Impalpable Constraints: Framing Requirements for Formal Methods

Travis D. Breaux and Annie I. Antón
North Carolina State University
{tdbreaux, aianton}@ncsu.edu

Abstract

Regulated software systems require a precise and unambiguous system specification that strictly conforms to the intent of policies and regulations. Formal methods for verification and validation can be used to show that specifications are consistent and complete. However, small and medium sized projects often lack access to the expertise and training required to apply such methods. Towards improving access to formal methods, we introduce a frame-based approach to model requirements that aligns a configurable natural language representation of requirements with a corresponding first-order predicate logic model. The natural language representation supports domain experts and other stakeholders who create, modify or interpret regulatory requirements, whereas the logic model supports machine manipulation of these requirements for automated reasoning. In this paper, we present:1) the frame-based theory; 2) a case study design and associated metrics for evaluating different models in this theory; 3) the results of a case study in the domains of aviation, healthcare and privacy; 4) constraint patterns that were acquired using this approach and that can be re-used to increase consistency in stating requirements; and 5) a prototype that implements the frame-based theory for specification of regulatory requirements.

1. Introduction

Requirements engineers must transcribe conceptual phenomena that describe the environment into natural language statements that designers and developers can use to build systems [14, 26]. As a communication medium, natural language provides a range of expression that allows stakeholders to share ideas that are largely unrefined or under-specified. This allows stakeholders to broadly envision their understanding of a system without overburdening themselves with excessive details early on. The quality of requirements depends upon removing ambiguities and resolving conflicts [2]. However, the freedom of natural language that benefits the early development phases can negatively impact requirements quality as the number of project requirements and stakeholders increase. To illustrate, consider the following four security requirements from the U.S. Federal HIPAA[1] Privacy Rule [25], labeled A-D, below:

(A) The covered entity may disclose patient information to a foreign government entity for public health activities.
(B) The covered entity, who has an agreement with the patient to restrict disclosures, may not disclose patient information.
(C) The health care provider may disclose patient information to an employer for public health activities.
(D) Patient information may be disclosed to a foreign government entity by a covered entity for public health activities.

Each of these requirements affects the design of patient information systems in the U.S. We identified over 300 such requirements in the HIPAA Privacy Rule and these do not account for the various other potentially conflicting requirements found in state laws. Adept requirements engineers will identify three issues: (1) requirement D is redundant because it simply re-topicalizes requirement A (e.g., the topic is *patient information*, instead of *the covered entity*); (2) requirement B is a potential conflict with requirements A, C, and D; and (3) requirements A, C and D all share the same purpose: for public health activities.

We have shown that the Semantic Parameterization process yields logical expressions that a machine can use to automatically reason about and detect these three problems [3, 4, 6]. This process requires engineers to have expertise and training to consistently map each requirement into first-order logic; for small development teams with limited budgets, this expectation is too costly. According to a 2006 Ernst & Young survey of nearly 1200 corporate and IT executives, over half said compliance with regulations is the primary driver of information security from 2005-2007 [8]. To support this continuing need, developers and regulators need formal tools that are

[1] U.S. Public Law 104-191, 110 Stat. (1996)

accessible and that provide increased assurance that systems conform to the law.

To this end, we introduce a frame-based theory that formally describes *variability* and *context-sensitivity* in NL requirements. Variability concerns alternative ways to specify formal requirements models whereas context-sensitivity affects our ability map between natural language requirements and formal models using parsers and generators. Requirements engineers can apply this theory to scenarios and requirements to acquire consistent and expressive constraint patterns using our declarative, frame-based markup. We apply this theory in two applications: (1) a tool suite used to conduct case studies with the markup; and (2) a prototype NL interface to specify formal requirements. We used the tool suite to conduct the case studies that are presented in this paper.

This paper is organized as follows: Section 2 discusses related work; Section 3 introduces the formal frame-based theory and metrics for evaluating correctness; Section 4 discusses the case study design; the resulting validation from an exploratory study appears in Section 5; Section 6 discusses theoretical and practical contributions, including the constraint patterns and our prototype; and Section 7 concludes with plans for future work.

2. Related Work

Frames were proposed in the late 1960s and mid-70s as a linguistic and conceptual structure to model knowledge about the world [10, 18, 23]. In general, a *frame* corresponds to a concept that has one or more slots; each *slot* describes a stereotypical property of that concept. Slots are assigned an atomic value or another frame, called a sub-frame. Sometimes, slots have default values. Frames correspond naturally to *objects* in object-oriented programming and *patterns* or *templates* in requirements engineering. In this paper, a pattern is a frame with one or more sub-frames. Templates are "shallow" frames because slots in templates are only assigned atomic values; thus they never contain sub-frames.

In requirements engineering, frames, patterns and templates have been employed to formalize constraints on requirements. Fillmore's *case frames* or *case roles* specifically model the properties of actions [10], such as the *actor* who performs the action or the *object* upon which the action is performed. Case frames have been used to model scenarios [20] and goals [4, 17], the latter of which are limited to simple action statements. In addition to case frames, other approaches have employed patterns or templates to model deontic [3, 4] and temporal constraints [16, 24]. Breaux et al. identified patterns to formalize permissions and obligations [3, 4] and proposed a template-based method to generate a controlled subset of NL [4]. The more expressive frame-based approach in this paper extends that work. Konrad et al. [16] and Smith et al. [7, 24] employ patterns and templates, respectively, to align temporal constraints with NL.

While the frame-based theory in this paper exposes case frames, deontic and temporal constraints in NL texts, we focus on its ability to identify new domain-specific constraints on requirements. In this paper, we show how the theory is used to demarcate phrases in NL documents that contain requirements, in which some phrases represent concepts (frames) and other phrases link these concepts together as roles (slots). Similar to Reubenstein et. al [22], we also align our declarative frame-based theory with a denotational semantics for generating logical expressions.

In natural language processing, work to identify a generalized formal semantics for NL includes case grammars [10], transformational grammars [15] and phrase-structure grammars [12]. These grammars are driven by explicit theories of NL syntax and are intended to express a broader scope of NL than is required for requirements. Despite any correlation between syntax and semantics, we rely instead on human judgment to demarcate phrases in a statement based on the reader's comprehension of semantic relationships between phrases. Our approach is light-weight, by design; it does not require extensive linguistic knowledge or elaborate grammar rules.

3. Frame-based Requirements Model

The frame-based requirements model describes a controlled set of NL requirements. We illustrate the model using a declarative markup that is applied to requirement (B) in Section 1. Each frame is a word or phrase enclosed in brackets. Curly brackets denote optional frames that can be removed from a source sentence without making the sentence grammatically incorrect; otherwise the frame is required and denoted using square brackets. Optional frames are used to elaborate or refine knowledge expressed in a requirement statement. Each frame has a sequence of one or more slots comprised of words and sub-frames. For example, in Figure 1 the frame "[restrict [disclosures]]" from line 6 contains two slots: the word "restrict" and the sub-frame "[disclosures]."

The case frame in Figure 1 is comprised of the actor (covered entity) on line 2, the modal-action frame (may) on line 10, the act (disclose) on line 11 and the object of the act (information) on line 12. The spacing is used to denote association: for example, the agreement has two optional sub-frames "{with…}" and "{to…}" that elaborate or refine knowledge about the agreement. The markup language has an associated parser based on the context-free grammar in Appendix A. The parser generates programmable frames based on the syntactic theory presented in Section 3.1. After presenting the formal model in detail, we introduce metrics for checking consistency and correctness in a frame-based model.

```
1   [
2       [[the] {covered} entity
3           {who [has
4               [[an] agreement
5                   {with [[the] patient]}
6                   {to [restrict [disclosures]]}
7               ]
8           ]}
9       ]
10      [may {not}
11          [disclose
12              [{patient} information]
13          ]
14      ]
15  ]
```

Figure 1: Requirement with Frame Markup

3.1. Syntactic Theory

A frame-based requirements model satisfies two separate theories that describe a frame's syntactic and semantic features. The syntactic features concern how NL statements are composed from words using frames, whereas the semantic features concern how logical expressions are generated from a composition of frames using a denotational semantics. The term *model* refers to an instance of this theory; engineers can construct different models of the same requirements using the theory. This paper presents the syntactic theory in detail, while the semantic theory will be addressed in a follow up paper. The syntactic theory is comprised of a frame, composition and model as formally defined below.

Definition 1: A *frame* describes a NL phrase or sentence through a finite sequence of *n* slots $\langle s_1, s_2, \ldots, s_n\rangle$ in which each slot represents a fixed or variable portion of the phrase. For each slot *s* in a frame, we define the following three functions: *label* (*s*) that maps *s* to a finite string of characters that describe the slot; *variable* (*s*) that maps *s* to a possibly empty set of frames, called a *variable*; and *required* (*s*) that is true only if slot *s* is required for grammatical correctness. In Figure 1, for example, removing the required frames "the" or "an" would yield a grammatically incorrect sentence. Each frame is a member of one or more variables. A principal variable *T*, called the *top variable*, contains only frames that correspond to whole sentences.

Definition 2: A *composition* is an interpretation of one or more frames that yields a single sentence through a set of selections over variables. In any phrase, English conjunctions (and, or) map directly to the set of selections using disjunctive normal form: the set contains *alternate* selections in which each selection is a conjunction of frames. A composition is the pair $\langle f, \sigma\rangle$ that consists of a top frame $f \in T$ corresponding to a sentence and a selection function σ that maps each slot *s* to a subset of the permutations of *variable* (*s*). If the slot variable is empty, σ maps that slot to the empty set.

For example, Figure 1 has a corresponding composition in which a top frame in *T* is bounded by the open and closing square brackets on lines 1 and 15, respectively. The top frame has exactly two slots, each with a separate variable whose selection corresponds to lines 2-9 and 10-14, respectively. The first slot variable is selected for the "covered entity" frame on line 2, which is just one of many possible *actors* who may not disclose patient information. The selection function σ maps that first slot variable to the set containing a singleton set that contains only the "covered entity" frame. Natural language statements are generated by performing an in-order traversal of a composition; special syntactic consideration is given to include correct punctuation and handle logical connectives.

Definition 3: A *model* is a set of frames that describe a controlled set of NL requirements. Models differ for several reasons, most notably because the markup can be applied in different ways to the same text. These primarily differences affect the degree of variability, which is logarithmically proportional to the number of possible statements a model can generate. For example, consider statement S_0, below:

S_0: `[may {not} [disclose [{patient} information]]]`

The model derived from S_0 will generate the following four phrases, depending on whether an optional slot (in **bold**) is included:

1. may disclose information
2. may disclose **patient** information
3. may **not** disclose information
4. may **not** disclose **patient** information

Frame models derived from markup are initially limited in their expressive power because each slot variable has only a few alternative frames from which to choose. Variables that appear in recurrent frames can be unified to increase a model's expressiveness. Moreover, our experiences to date suggest that building models from successive case studies will also make them more expressive.

3.2. Technical Correctness

Context-free grammars for natural language are evaluated for their generative qualities including generality, selectivity and understandability – factors that affect the variety and grammatical correctness of statements that are generated by the grammar [1]. In addition to these, we employ three metrics that compare two frame models by appealing to the semantics of the frame theory. The metrics assume that both models were derived from the same sample text. We evaluate a candidate frame in one model by comparing it to another frame, called an *oracle*, which is assumed to be correct by definition. The following metrics have been implemented in our tool suite to remove inter-rater bias:

Internal Demarcation. If a word or phrase in the oracle frame is contained in a sub-frame but the candidate frame does not demarcate the word or phrase as a separate sub-frame, this is an *internal demarcation error*. Consider the following frames:

```
O₁: [ The {covered} entity ]
C₁: [ The covered entity ]
```

The oracle frame O_1 contains the additional sub-frame "{covered}" that is not demarcated in the candidate frame C_1; thus the absence of this sub-frame is scored as an internal demarcation error.

Logical Demarcation. If the English conjunctions (and, or) in the oracle frame are replaced by the corresponding logical operators but not in the candidate frame, this is a *logical demarcation error*. Furthermore, it is an error to map a frame to a logical conjunction in the oracle frame and to a disjunction in the candidate frame, and vice versa. Consider:

```
O₂: [ obtain | inspect [copies] ]
C₂: [ obtain {and inspect} [copies] ]
```

The oracle frame O_2 maps the English conjunction "and" to a logical disjunction "|", whereas the candidate frame C_2 incorrectly offsets the phrase "and inspect" as an optional sub-frame. These errors are commonly due to ambiguities in English conjunctions; the conjunction "and" can mean either logical-or or logical-and depending on the intended interpretation.

Dissociation. If a phrase in the oracle frame is demarcated in two disjoint candidate frames (e.g., one frame is not the ancestor of the other), then this is scored as a *dissociation error*. For example, the frames O_3 and C_3 follow from the phrase "Patients generally should request corrections…"

```
O₃: [should [request [corrections] ] {if [...]} ]
C₃: [should [request [corrections {if [...]}] ] ]
```

The condition frame "{if […]}" in O_3 applies to the recommendation frame "[should…]"; that is, "the actor should request, if…" The candidate frame C_3, however, associates the condition as an optional slot in the sub-frame "[corrections …]" The association in C_3 means "to correct, if…" because the condition applies to the act of corrections, not to the recommendation to request corrections, as in O_3. Thus this difference is scored as a dissociation error. Logical demarcation errors due to optional sub-frames (see O_2 and C_2) are one other source of dissociation errors. A third source is context sensitivity, which we now discuss.

3.3. Context Sensitivity

Context sensitivity in natural language produces two types of dissociation errors in the context-free markup. These errors require that phrases be copied or moved between frames to ensure the markup is context-free. Three operations can be applied to a frame: *cut*, *copy* and *paste*, denoted by the operators \, / and *, respectively, and followed by a frame number. In the cut and copy operations, the frame number is assigned to that frame. In a paste operation, the frame number identifies the frame from a cut or copy operation. Consider Figure 2 in which lines 4-10 describe a right that is held by an individual.

The first type of dissociation error concerns the copy of patient information (lines 6-8) that is the object of both the verb *inspect* and *obtain*. However, due to the logical disjunction (line 5), a context-free markup only associates the copy as the object of the verb *obtain*. To correct this error, we copy (/1) the frame on lines 6-8 and paste (*1) it into the slot on line 5. The second type of dissociation error concerns the HIPAA exception for psychotherapy notes (line 12-14) that is applied to the patient information (line 7). However, by appearing after the temporal constraint (line 11), a context-free markup cannot properly associate the exception with the patient information. To correct this error, we cut (\2) the frame on lines 12-14 and paste (*2) it into the slot on line 7. All paste operations into the same frame are logically conjunctive.

```
1   [
2      [[the] individual]
3      [has
4         [[a] right
5            [to [inspect [*1] | obtain
6               [/1 [a] copy {of
7                  [{patient} information {*2}]
8               }]
9            ]
10        ]
11        {for [as [long [as…]]]}
12        {\2 except [for
13           [{psychotherapy} notes]
14        ]}
15     ]
16  ]
```

Figure 2: Resolving Context-sensitivity

A third type of dissociation error is due to context sensitivity concerns words that perform an *anaphoric* function [1] and affect how extensional knowledge is expressed in natural language. For example, the determiner "the" and the pronoun "it" refers to an entity that has been previously identified in a statement or broader context. We use a fourth operator (^), called *note*, to identify when these extensional references are shared between frames. The formal semantics for note operations are not realized in the syntactic theory, but are later used when generating logical expressions. In our case study (see Section 5), these four operations were sufficient to reduce all the context-sensitive phrases to context-free.

4. Case Study Design

The frame-based requirements model was validated in two studies using a single case study design. We present this design so that others can employ the design in future evaluations of similar frame theories. We provide an overview of the design before discussing each step in detail. In the study:

1. Each participating analyst is given a standard set of instructions[2] and examples that define the frame model and explain how to apply the markup from Appendix A to a sample text.
2. The markup text is parsed by the tool to acquire the frames identified by the participant and then the parsed frames are organized using a SORT algorithm.
3. Technical errors are identified in the SORT results using the three metrics from Section 3.2 for evaluating frame models.
4. After the technical errors are corrected by the participant, the participant unifies the frame variables using the new SORT results from the corrected markup.

In addition to providing an informal definition of a frame, the instructions include four rules for applying the markup:

(1) Each frame describes a phrase that can be replaced with an alternate phrase to yield a new grammatically correct sentence; many of these frames have associated concept names.

(2) Frames are optional if they can be removed from a phrase or sentence without making the sentence grammatically incorrect. Example 1 illustrates the use of concept names and an optional beneficiary frame:

Example 1.

$$[\overbrace{\text{[The agency]}}^{\textit{actor}}\text{ may }\overbrace{\text{[disclose information }\underbrace{\{\text{to third-parties}\}}_{\textit{beneficiary}}]}^{\textit{action}}].$$

(3) Frames that comprise lists are separated by logical connectives "&" for conjunction and "|" disjunction. In natural language, these lists are joined by zero or more commas and semi-colons followed by an English conjunction (and, or). English conjunctions are ambiguous: the word "and" does not always mean logical-and, but sometimes it means logical-or. Example 2 shows logical disjunction; logical conjunction is symmetric.

Example 2.

$$\underbrace{\text{doctors, physicians or nurses}}_{\textit{comma-separated disjunction}}\quad\underbrace{\text{[doctors | physicians | nurses]}}_{\textit{correct transcription}}$$

(4) Frames are context-free, meaning a frame that depends on another frame must be a sub-frame in that other frame. These dependencies are limited to the intensional meaning or the concepts expressed between frames, and not the extensional meaning or the actual entities being described. Example 3 shows that because the beneficiary depends on the action "disclose," the beneficiary frame should be a sub-frame of the action frame (as illustrated in Example 1). The participants may use the cut, copy and paste operations to remove context-sensitivity.

Example 3.

$$\dots\underbrace{\text{[disclose information]}}_{\textit{action}}\,\underbrace{\{\text{to third-parties}\}}_{\textit{beneficiary}}\dots$$

After participants markup the sample text, they use a parser to check for missing brackets: each open bracket should have a corresponding closing bracket. The process of identifying missing brackets may cause a frame to become associated with a different context, thus it is critical that the participant analyst be the person to perform this corrective step.

The SORT algorithm sorts frames into partitions based on the following equivalence rule: two frames are equivalent if both frames have the same number of required slots. For each of these slots in the frame, the corresponding pair of slots in both frames are equivalent, if either both slots are variables; or both slot labels are the same strings. The assumption is that each frame in a partition is a candidate for membership in the same frame variable.

[2] http://www4.ncsu.edu/~tdbreaux/frm-instruct.pdf

The SORT results are reported in record sets (see Figure 3), one record for each partition. Each record contains a description of the partition frame and, for each candidate frame, the descriptions of both the candidate frame and the *context* frame that contains the candidate as one of its slots. The partition is sorted by context to assist participants in identifying shared variables by comparing context descriptions. Figure 3 shows a partial record from the HIPAA study in Section 5. The record describes the frame "to [0]" where the "0" represents the location of a required slot, shared between all candidate frames.

Frame: to [0]	
Candidate Frames	**Context Frames**
to [[the] records]	access [to [0]]
to [[the] information]	access [to [0]]
to [[the] entity]	accessible {to [0]} {on-site}
to [[the] summary]	agrees [to [0]]
to [inspect [[a] copy]]	request [access] {to [0]}
to [take [[an] action]]	unable [to [0]]

Figure 3: Example Record Generated by SORT

Because markup is applied to the sample text using an unrestricted text editor, it is possible for the participant analysts to alter the sample text to improve the likelihood that frames will be assigned to the same partition during SORT. To counteract this threat to validity, we apply the diff algorithm [13] to compare the modified text (with markup removed) to the original sample text. This test is performed before SORT and after the completion of the markup and any corrections motivated by fixing unbalanced brackets.

Next, the analyst grades their results to identify technical errors using the metrics from Section 3.2. Because analysts generally evolve their own markup conventions as they see new phrases in a single study, they can often identify their own inconsistencies in the SORT results using these metrics. By marking these errors, analysts can return to the markup text and make corrections to improve their SORT results. A higher number of frames and increased convergence of frames into the same partitions generally indicates an improvement in the model.

After correcting technical errors, the participating analyst identifies which frames share a common variable, a process called unification. Because frames correspond to phrases, unifying two variables means that all phrases represented by the unified variable are valid alternatives or sub-phrases in any frame that contains either of the two variables. For example, in Figure 3, the slots in the first and fourth contexts "access [to [0]]" and "agrees [to [0]]" that contain the frame "to [0]" both describe the object of the access and the agreement, respectively. Unifying the two slot variables for these frames allows composing new statements, such as "access to the summary" and "agrees to the records." However, the other candidate frames do not share the same semantics for this slot. For example, the fifth context (request access) uses this slot to define the purpose of the request (e.g., to inspect) and not the object (e.g., the access) as before.

The SORT algorithm eases the unification process by organizing frames into comparable sets, but it is not a complete solution. Rather, unification is an iterative activity and the analyst may over-unify slot variables based on observations that are limited to a single text. As a future verification step, we intend to generate exemplary statements from a frame model and test these using grammar checkers to identify cases of over-unification. Like software testing, however, the state-space limits ensuring complete test coverage for an entire frame model.

5. Validation Results

The case study design was first applied in an exploratory study, as defined by Perry et al [21]. In the exploratory study, one analyst applied a single frame model to identify and compare regulatory constraints in the domains of aviation, health care and privacy. The data from this study was used to generalize constraint patterns that we present later in this section. The following U.S. federal and state regulations were analyzed in this study:

1. **ETOPS**: §121.374 Continuous Airworthiness Maintenance Program for Two-Engine Planes; in the Extended Operations of Multi-Engine Airplanes regulation [9].
2. **HIPAA**: §164.524 Access of Patients to Health Information; in the Privacy Rule of the Health Information Portability and Accountability Act [25].
3. **MGDPA**: Section 3, Access to Government Data, Chapter 13 of the Minnesota State Statutory Rules [19].

The ETOPS document describes procedures that must be in place to certify two-engine planes before they fly routes longer than one hour. The HIPAA and MGDPA documents both concern information privacy. Whereas the HIPAA study is specific to healthcare information in industry, the MGDPA study generally concerns information for the Minnesota state government. The ETOPS, HIPAA and MGDPA studies required 1.9, 3.3 and 4.4 hours to complete, respectively. Table 1 compares these studies by the number of: words in each text, SORT partitions, total frames and top-level frames, total variables and unified variables, and context-sensitive (C-S) cuts, copies and pastes.

Table 1: Case Study in Three Domains

Property	ETOPS	HIPAA	MGDPA
Words	1323	1727	1858
Partitions	457	374	514
Frames	1400	1822	1928
– Top-Level	46	29	52
Variables	758	649	1038
– Unified	213	117	211
C-S Cuts	15	1	4
C-S Copies	21	11	32
C-S Pastes	26	12	64

In each study, the total number of frames comprised the total number of case frames plus one frame for each word in the text. Because the number of words dominates the number of frames, there is a linear correlation in the number of words, partitions, frames and variables. The numbers in the remaining properties do not share this correlation, however, which we discuss in the following sub-sections in the context of top-level frames and context-sensitivity.

5.1. Top-level Frames

In both the ETOPS and HIPAA studies, the top-level frames solely used patterns that were classified as rules. Each rule is either a stakeholder action that is permitted, called a *right*, or a required action, called an *obligation* [6]. In both studies, we observed a hierarchical paragraph structure, commonly used in U.S. federal regulations, and a specialized refinement pattern (see Figure 4). The refinement pattern provides

```
1   [
2      [[the] {covered} entity]
3      [must
4         [act {on [[a] request]}]
5         {as [follows
6            [
7               {if [...]}
8               [it]
9               [must [...]]
10           |  {if [...]}
11              [it]
12              [must [...]]
13           ]
14        ]}
15     ]
16  ]
```

Figure 4: Frame-based Refinement Constraint in HIPAA §164.524

a syntactic device to directly associate alternatives (other specialized rights or obligations) with a general or abstract rule. For example, in Figure 4 below, the general rule "must act on a request" appears in lines 2-4 followed by the refinement pattern "{as follows […]}" in lines 5-14. The alternative refinements, on lines 7-9 and 10-12 are separated by a logical disjunction (vertical bar) at the start of line 10.

Legal professionals refer to a *term-of-art* as "a word or phrase having a specific, precise meaning in a given specialty, apart from its general meaning in ordinary contexts" [11]. In U.S. Federal regulations, each term-of-art has a definition that includes alternative names for the term and case roles that relate the term to associated actions that use the term in practice. In HIPAA, the term "covered entity" is defined both by its specializations, such as "health care provider" or "health plan," and also by roles that the entity is involved in, such as treating patients or billing individuals for services. Eight of the 52 top-level frames in the MGDPA study were definitions. In contrast, the ETOPS and HIPAA texts, consistent with other federal regulations, organize these definitions in a separate section from the rules.

Because the top-level frames were so consistent between studies (e.g., all were rules or definitions modeled by case frames), we developed a partially automated procedure for unification, by recursively: (1) applying the SORT algorithm to the top variable; frames that were equivalent had their slot variables unified; and (2) repeating step (1) on each unified variable. The procedure terminates when all equivalent frames have had their variables unified. For the three studies, the procedure unified between 18-28% of all variables. We believe more sophisticated approaches will yield higher percentages, thus increasing the re-usability of a single frame-based model with other texts in the same domain.

5.2. Context-Sensitive Differences

The number of context-sensitive operations that were required to derive the different frame models is noteworthy. The ETOPS, HIPAA and MGDPA studies each incurred a unique operation, on average, once every 38, 151 and 30 frames, respectively. The higher frequency in ETOPS and MGDPA is due to linguistic conventions used in these documents. In ETOPS, an introductory rule refers to a list of objects that were distributed as section headers; thus organizing subsequent rules by using the refinement pattern similar to Figure 4. This stylistic device required 11 cut and paste operations to reconstruct the introductory rule in a context-free markup.

The larger number of context-sensitive operations in the MGDPA study, however, is partly due to undeclared terms-of-art. Consider Figure 5, in which the concept of *intellectual property* is elaborated in lines 6-19. We assume the frames for "{entire}" and "{developed [with…]}" would be copied into the slots "{*1}" and "{*2}", respectively.

```
1   [data
2      [that
3         [has
4            [{commercial} value]
5         & is
6            [[a] {substantial & discrete} portion
7               {of [[an] {/1 entire} formula {*2}
8                  | {*1} pattern {*2}
9                  | {*1} compilation {*2}
10                 | {*1} program {*2}
11                 | {*1} device {*2}
12                 | {*1} method {*2}
13                 | {*1} technique {*2}
14                 | {*1} process {*2}
15                 | {*1} database {*2}
16                 | {*1} system
17                    {/2 developed [with …]}
18              ]}
19           ]
20        ]
21     ]
22  ]
```

Figure 5: Using Elaboration vs. a Concept Name in the MGDPA study.

The context that contains the frame "data" that does not appear in Figure 5 describes a right to charge the public for disclosing the data; the frame in lines 2-21 is a condition on that data. Assuming the description in lines 6-19 were replaced with a term-of-art (e.g. intellectual property), the unfamiliar reader would be able to understand that some situations permit the government to charge for disclosures without necessarily understanding the term, itself. In the situation where the reader needs to act on or implement this right, extra effort is only then required to reference a definition in another section to understand the term. By replacing explanatory phrases such as the one in Figure 5 with terms-of-art, the regulatory rules become more concise and less prone to variations in the syntax; thus improving readability. In addition, due to the copy-paste operations associated with disjunctions that appear in these explanatory phrases, using a term-or-art decreases the likelihood of demarcation and dissociation errors during formal transcription.

6. Contributions

The frame-based theory yields both theoretical and practical contributions in the form of regulatory constraint patterns and a natural language interface to specify requirements, respectively.

6.1. Regulatory Constraint Patterns

Informally, binary constraints represent meaningful relationships between a *domain*, or the set of things that we wish to restrict, and a *range* that describes the things that restrict the domain. For example, the phrase "the requested access" constrains the set of accesses (the domain) by those that have been *requested* (the range). By breaking similar phrases into frames and slots, we can reason about which phrases comprise a binary constraint and, furthermore, by which values we should define their domain and range.

We present regulatory constraints in two categories as follows: in Table 2, *refinement* constraints elaborate the domain by listing its specializations; and in Table 3, *exceptions* remove elements from consideration in a domain. For each constraint in Tables 2 and 3, the number of constraint frame appearances across the ETOPS (E), HIPAA (H) and MGDPA (M) texts are followed by the constraint phrase. In the Constraint column, the first word describes the domain and the subsequent bracketed phrase is the constraint. The italicized words in the constraint frame slots describe the range or unified slot variable. The frequency with which constraints appear in different regulations is indicative of their generalizability across different regulated domains. For example, many constraints, such as "including" are common across these studies; others, such as "in whole or in part" are relevant only to domain-specific activities, such as access to parts of records, data or information.

Table 2: Refinement Constraints

H	E	M	Constraint Pattern
26	5	0	*rule* {as [applicable {to [*act*]}]}
11	50	4	*rule* {as [follows [*rule*]]}
9	1	6	*rule* {as [necessary {to [*act*]}]}
4	6	2	*rule* {in [a manner {to [*act*]}]}
17	0	1	*action* {in [whole | part]}
111	80	65	*thing* {including [*a thing*]}
115	45	87	*rule* {[the] {following} *elements* [*rule**]]}
31	7	32	*rule* {to [[the] extent {*possible* | of [*an action*]}]}

Table 3: Exception Constraints

H	E	M	Constraint
29	18	21	*rule* {except [as [*acted upon*]]}
11	12	7	*thing* {except [for [*a thing*]}
2	5	3	*thing* {excluding [*a thing*]}
4	1	1	*rule* {in lieu of [*acting*]}
44	39	45	*rule* {unless [*an action*]}
21	56	18	*rule* {without [*action* | *acting*]}

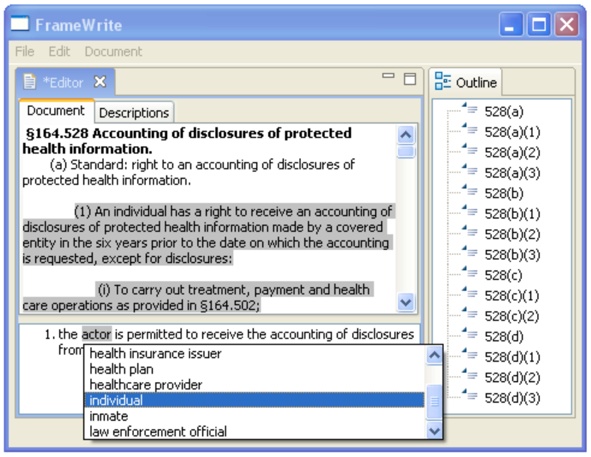
In addition to domains and ranges, each of these constraints has a formal semantics. For example, policy writers provide examples within rules to illustrate the range of interpretations for that rule. The constraint "including" in Table 2 achieves this goal without necessarily excluding interpretations (compared to the constraint "excluding" in Table 3). On the other hand, the constraints "as applicable," "as necessary," "in a manner" and "to the extent" in Table 2 all limit the interpretations of rules using other actions (except for the range "possible" which is always limited by circumstance). In Table 3, the exceptions either define conditions in which rules are not applied or concepts to which other concepts are excluded. In the special constraint "in lieu of," applying the rule removes any requirement to perform the mentioned action. Similarly, the constraint "without" in Table 3 precludes any need to initiate the action or acting. In both cases, these actions may be required by a rule elsewhere in the regulation; presenting a potential conflict.

Full comprehension of regulatory requirements requires engineers to associate constraints with the correct domain and range of interpretation. These patterns highlight a few of the relevant semantics in understanding regulations, however, the act of applying these constraints still carries considerable responsibility. Additional analysis over applications of these constraints can lead to transparent practices that engineers and auditors can use to mutually assure systematic compliance with regulations.

6.2. Natural Language Interface

The frame-based theory has been implemented in a prototype that provides a natural language interface to requirements specification. The Eclipse-based prototype allows users to configure requirements statements using a frame model. If the model has a corresponding denotational semantics, the prototype will generate a first-order predicate logic expression for each statement. The goal is to map these expressions into other formalisms that themselves have established inference and analysis support.

Figure 6 shows a screenshot from the prototype. A user first loads a requirements document formatted in HTML into the upper-left view; the sections in the document appear in the outline along the right side. The user then selects a statement within the document to model in the lower-left view. This lower view allows the user to select frames, starting with frames in the top variable (at the statement-level) and proceeding with subsequent required and optional slots in each selected frame. Each slot has a label which contains an abstract description of the slot. The user navigates slots using the arrow keys: left and right arrows select slots adjacent to the current selection (in printed order), either the next or last slot in the given frame or its context; the up and down arrows select alternate frames for the slot. Holding the ALT key presents optional slots relevant to a selected frame.



§164.528 Accounting of disclosures of protected health information.
(a) Standard: right to an accounting of disclosures of protected health information.

(1) An individual has a right to receive an accounting of disclosures of protected health information made by a covered entity in the six years prior to the date on which the accounting is requested, except for disclosures:

(i) To carry out treatment, payment and health care operations as provided in §164.502;

1. the actor is permitted to receive the accounting of disclosures from

health insurance issuer
health plan
healthcare provider
individual
inmate
law enforcement official

528(a)
528(a)(1)
528(a)(2)
528(a)(3)
528(b)
528(b)(1)
528(b)(2)
528(b)(3)
528(c)
528(c)(1)
528(c)(2)
528(d)
528(d)(1)
528(d)(2)
528(d)(3)

Figure 6: User Selects Frame in the Prototype

The prototype presents some interesting issues that we hope to study in future work, including the time-to-select vs. the time-to-type a requirement and the analyst's natural expectation to select specific phrases in a particular order when composing a requirement. Both of these issues share specific trade-offs. If the frame model is particularly robust, the analyst must navigate larger lists of alternative frames when filling a slot; too many alternatives may be distracting. On the other hand, a robust frame model would be more likely to provide variations in constructing semantically equivalent statements but with different topicalizations; thus accommodating users with different viewpoints on the same requirement.

7. Summary and Future Work

This paper introduces a frame-based theory that can be used to model and analyze domain descriptions and identify constraints on requirements; the theory has been applied to regulatory texts in a cross-domain study. In addition, a Frame-based Requirements Model, developed using this theory, can be given a denotational semantics to align NL requirements with first-order predicate logic. Thus, a unique contribution of this work is that it offers a significant step forward in bridging the gap between natural language requirements and more formal requirements modeling. In contrast to other requirements models, such as goals, our frame-based model provides finer granularity in specifying constraints and greater precision because the model strictly conforms to the text of domain descriptions. The increased granularity can lead to constraint discovery in new domains, as shown in Section 6.1. This greater precision in specification is especially critical in regulatory domains because it is imperative to ensure traceability when requirements models are assigned a formal semantics: if the semantics are misaligned with the intent of the regulation, inferences on those models may lead to non-compliant system behavior. We are currently investigating new ways to generalize frame models using unification and alternative frame-based approaches to specify formal requirements. In tandem, we are evaluating the need for different logics to express formal constraints and requirements. Finally, we are finishing a separate study to compare multiple participants using a single regulatory text.

Appendix A

The frame markup uses a context-free grammar presented below in Bachus-Naur Form with regular expression operators *, +, and ? to denote lists of zero or more, one or more, or zero or one, respectively. The capitalized words and brackets are terminal symbols: in practice, AND is ampersand, OR is vertical bar, COPY is slash, CUT is backslash, PASTE is asterisk, NUMBER is one or more digits 0-9 and TEXT is any printable character except for [,], {, }, &, |.

```
⟨s⟩         := ⟨frame⟩*
⟨frame⟩     := [ ⟨content⟩ ] | { ⟨content⟩ } | TEXT
⟨content⟩   := ⟨op⟩? ⟨frame⟩+ ⟨alt⟩*
⟨alt⟩       := AND ⟨content⟩ | OR ⟨content⟩
⟨op⟩        := ⟨code⟩ NUMBER
⟨code⟩      := COPY | CUT | PASTE
```

Acknowledgements

This work was supported, in part, by NSF ITR grant #0325269 and the IBM CAS PhD fellowship. We thank the members of the NCSU Privacy Place reading group for their comments.

References

[1] J. Allen, *Natural Language Understanding*, Benjamin/Cummings Pub. Co. , New York, NY, 1995.

[2] B.W. Boehm, *Software Engineering Economics*, Prentice Hall, 1981.

[3] T.D. Breaux, A.I. Anton, “Deriving Semantic Models from Privacy Policy Goals,” *IEEE 6th Int’l Work. Policies for Dist. Sys. Nets.*, Stockholm, Sweden, 2005, pp. 67-76.

[4] T.D. Breaux, A.I. Anton, “Analyzing Goal Semantics for Rights, Permissions and Obligations,” *IEEE 13th Int’l Conf. Req’ts Engr.*, Paris, France, 2005, pp. 177-186.

[5] T.D. Breaux, A.I. Anton, “Mining Rule Semantics to Understand Legislative Compliance,” *ACM Work. Privacy in Elec. Soc.*, 2005, pp. 51-54.

[6] T.D. Breaux, M.W. Vail, A.I. Anton, “Towards Compliance: Extracting Rights and Obligations to Align Requirements and Regulations,” *IEEE 14th Int’l Conf. Req’ts Engr.*, 2006, pp. 49-58.

[7] R.L. Cobleigh, G.S. Avrunin, L.A. Clarke, “User Guidance for Creating Precise and Accessible Property Specifications,” *ACM SIGSOFT 14th Int’l Sym. Found. Soft. Engr.*, Portland, OR, Nov. 2006, pp. 208-218.

[8] Ernst and Young, *Global Information Security Survey*, 2006.

[9] “Extended Operations of Multi-Engine Airplanes; Final Rule,” 14 CFR Parts 1, 21, 25, 33, 121, 135. *Federal Register*, vol. 72, no. 9, Jan. 16, 2007, pp. 1807-1887.

[10] C.J. Fillmore, “The Case for Case,” In E. Bach and R. Harms (eds.), *Universals in Linguistic Theory*, Holt, Rhinehart, Winston, NY, 1967, pp. 1-90.

[11] B.A. Garner (ed.), *Black’s Law Dictionary*, 8th ed., Thompson West, St. Paul, MN, 2008.

[12] G. Gazdar, *Generalized Phrase Structure Grammar*, Havard Univ. Press, Cambridge, MA, 1985.

[13] J.W. Hunt, M.L. McIlroy, “An Algorithm for Differential File Comparison,” *Bell Laboratories Computing Science Technical Report #41*, Bell Laboratories, Murray Hill, NJ, Jul. 1976.

[14] M. Jackson, P. Zave, “Domain Descriptions,” *IEEE Symp. Req’ts Engr.*, San Diego, CA, 1993, pp. 56-64.

[15] B. Jacobsen, *Modern Transformational Grammars*, Elsevier Sci. Pub. Co., New York, NY, 1986.

[16] S. Konrad, B.H.C Cheung, “Real-time Specification Patterns,” *IEEE 27th Int’l Conf. Soft. Engr.*, Shanghai, China, May 2005, pp. 372-381.

[17] S. Liaskos, A. Lapouchnian, Y. Yu, E. Yu, J. Mylopoulos, “On Goal Variability Acquisition and Analysis,” *IEEE 14th Int’l Req’ts Engr. Conf.*, Minneapolos, MN, 2006, pp. 76-85.

[18] M. Minksy, “A Framework for Representing Knowledge,” In P. Wilson (ed.) *The Psychology of Computer Vision*, McGraw-Hill, 1975, pp. 211-277.

[19] “Minnesota Government Data Practices Act,” Chapter 13, *Minnesota Statutes*, State of Minnesota, 2006.

[20] A. Ohnishi, C. Potts, “Grounding Scenarios in Frame-based Action Semantics,” *7th Workshop on Req’ts Found. for Soft. Quality*, Interlaken, Switzerland, 2001, pp. 177-182.

[21] D.E. Perry, S.E. Sim, S. Easterbrook, “Case Studies for Software Engineers,” *Int’l Conf. Soft. Engr., Tutorials Notes*, Apr. 2005, pp. 96-159.

[22] H.B. Reubenstein, R.C. Waters, “Requirements Apprentice: Automated Assistance for Requriements Acquisition,” *IEEE Trans. Soft. Engr.*, 17(3), 1991, pp. 226-240.

[23] R.C. Schank, R.P. Abelson, *Scripts, Plans, Goals and Understanding: An Inquiry into Human Knowledge Discovery*. Lawrence Erlbaum Assoc., Hillsdale, NJ, 1977.

[24] R.L. Smith, G.S. Avrunin, L.A. Clarke, L.J. Osterweil, “PROPEL: An Approach Supporting Property Elucidation,” *24th Int’l Conf. Soft. Engr.*, Orlando, FL, May 2002, pp. 11-21.

[25] “Standards for Privacy of Individually Identifiable Health Information.” 45 CFR Part 160, Part 164 Subpart E. *Federal Register*, vol. 68, no. 34, Feb. 20, 2003, pp. 8334 – 8381.

[26] P. Zave, M. Jackson, “Four Dark Corners of Requirements Engineering.” *ACM Trans. on Soft. Enger. Methods.*, 6(1), pp. 1-30, 1997.