-conditions, clauses and other definitions; and the cross-linking of exemptions to modify pre-conditions and clauses.

The RSL parser cross-links definitions to requirements by matching terms-of-art in definitions with phrases in requirements pre-conditions and clauses. Recall from Figure 3 the definitions for terms data storage device (line 4) and facsimile (line 11) and the imported term payment card (line 15) from another law, NV §205.602. The instructions `INCLUDE` (lines 2 and 15) orchestrate these definitions by applying them to all sub-paragraphs in §603A.215, which in turn instructs the parser to link each term to each matching phrase in the pre-conditions and clauses for all requirements contained therein. This includes other definitions, such as the phrase on line 13 that excludes "data storage device" from the onward transmission of a facsimile. Figure 7 illustrates the implication these definitions have on requirements in paragraphs §603A.215(1) and (2): the underlined phrases correspond to those phrases that match the terms-of-art from Figure 3 as determined by the parser.

Both when to apply a prescription and the extent of the prescription can be computationally adjusted by relaxing or tightening definitions using the includes "<" and excludes "~" operators, respectively. For example, if we redefine payment card to exclude gift card, then the scope of when to apply the requirement to comply with the PCI DSS standard (on line 6) would be further restricted to omit the case of gift cards. Alternatively, if data storage device were redefined to include USB drives, then the extent of the prohibition on moving such devices (on line 16) would be extended to include this interpretation. The ability to shape when to apply and the extent of prescriptions using the RSL can enable regulators and businesses to evolve the conditionality of regulations as new technologies emerge over time.

```
1     SECTION 603A.215
2     PAR 1.
3     data collector!
4        & doing business in this State
5        & accepts a payment card in connection with a sale of goods or
           services
6        : shall comply with the current version of the Payment Card
           Industry (PCI) Data Security Standard...
7     PAR 2.
8     data collector!
9        & doing business in this State
10       EXCEPT 1.
11    PAR (a)
12       & does not use encryption to ensure the security of electronic
           transmission
13       : shall not transfer any personal information through an
           electronic, non-voice transmission other than a facsimile to
           a person outside of the secure system of the data collector
14    PAR (b)
15       & does not use encryption to ensure the security of the
           information
16       : shall not move any data storage device containing personal
           information beyond the logical or physical controls of the
           data collector or its data storage contractor
```

Figure 7. Excerpt from Nevada §603A.215(1) and (2)

Whereas definitions shape terms used in pre-conditions and clauses of requirements, exemptions fine-tune what is excluded from pre-conditions and clauses. Figure 8 shows a description of the role "telecommunications provider" with a role constraint on line 4. The `EXEMPT` keyword instructs the parser to exclude this role and constraint from all rules in §215 and all sub-paragraphs therein. While such an exemption could be stated in a definition using the excludes operator "~", exemptions provide a mechanism to tighten meanings across a document cross-section, unbounded by a single term-of-art or definition.

```
PAR 4.
PAR (a)
telecommunications provider!
   & acting solely in the role of conveying the communications of
     other persons, regardless of the mode of conveyance used,
     including, without limitation (1) optical, wire line and
     wireless facilities; (2) analog transmission; and (3)
     digital subscriber line transmission, voice over Internet
     protocol and other digital transmission technology
   EXEMPT 603A.215 *
```

Figure 8. Excerpt from Nevada §603A.215(4)(a) expressed in RSL

Figure 9 illustrates a high-level architecture for how constraints, expressed as definitions and exemptions, are traced by the parser to requirements. The arrows route constraints through parser instructions as follows: the `INCLUDE EXTERNAL` instruction imports (in purple) the payment card definition from another regulation, NV 205.602, into NV 603A.215(5)(d). The `INCLUDE` instruction maps (in blue) the definitions from 603A.215(5), including any imported definitions, onto 603A.215; this mapping includes the inner link from data storage device to facsimile, and the outer links to requirements in 603A.215(1) and (2). Finally, the exemption from 603A.215(4)(a) is mapped (in red) onto the requirements 603A.215 to specifically exclude interpretations that may be implied by the definitions.

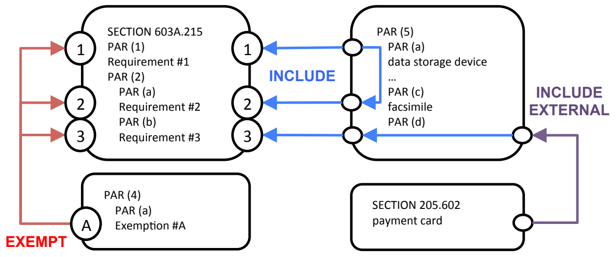

Figure 9. Summarizing the Effects of Conditionality

4.2. Narratives and Conditional Surface Structures

The RSL formalization consists of several constructs that, when considered together, form a regulatory narrative that traverses the prescriptions of regulations, such as who is required or permitted to do what, when and to what extent. The narrative contains a conditional surface structure that indicates the “first-glance” requirements and an extension that contains requirements that follow from this surface. This narrative can be accessed through the RSL-generated model to direct an organization to answer the questions above and guide their compliance process. The surface and extension consists of requirements and relations, which are discussed in Section 4.3 and include goal-oriented (`REFINES`), exceptional (`EXCEPT`), and temporal (`FOLLOWS`) relations.

We now establish three assumptions that govern how we interpret cross-references within the surface and extension. We distinguish between the prescription of an act, which is the declaration that an act is permitted, required or prohibited, from the performance of the act. Figure 10 presents a model of the relations that were used to validate the following assumptions A1-A3 for consistency across the conditional surface structure and extension:

A1. Every refinement y to a requirement x is either a:

a. Sub-process or task, which means that the act of *y* is temporally contained (begins and ends) within the course of performing *x*, and is necessary to achieve, maintain or avoid *x* (the definition of goal by Dardenne et al. [6]);

b. Quality attribute, which elaborates on either the act of *x* or some object (a noun) within the requirement clause for *x*.

A2. At least one act expressed in the pre-condition a must be performed prior to the prescription of *y*.

A3. For any exception *u* to requirement *y*, if *u*'s pre-conditions are satisfied then the *y*'s modality changes to exclusion (e.g., is required changes to is not required);

a. Every refinement *z* to requirement *y* inherits this change to exclusion; and

b. Every post-condition *b* inherits this change to exclusion.

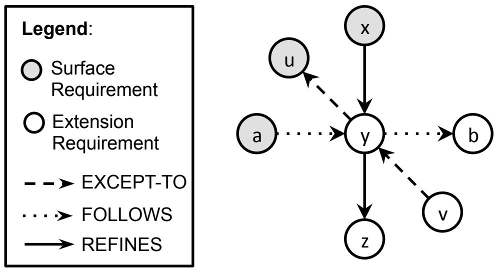

Figure 10. Generic Model of Requirements Relations

We validated the three assumptions by automatically generating and manually inspecting traces (x, y, z) and (a, y) for assumptions A1 and A2, respectively, and for assumption A3, traces (u, y), including traces (y, z) for assumption A3(a) and traces (a, y) and (y, b) for assumption A3(b) and, recursively, all corresponding sub-traces from requirements a, z, and b. In this analysis, we encountered obstacles to the initial design of the RSL, such as delay handling, or seemingly ambiguous relationships between requirements that fit multiple relations. For example, one meaning of a goal refinement (how we achieve a goal) is an act performed in preparation for a future event, which may be realized as either a goal refinement or a pre-condition. These discoveries led us to refine our definitions for relations (e.g., to exclude this case from our definition refinement). Following these revisions and revalidation across the dataset, our analysis indicates that the above assumptions are valid.

We also observed how these assumptions facilitate debugging the specification. Consider Figure 11, in which a requirement may precede a number of others: a delay (VT-4) precedes an investigation conducted by law enforcement, the notification that the investigation is concluded (VT-8) and the act of resuming the initial notification (VT-9). Originally, the delay (VT-4) posed a problem, given that the assumption A3 behind `EXCEPT` removes the obligation (VT-1), but this notification must eventually occur after the delay. We can resolve this issue by implementing a return temporal relation `PRECEDES` to VT-1 so that upon completion of the delay sequence, the notification is resumed. This can be seen in the link from VT-9 to VT-1 in Figure 11.

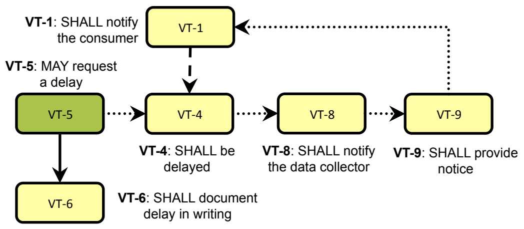
VT-1: SHALL notify the consumer
VT-1
VT-5: MAY request a delay
VT-5
VT-4
VT-8
VT-9
VT-4: SHALL be delayed
VT-8: SHALL notify the data collector
VT-9: SHALL provide notice
VT-6
VT-6: SHALL document delay in writing

Figure 11. Assumption Validity Affects Translation in VT

Extensions with valid assumptions A1-A3 enable the computation of a surface area that consists of the "first-glance" requirements for an organization. The surface area contains the starting point for the requirements narrative, which includes all pre-conditions and high-level goals from which all other prescriptions follow. In Figure 10, the surface area includes all requirements x, u and a and excludes all requirements y, b, v, and z. The surface structure can appear in documents as "centers" in thematic clusters. In Figure 12 for example, surface requirements appear as dark-colored nodes and requirements in the extension are light-colored. Two salient surface requirements describe security program implementation and maintenance and program runtime monitoring, from which personnel and software requirements radiate outward in the extension, respectively. Requirements that are disconnected from the narrative (e.g., unreferenced by other requirements) appear as part of the surface and may indicate needed elaboration to connect these requirements with the overall narrative. As we discuss in Section 5.4, Figure 12 represents a cascading refinement style, which is rare among our observations in this dataset.

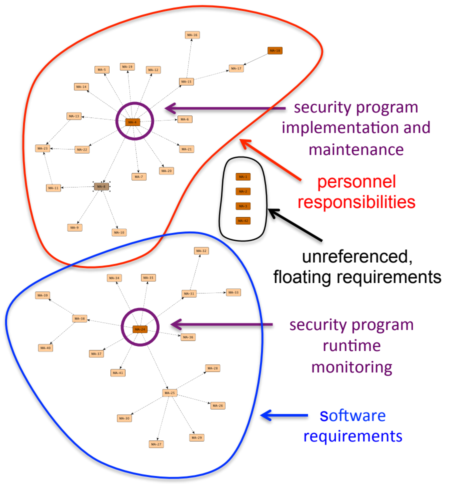
security program implementation and maintenance
personnel responsibilities
unreferenced, floating requirements
security program runtime monitoring
software requirements

Figure 12. Surface Structure with Themed Clusters in MA-S

4.3. Regulatory Specification Patterns

When visualized graphically, the narrative reveals several regulatory specification patterns. These patterns describe mechanisms for prescribing the behavior of personnel and systems in the environment. In Figure 4, we presented the first pattern, called a suspension, in which a permission (AR-10) is an exception to an obligation (AR-7) and satisfying the conditions of the permission causes the obligation to be suspended. We now discuss three other patterns: system design alternatives and scaling restrictions; standards and indemnification; and limited exceptions for legacy systems. We believe these patterns can be re-used in writing new government regulation and industry standards.

Figure 13 shows three system design options for sending written (MD-15), electronic (MD-16) and telephonic (MD-17) notices as means for notifying individuals, data owners and data licensees of a security breach under MD §14.3504(e). These alternatives are intended to allow businesses to leverage a diverse set of contact options based on the level of technological sophistication of the business. In addition, the exception MD-18 permits a substitute notice via statewide media and other broadcast mechanisms, when the cost of notification becomes prohibitive. This type of scaling mechanism (a permitted exception conditioned on measurable limits of effect size, in this case a finite number of notices or monetary value) can be used to control regulatory system costs across an entire industry.

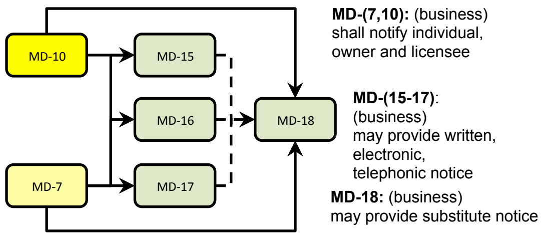
MD-10
MD-7
MD-15
MD-16
MD-17
MD-18
MD-(7,10): (business) shall notify individual, owner and licensee
MD-(15-17): (business) may provide written, electronic, telephonic notice
MD-18: (business) may provide substitute notice

Figure 13. System Design Alternatives and Scaling Restrictions

Figure 14 shows the combined uses of deference to external standards with indemnification from NV §603A.215. The Payment Card Industry (PCI) Data Security Standard, cited in NV-5 in Figure 14, prescribes several technical security requirements for businesses that handle payment cards. In Figure 14, a business is prohibited (in red) from transferring data (NV-6) or moving data storage devices (NV-7), excluding facsimiles. However, complying with the PCI standard (in yellow, NV-5) is an exception that permits transferring data and moving devices. Whether a business chooses to accept the more prohibitive restrictions or to comply with the exception, NV §603A.215 prohibits the business from being liable for data breach damages. This prohibition is an example of a safe harbor, which is a regulatory mechanism designed to encourage industry to act against uncertainty (the uncertain costs of data breach damages vs. the certain costs of PCI-DSS compliance).

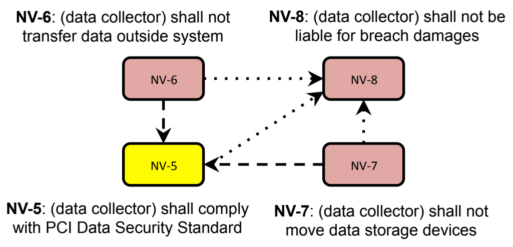
NV-6: (data collector) shall not transfer data outside system
NV-8: (data collector) shall not be liable for breach damages
NV-6
NV-8
NV-5
NV-7
NV-5: (data collector) shall comply with PCI Data Security Standard
NV-7: (data collector) shall not move data storage devices

Figure 14. External Standards and Indemnification

In Figure 15, the State of Vermont describes a set of prohibitions (in red, VT-40 through VT-43) on the use of Social Security numbers (SSNs) of Vermont residents. In the United States, SSNs are issued by the government for tracking government-sponsored pensions, but have over time been used to track individuals for other purposes, such as health benefits and credit-based services, including cellular telephones, utilities, loans and credit cards. Because of the prevalent and historic use of SSNs to authenticate and identify individuals, VT §62.2440(c)(8)(A) includes an exception, which permits (in green) continuous use of SSNs to accommodate legacy systems. Continuous use includes the obligations (in yellow) to notify residents about such use (VT-48) and provide the option to halt such use (VT-49). Such exceptions provide businesses with the ability to scale their business practices to a new standard of care based on individual consumer preferences over time.

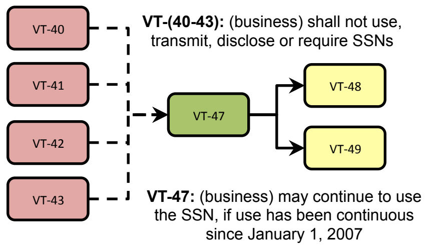
VT-40
VT-41
VT-42
VT-43
VT-(40-43): (business) shall not use, transmit, disclose or require SSNs
VT-47
VT-48
VT-49
VT-47: (business) may continue to use the SSN, if use has been continuous since January 1, 2007

Figure 15. Limited Exceptions for Legacy Systems

As technology evolves, we foresee increasingly invasive regulations that affect system design. Thus, we believe patterns such as these should become part of the requirements nomenclature, to aid business analysts and engineers in understanding the scope and implication of regulations on system design.

4.4. Measuring Styles of Specification

Expressions in the RSL represent stylistic variations in how different state regulators in the U.S. express regulations. Table II presents measures computed from the RSL-generated model for each law studied. The metrics in Table II include: pre-clause continuations, which are conditions that appear in parent paragraphs and apply to rules in sub-paragraphs; post-clause continuations, which are conditions that appear in sub-paragraphs, following a single rule; reference ambiguity, which is the average number of rules per paragraph, computed only for paragraphs that contain rules; and surface area, which is the proportion of rules that are not refinements and have no rules as pre-conditions.

Using these metrics, we observed the following styles:

Cascading Refinement occurs when sections are organized around high-level goals in which goal-refinements and post-conditions are expressed in nested paragraphs. MA-S is organized around two high-level goals (see Figure 12):

businesses must implement a security program and the program must monitor software security. The remainder refines these goals in sub-paragraphs, as evidenced by the high pre-clause continuation rate of 78.6%, no post-clause continuations, and the minimal surface area of 14.3%. The surface area measure typically indicates that requirements are clustered around common themes via relations.

Referential Uniqueness occurs when cross-references refer to the fewest number of requirements, ideally one. MD, despite a higher than average pre-clause continuation rate of 43.8%, exhibits a lower than average reference ambiguity of 1.063 rules per reference: almost to the floor measure of 1.000. Laws with high reference ambiguity in Table II, such as MA, VT, OR and AK, also exhibit higher than average ambiguity loss (see Table I) through operationalized references using the RSL. These measures indicate when readers of laws will need to review fewer or more requirements when resolving cross-reference.

Block Formatting occurs when the paragraphs are rarely nested and each paragraph contains multiple requirements. MA-93H uses lower than average pre- and post-clause continuations, both less than 5.1%, and has a higher than average referential ambiguity at 4.100 rules per reference. This style is in contrast to cascading refinement and referential uniqueness and indicates a law that requires extra time to formalize relations between requirements.

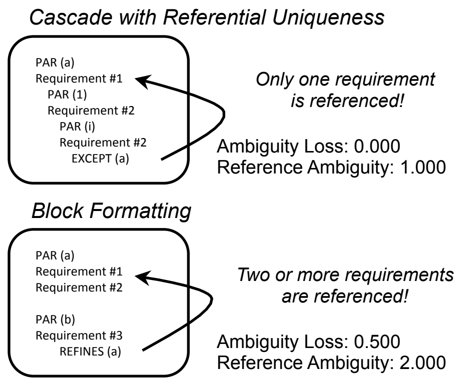

Figure 16. Comparing Cascade and Block Formats

Our experience in codifying these regulations is preferential to cascading refinement and referential uniqueness, since these styles are logically supported by our narrative structure and yield more precise cross-references.

TABLE II. METRICS FOR SPECIFICATION STYLES

State Law	Continuations		Reference Ambiguity	Surface Area
	Pre-clause	Post-clause		
AK	0.203	0.241	1.571	0.253
AR	0.316	0.474	1.118	0.421
MA	0.051	0.051	4.100	0.231
MA-S	0.786	0.000	1.676	0.143
MD	0.438	0.125	1.063	0.281
NV	0.269	0.038	1.130	0.346
OR	0.404	0.163	1.873	0.279
VT	0.305	0.190	1.966	0.352
WI	0.000	0.167	1.500	0.278

5 Threats to Validity

In grounded analysis, multiple analysts derive theoretical constructs from a dataset to describe or explain the data and the constructs are assumed to only generalize to that dataset [10]. Recall from Section 4 that we selected regulations that share a theme (data breach notification), thus our theory may not be externally valid in other regulated domains, such as medical devices or aviation, which may require new language constructs. However, to challenge our assumptions, we validated the schema notation and document model by visually inspecting data breach notification laws in all 46 U.S. states and territories, two U.S. Federal regulations (HIPAA Privacy Rule and Access Standards), the European Union Directive 95/46/EC and a Canadian law (PIPEDA). We found the schema sufficiently robust to model these regulatory documents and express their cross-references.

Construct validity is the correctness of operational measures used to collect data, build theory and report findings [24]. To improve construct validity, we maintained a caveats list of translation strategies that reflect unusual cases and how the parser should treat such cases, and a proposed changes list of requirements with examples for new language constructs. As a new construct was introduced into the language, we reviewed each law to update the translation to reflect the new construct to ensure consistency across the translated datasets. In addition, we developed analytic tools using the parser and a research database to collect all the statistics reported in this paper.

Internal validity is the extent to which measured variables cause observable effects within the data [24]. Our results show that writing styles can positively or negatively impact reference ambiguity and ambiguity loss, as measured by our RSL translation presented in Table II. We found these ambiguities can lead to cycles within the same relation class and that consistently reducing the use of cross-referencing, in general, increases the surface area. In Section 4.3, we introduce several assumptions regarding the effects of typed cross-references. Our research introduces these assumptions for the first time: while we recognize they may not be internally valid across repeated measures of other datasets, we believe establishing these assumptions now allows us and others to probe deeper into the challenges and limits of canonicalization and formalization of legal requirements.

Reliability describes the consistency of the theory to describe or explain environmental phenomena over repeated observations [24]. To improve reliability, both investigators (the authors) separately translated the datasets into the RSL and compared their results afterwards to identify alternate modes of expression and language caveats.

6 Discussion and Summary

In this paper, we introduce a requirements specification language (RSL) for codifying legal requirements with typed cross-references. In Section 4, we show how the RSL can be used to shape conditionality of regulatory coverage, express regulatory narratives and conditional surface structures, make regulatory requirement patterns visually salient and measure different styles of regulatory document construction. These RSL capabilities provide document authors with new tools to design and debug specifications, to remove ambiguity and organize requirements around central themes. The RSL's ability to reuse and extend definitions and link to regulatory rules across multiple regulations supports our vision of requirements as open, dynamically evolving systems wherein the discovery of conflicts becomes increasingly critical to creating regulatory harmony. Finally, the RSL parser supports several features that can be used to "debug" regulatory specifications, by identifying cycles in cross-references, definitions for terms not used in the regulation, and possible conflicts or contradictions through visual inspection of the generated graphs. We found the time required to translate the regulations into the RSL well worth the ability to debug and analyze the relations using the RSL-generated model. While regulations were not originally written for this type of technical analysis, we believe our analysis can be used to improve the construction of these documents to reach a broader, more participatory audience throughout industry and academia by allowing participation to focus on alternative regulatory structures and the logical implications of those structures.

Acknowledgment

This research was supported by the U.S. Department of Homeland Security under Grant Award Number 2006-CS-001-000001, under the auspices of the Institute for Information Infrastructure Protection (I3P) research program.

References

[1] D. Bourcier, P. Mazzega, "Toward measures of complexity in legal systems." Int'l Conf. AI & Law, 2007, pp. 211-215.

[2] T.D. Breaux, A.I. Anton, J. Doyle, "Semantic parameterization: a process for modeling domain descriptions."ACM Trans. Soft. Engr. Method., 18(2): 5, 2008.

[3] T.D. Breaux, M.W. Vail, A.I. Antón. "Towards compliance: extracting rights and obligations to align requirements with regulations."IEEE 14th Int'lReq'tsEngr.Conf., 2006, pp. 49-58.

[4] T.D. Breaux. “Exercising due diligence in legal requirements acquisition: a tool-supported, frame-based approach.”IEEE 17th Int’l Req’ts Engr. Conf., 2009, pp. 225-230.

[5] T.D. Breaux, Legal requirements acquisition for the specification of legally compliance informaiton systems, North Carolina State Univetsity, Ph.D. thesis, 2009.

[6] A.. Dardenne, S. Fickas, A. van Lamsweerde. “Goal–directed requirements acquisition," Sci. Comp. Prog., 20:3-50, 1993.

[7] M.D. Fraser, K. Kumar, V.K. Vaishnavi, “Informal and formal requirements specification languages: bridging the gap.” IEEE Trans. Soft. Engr., 17(5):454-466, 1991.

[8] A. Fuxman, L. Liu, J. Mylopoulos, M. Pistore, M. Roveri, P. Traverso. “Specifying and analyzing early requirements in Tropos.” Req’ts Engr. Journal, 9(2): 132-150, 2004.

[9] P. Giorgini, F. Massacci, J. Mylopoulos, N. Zannone. “Modeling security requirements through ownership, permissions and delegation.”IEEE 13th Int’l Req’ts Engr. Conf., 2005, pp. 167-176.

[10] B. Glaser, A. Strauss. The Discovery of Grounded Theory: Strategies for Qualitative Research. Aldine Transaction, 1967.

[11] M. Glinz, S. Berner, S. Joos. “Object-oriented modeling with ADORA.”Info. Sys. 27: 425-444, 2002.

[12] P. Laurent, J. Cleland-Huang, C. Duan, “Towards automated requirements triage.” IEEE 15th Int’l Req’ts Engr. Conf., 2007, pp. 131-140.

[13] M. Lauritsen, T.F. Gordon, “Toward a general theory of document modeling.”Int’l Conf. AI & Law, 2009, 202-211.

[14] A.A. Levene, G.P. Mullery, “An investigation of requirement specification languages: theory and practice.”IEEE Computer, 15(5):50-59, 1982.

[15] S. Liaskos, S.A. McIlraith, S. Sohrabi, J. Mylopoulos. “Integrating preferences into goal models for requirements engineering.” IEEE 18th Int’l Req’ts Engr. Conf., 2010, pp. 135-144.

[16] A.K. Massey, A.I. Anton, “Triage for legal requirements,” NCSU Technical Report #TR-2010-22, October 11, 2010.

[17] J. Maxwell, A.I. Anton, “Developing production rule models to aid in acquiring requirements from legal texts.” IEEE 17th Int’l Req’ts Engr. Conf., 2009, pp. 101-110.

[18] J. Maxwell, A.I. Anton, “Discoverying conflicting software requirements by analyzing legal cross-references,” In Submission: ACM/IEEE Int’l Soft. Engr. Conf., 2011.

[19] J. Martinek, J. Cybulka, “Dynamics of legal provisions and its representation.” Int’l Conf. AI & Law, 2005, pp. 20-24.

[20] J. Mylopoulos, A. Borgida, M. Jarke, M. Koubarakis. "Telos: representing knowledge about information systems," ACM Trans. on Info. Sys., 8(4):325-362, 1990.

[21] S. Romanosky, R. Telang, A. Acquisti. "Do data breach disclosure laws reduce identity theft?" Workshop on the Economics of Information Security (WEIS), June 25-28, 2008.

[22] I. Rubinstein, “Privacy and Regulatory Innovation: Moving Beyond Voluntary Codes.” (In Press) I/S: A Journal of Law and Policy for the Information Society, April, 2011.

[23] R. Winkels, A. Boer, E. de Maat, T. van Engers, M. Breebaart, H. Melger. “Constructing a semantic network for legal content,” Int’l Conf. AI & Law, 2005, pp. 125-132.

[24] R.K. Yin. Case study research, 4th ed. In Applied Social Research Methods Series, v.5. Sage Publications, 2008.

[25] E. Yu. “Modeling organizations for information systems requirements engineering.” Int’l Symp. Req’ts Engr., 1993, pp. 34-41.