Freie Universität Berlin
Department of Mathematics and Computer Science

A Knowledge Representation Framework for Handling Elementary (Patent) Pragmatics

by
Ramakrishna Shashishekar

Submitted in partial fulfillment of the requirements for the degree of doctor rerum naturalium (Dr. rer. nat.)

September 2016

Supervisors

First Supervisor:

Prof. Adrian Paschke
Freie Universitat Berlin, Germany
Institute for Computer Science
Corporate Semantic Web Research Group
&
Director - Domain-Specific Big Data & Analytics Group
Fraunhofer Institute for Open Communication Systems
Berlin, Germany

Additional Supervisors:

Prof. Burkhard Schafer
Director- SCRIPT Center for IT and IP Law
Department of Computational Legal Theory
University of Edinburgh
Edinburgh, Scotland

Prof. Monica Palmirani
Associate professor
Department of Computer Science and Law
University of Bologna
Bologna, Italy

Submission Date September 20, 2016
Disputation Date May 16, 2017

Eidesstattliche Erklarung:

Ich erklare hiermit an Eides Statt, dass ich die vorliegende Dissertation selbstst andig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Diese Arbeit wurde bisher noch nicht in einem fruheren Promotionsverfahren eingereicht. (7 Abs. 4 Promotionsordnung des Fachbereichs Mathematik und Informatik der Freien Universitat Berlin, Stand 2007)

Statement in Lieu of an Oath:

I hereby confirm that I have written this thesis on my own and the contributions by other authors which are used in the thesis or led to the ideas behind the thesis are properly referenced in written form.

I am aware that a dissertation, developed under guidance, is part of the examination and may not be commercially used or transferred to a third party without written permission from my supervisory professor.

Berlin, September 2016

.

[Ramakrishna Shashishekar]
Berlin, Germany

.

Acknowledgement

Foremost, I would like to express my sincere gratitude to my thesis adviser Prof. Adrian Paschke for supervising my research during the past several years. He has not only been a Ph.D supervisor to me, but also a friend, philosopher and a very nice colleague.

I would like to thank both Prof. Burkhard Schafer from University of Edinburgh and Prof. Monica Palmirani from University of Bologna for sharing their experience and unique wisdom with me. Their feedback at different stages of my research has helped me to understand the fine grained intricacies in this domain.

I would like to thank Dr. Naouel Karam for her valuable time spent on our end-less discussions involving both technical and interpersonal aspects.

I would like to thank Prof. Harold Boley from University of New Brunswick, Canada, Dr. Guido Governatori from NICTA- Australia, Prof. Antonino Rotolo from University of Bologna, Italy, Dr.Tara Athan and the think-tanks of legal informatics domain for their precious time spent in providing helpful insights on different aspects of the research problem addressed in this thesis.

I thank the anonymous reviewers for their comments on the early versions of my scientific publications during my work at the Freie Universitat Berlin as research associate and Ph.D student

It has been a great pleasure to work with Marko Harasic, Lukasz Gorski, Moritz von Hoffen, Alexandru Todor, Ralph Schafermeier, Wojciech Lukasiewicz, Alexandra La Fleur, Fiona Dick and many others in the research working group of corporate semantic web at the Freie Universitat Berlin. Specially, I would like to thank Paulina Przyraska for her friendly co-operation during the weekend and off-time research related discussions with Lukasz Gorski.

I would like to dedicate this thesis to my family members, who went to a great extent to, listen, understand and provide the much needed moral support during the entire course of this thesis.

Abstract

Knowledge is a relation between a *'knower* and a *'proposition'*, as defined by Brachman, and its representation depicts the relationship between two domains. As like in any other domain, there exists a gap concerning the understanding of the knowledge from a particular domain between a domain expert and a knowledge engineer who models such domain knowledge - often in a structured, formal language for its use in semi-/automated reasoning. Also, such knowledge representations modeled by the knowledge engineer are not generally automatically reusable outside the specific context for which the knowledge representation was originally developed. Such a problem can be easily seen in the legal domain, wherein, the cost associated with not reducing such gaps is substantially high.

Despite the *'apparent clarity'* of a given legal provision, its application may result in an outcome that does not exactly conform to the semantic level of a statute. In general, laws are designed to be vague. Their vagueness is to accommodate different possible scenarios under which a law can be applied. Pragmatics is an important aspect in the legal domain. It explains the context in which such laws are being applied. Pragmatics within law/legal texts is both boon and bane. The use of pragmatics in applying the laws to different scenarios is good, when viewed from a legal practitioner perspective and the same pragmatics is hard to deal with for a legal knowledge engineer who needs to model it in a precise knowledge representation for semi-/automated legal reasoning systems. The negative aspects of the pragmatics is due to the difficulty involved in separating their concerns.

Adhering to the changes from past few decades within the legal practice, including the vision towards having semi-/automated support to model and reason legal norms using expert systems, for its use in scenarios such as court-fillings or argumentation has increasingly become a subject of interest both in the domain of artificial intelligence and legal community itself.

The core problem in all such automation is handling the vagueness embedded within legal texts/sections and to bridge the gap between legal practitioner and the knowledge modeler. This thesis investigates the research questions of how to represent the separate the concerns thereby indirectly handling such pragmatics.

As a direct contribution the framework presented in this thesis help in minimizing the vagueness, which results in a legal language which is simpler than before. Thereby, providing enough information in an elementary form which can be easier represented using rule representation formats, enabling further automated processing down the line (e.g., automated legal argumentation). The framework, KR4IPLaw-framework, presented in this

thesis is loosely coupled with the OMGs model driven architecture framework. While, the latter focuses on a generalized framework, the framework presented in this thesis is tailored towards the needs and requirements of legal information representation. The framework introduces several new aspects such as decision model layer, a controlled natural language layer, formal representation layers. The thesis contributes in proposing an ad-hoc formal & platform independent representation format, KR4IPLaw-s. As to the logical layer, the framework introduces ad-hoc extension to the existing FODAL logic - an first order logic for deontic and alethic legal rules.

The knowledge based system built based on such proposed framework is validated using a system level evaluation, which not only considers the knowledge in the system, but also the inferencing mechanism, knowledge acquisition mechanism and on how user intuitive it is.

Contents

Chapter 1

Introduction

Contents

1.1 Problem Statement

Automated support to model and reason based on such modeled legal norms using expert systems, for its use scenarios such as court-fillings or argumentation has increasingly become a subject of interest in last few decades. The core problem in all such automation is removing the vagueness embedded with a legal texts/sections and this vagueness is due to the pragmatics involved. Generally, laws are designed to be vague and its vagueness is to accommodate different possible scenarios under which such a law can be applied. Pragmatics is an important aspect in legal domain, it explains the context in which such laws are being applied. Pragmatics within law/legal texts are both boon and bane. This when viewed from the perspective of a legal practitioner, his cognitive reasoning ability allows him to re-apply a law/legal norm to different scenarios (i.e. associating it to different pragmatics) making it the boon. In contrast, the inability of (semi/-)automated decision systems to reason cognitively makes the handling of such pragmatic information a bane. The negative aspects of pragmatics is due to the difficulty involved in separating their concerns. Thus a need for an efficient knowledge representation framework capable for handling legal knowledge (i.e. legal norms and its associated pragmatics) for its use in a decision support system becomes indispensable.

A legal Knowledge Representation (KR) framework also needs to address the non-functional necessities/requirements associated to a legal knowledge, such as:

(a) **Granularity**: The granularity of a representation is a measure of its level of detail. The grain size is the size of the smallest subdivision of reality that is represented in the representation [1]. The smaller the *'grain size'*, the more detail in the KR. For having less uncertainty in the represented knowledge, its is required to have a fine granularity.

(b) **Explicit Vs. Implicit Knowledge**: While the representation of the explicit knowledge is a straight forward requirement, in legal knowledge representation, the implicit knowledge in the form of implicit pragmatics provide aspects which makes a legal norm less uncertain when reasoning.

(c) **Modularity**: Regulations/legal norms are frequently referenced to other sections within a given legal text and even to other pieces of law. Modularization, provides the ability to manage cross-references and maintain traceability from the originating law or regulation, thus forming the basis also for legal information reuse.

(d) **Interoperability**: Interoperability is defined as the ability to exchange information and to mutually use the information which has been exchanged [2]. In law, this refers to as use of the same concepts in different *Acts* with one uniform definition or between Acts of Parliament, secondary regulatory levels and instruments of their implementation.

(e) **Evolvability**: Law evolves over time and so does the pragmatics associated to this law/legal norm (eg. with case-laws). A methodology to accommodate/capture such changes within the considered legal KR format become indispensable.

(f) **Reusability**: Independent of the domain, knowledge reusability is an important KR requirement and is closely tied with the *'Modularity'* requirement [3]. The notion of applying the same law/legal norm under different pragmatics makes the *'Reusability'* aspect an integral part of the core requirements for defining a legal KR format.

1.2 Research Questions

The object of our concern in this thesis is to introduce an efficient disaggregation process for representing legal norms and its associated pragmatics from its natural language non procedural representation format to a formal declarative logical representation format which thereafter can be used within any decision support system for generating legal arguments for an in-court argumentation.

While defining such a process, this thesis addresses a set of research questions closely tied to the general requirements necessary for defining/adapting legal knowledge representation format's.

The research questions addressed in this thesis are as described below:

(i) **What knowledge representation format's/structures are necessary to represent a legal knowledge?** There is no single best knowledge representation language. It is always an instance on the expressiveness versus computational tractability trade-off curve [4]. While there exists few legal KR formats, the question is, the capability of these formats providing support for handling the legal knowledge involving different roles. It leads to a further query as to whether we could expand or adapt or combine them to suit our requirements (i.e. Different representation languages from natural-language format for expressing questions, answers, proofs and explanations to platform-independent serializations in XML and Semantic Web formats to platform-specific executable formats on the logical reasoning layer)?.

(ii) **How are the underlying logics for the proposed legal representation format/'s defined?** Despite the arguments made by many leading philosophers that formal logic rarely controls the outcome of a decision, it is still a critical element when formalizing legal norms from their natural language non procedural form to a formal declarative logical representation format. Modal logics and Description Logics are still a common logical framework used for defining legal norms and legal ontologies respectively. This leads us to a question as to, how do we re-model them to fit to our underlying KR formats defined above.

(iii) **How does the process support life-cycle management of the legal knowledge represented?** Law changes over time and handling these changes is of paramount importance for defining any KR format/'s. The life-cycle management addresses, collaborative knowledge engineering and management (versioning, different roles such as author, maintenance), updates in any National Patent System (NPS) by new decisions which lead to corresponding isomorphic updates in the legal knowledge bases, integration of internal and external (semantic) background knowledge e.g. about skill, elementary pragmatics, usage data (annotations, proofs, etc.).

(iv) **How does the process support the evaluation (verification & validation) of legal knowledge?** As stated before, the process is intended to transform the legal knowledge from its natural language non procedural form to a formal declarative logical representation form. Evaluation of such transformations become indispensable as, for example, it increases the intended users confidence in using such evaluated knowledge for further operations such as deriving inferences for an in-court argumentation.

1.3 Research Methodology

This thesis follows a general design science methodology [5] [6], which offers specific guidelines for building and evaluating the utility of information system research artifacts. Fig 1.1 shows a general science research. A process model is visualized on three layers; the process steps layer, the cognitive process layer and finally the output layer. The *Awareness of problem* and *Suggestion* phase contribute to the *Abduction* part of the

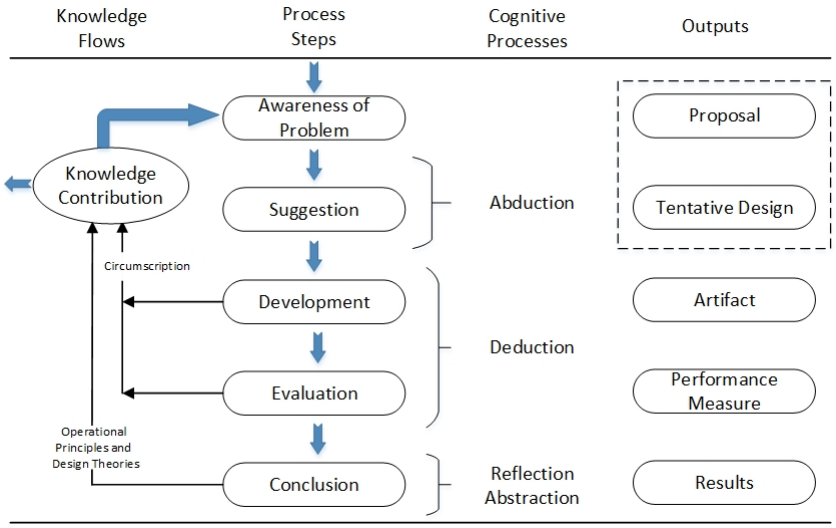


Figure 1.1: The General Methodology of design research with respective cognitive process involved.

cognitive process, wherein *Suggestions* for the problem are drawn from the existing knowledge base of the problems resulting in providing *Proposal* and a *Tentative Design* to the problem under consideration. After proposing a *Tentative Design* in the form of a legal framework, we proceed to the *Development* and *Evaluation* phase of the *Process*, wherein, we introduce the development and evaluation of our semantic-system, KR4IPLaw capable of handling the proposed process. Finally, we provide with a *Conclusion* as a part of our *Reflection & Abstraction* cognitive process to make knowledge contributions of operational principles and possibly design theories.

Development, Evaluation and Suggestion were iteratively performed during the course of this research effort. The basis of the iteration, i.e., the flow from partial completion of the cycle back to Awareness of the Problem, is indicated by the Circumscription arrow.

1.4 Thesis Contributions

The main contribution of this thesis is in the domain of Knowledge Engineering (KE). For the purpose of representing legal norms with its associated pragmatics, this thesis proposes a legal process, KR4IPLaw-p built based on a novel patent information system KR framework, KR4IPLaw-f/w. In particular, the major contributions of the thesis are as follows:

(a) **State of the art**: an in-depth analysis of the feasibility of existing methodologies, methods and tools targeted at the completion of specific reuse activities, concluded by a comprehensive requirements specification.

(b) **Conceptual framework for interpretating pragmatics**: a conceptual framework grounded on the legal theories from Grice and Marmor [7][8] on how one may interpret pragmatics while dealing with the task of legal norm representation.

(c) **KR4IPLaw-framework**: a framework loosely coupled with OMG's Model Driven Architecture[9] for representing knowledge on different layers of granularity. This includes, the decision model layer to represent the procedural aspects of both the substantive law and its related judgments in the form of a generic decision model, a structured legal english layer, wherein SBVR Structured English (SSE) based structured legal english framework is proposed for representing norms from their procedural form (through decision models) to a semi-formal representation, a an ad-hoc formal representation format, KR4IPLaw-s, to existing formats like Legal Rule Markup Language (LegalRuleML) and Rule Markup Language (RuleML) for handling elementary legal (esp patent) norms with elementary pragmatics. The underlying formal semantics for patent law representation language described above are provided using First Order Deontic Alethic Logic (FODAL). On the platform specific representation, a mapping of Prova built-ins to handle modalities were added. The framework is built ground up to support the life-cycle management of the represented legal knowledge.

(d) **Mapping Schemes**: Both contextual and conceptual mapping schema's, as described below to transform the legal knowledge from one layer of KR4IPLaw-f/w to another were proposed.

- Contextual mapping between the frameworks using semiotic/semantic triangle.
- Mapping scheme for transforming knowledge from KR4IPLaw Structured Legal English to KR4IPLaw formal legal rule representation format.
- Mapping scheme for transforming knowledge from KR4IPLaw Structured Legal English to OWL2.
- Mapping scheme for transforming knowledge from KR4IPLaw formal legal rule representation format to platform specific rule representation format, Prova.
- Mapping scheme form parsed legal text to building/populating legal concepts within legal concept recommender system.
- Mapping scheme for informal to formal queries transformation for its use in legal vocabulary evaluation.

(e) **Evaluation Mechanism** Along with the well known information retrieval evaluations approaches, a Competency Question (CQ) based evaluation approach for evaluating legal knowledge transformation and a modality matching approach for legal rule evaluation is proposed.

(f) **Tools**: a context-sensitive, extensible reuse platform architecture and prototypical implementations of the proposed methods.

1.4.1 List of Publications

Parts of the methodology and results described in this thesis have been published and presented at several workshops and conferences. The list of publication are as here under:

(a) S. Ramakrishna, L. Gorski and A. Paschke, "A dialogue between a lawyer and computer scientist. The evaluation of knowledge transformation from legal text to computer-readable format" , Journal of Applied Artificial Intelligence, Volume 31, pp. 216-232, 2016 (accepted) .

(b) S. Ramakrishna, L. Gorski and A. Paschke, "KR4IPLaw Judgment Miner- Case-Law Mining for Legal Norm Annotation" , Artificial Intelligence and the Complexity of Legal Systems (AICOL), JURIX 2015 .

(c) S. Ramakrishna, L. Gorski and A. Paschke, "Legal Vocabulary and its Transformation Evaluation using Competency Questions", International Conference on AI and Law (ICAIL), ACM 2015.

(d) S. Ramakrishna, L. Gorski and A. Paschke, "The Role of Pragmatics in Legal Knowledge Representation", Workshop on Legal Domain And Semantic Web Applications (LegalSW), CoRR- arXiv:1507.02086 [cs.CL] 2015.

(e) S. Ramakrishna and A. Paschke, "A Process for Knowledge Transformation and Knowledge Representation of Patent Law", Rules on the Web. From Theory to Applications (RuleML), LNCS, pp. 311-328, 2014.

(f) S. Ramakrishna and A. Paschke, "Semi-automated Vocabulary Building for Structured Legal English", Rules on the Web. From Theory to Applications (RuleML), LNCS, pp. 201-215, 2014.

(g) S. Ramakrishna and A. Paschke, "Bridging the gap between Legal Practitioners and Knowledge Engineers using Semi-Formal KR", The 8th International Workshop on Value Modeling and Business Ontology (VMBO), 2014.

(h) S. Ramakrishna, "First Approaches on Knowledge Representation of Elementary (Patent) Pragmatics" Rules on the Web. From Theory to Applications (RuleML) 2013 Doctoral Consortium, CEUR-WS vol. Vol-1004, 2013.

(i) S. Ramakrishna, "Cognitive System for Knowledge Representation of Elementary Pragmatics", Rules on the Web. From Theory to Applications (RuleML) 2012 Doctoral Consortium, Seattle, CEUR-WS, vol. 874, pp. 15-24, 2012.

Few sections of this thesis have been presented as use-cases at several tutorials such as:

(a) A. Paschke, N. Karam and S. Ramakrishna, "Examples and Tools - LegalRuleML for Legal Reasoning in Patent Law," LegalRuleML Tutorial, JURIX-2012, Amsterdam, 2012.

(b) A. Paschke and S. Ramakrishna, "Legal RuleML Tutorial Use Case - LegalRuleML for Legal Reasoning in Patent Law," LegalRuleML Tutorial, RuleML-2013, Seattle, 2013.

1.5 Thesis Outline

The thesis is structured as follows:

Chapter 2 provides the necessary foundations to the domain of patent law, semantic web and technologies around it. For a person having pertinent skills in the domain of patent law, National Patent Systems (NPSs) and the Semantic Web, the chapter may seem superflous and may be skipped at times.

Chapter 3 discusses in detail the important aspect of pragmatics in law and its interplay within a legal expert system. The chapter provides a set definitions used through out this thesis.

Chapter 4 starts with the discussion on necessary requirements for a legal knowledge representation and later provides a thorough analysis on the current state of the art in the light of the previously discussed requirements.

Chapter 5 starts with a description of the conceptual (patent information systems KR) framework. It then introduces the process using a generalized process model which later is instantiated to form the KR4IPLaw- Process (KR4IPLaw-p). Through this process, this chapter introduces the different representation formats and its underlying logics used for representing legal norms with its associated pragmatics.

Chapter 6 introduces the implementation of the Knowledge Representation for Intellectual Property Law (KR4IPLaw) as a proof-of-concept,

Chapter 7 evaluates the the domain knowledge representation both from LP and DL perspectives.

Chapter 8 summarizes the achievements and highlights of this thesis. Finally, this chapter outlines directions for future work.

Chapter 2

Foundations

Contents

This chapter provides the necessary foundations to the domain of patent law, semantic web and technologies around it. For a person having pertinent skills in the domain of patent law, National Patent Systems (NPSs) and the Semantic Web, the chapter may seem superflous and may be skipped at times.

2.1 Patent Law

A patent is a set of exclusive rights granted by a sovereign state to an inventor or assignee for a limited period of time in exchange for detailed public disclosure of an invention [10]. The grant and enforcement of patents are governed by NPSs through its national laws, henceforth referred to as "Patent Laws".

Even though, the approach proposed in this thesis can be applied on laws defined by different NPSs, we exemplify our approach on patent laws from United States of America (US) and European Union (EU). We can broadly classify these laws into four groups:

(a) Norms for Formal Rules for Filing.

(b) Norms for Patentability Conditions.

(c) Norms for Post Grant Procedures.

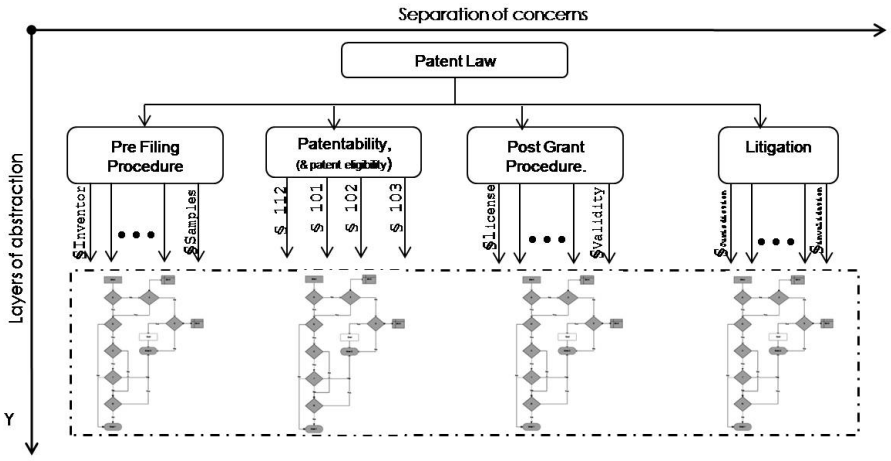


Figure 2.1: A general classification of patent laws in any NPS.

(d) Norms for Litigation.

Figure 2.1 shows the main four broad classification of patent norms in any given NPS. The x-axis focuses on *separation of concerns* and the y-axis shows the *layers of abstraction* defined. It becomes more evident by the end of this thesis as to why separation of concerns and layers of abstraction are needed.

The patent norms as a whole or as individuals classified above, may be referred to as Substantive Patent Law (SPL). Such laws are passed through Acts (e.g. "An Act to promote the progress of useful Arts" passed by US in 1790) by different authorities responsible for patent statues. The responsibility for enforcing the patent norms on received patent application is carried out by respective National/International Patent Offices like United States Patent and Trademark Office (USPTO) in US, European Patent Office (EPO) in EU etc.

SPLs are rich in pragmatics. It is often a complex process for an examiner to apply these SPLs on a patent application to check for its fulfillment of requirements under the governing patent laws. As a solution to this problem, the SPL are converted into Procedural Patent Law (PPL) which are then codified in a manual which acts as guidelines for an examiner helping him in applying respective SPLs on any considered application. Examples for such manuals comprising PPL are Manual for Patent Examination Procedure (MPEP) by USPTO, Manual of Patent Office Practice (MOPOP) by Canadian Patent Office, Manual of Patent Office Practice and Procedure (MPPP) by Indian Patent Office and others

2.2 Semantic Web

Knowledge is a relation between a *knower* and a *proposition*, that is the idea expressed by a simple declarative sentence [11].

e.g. John knows that 'there is a speed limit of 60kmph'.

Representation is a relation between two domains. In the example considered above, a traffic symbol denoting speed limitation is considered as the representation of the knowledge from one domain to another. In the field of Artificial Intelligence (AI), KR focuses on capturing an information which can be further processed with the help of a computer. KR makes the complex and tedious task of knowledge transformation from one domain to another easier.

Semantic Web is an extension of the World Wide Web (WWW) that enables people to share content beyond the boundaries of applications and websites. Berners-Lee, Hendler and Lassila [12] provided the following definition of semantic web:

The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-dened meaning, better enabling computers and people to work in cooperation.

Knowledge representation is a key task enabling technology for the Semantic Web. Different KR techniques used in Semantic web include eXtensible Markup Language (XML), Resource Description Framework (RDF), Ontologies and Web rules. An overview on these techniques are described below:

2.2.1 XML and XMLS

XML itself is not used as a knowledge representation language, but the syntax is used in many KR languages like UML, OWL etc.. They are used to add arbitrary structure to the documents. XML being an application independent language, is often used on the Web to store meta data and XML Schema (XMLS) is a template and a validation document which defines the valid elements and attributes of an XML file. It is also referred to as XML Schema Definition (XSD). The purpose of such schema is to define the legal building blocks of an XML document, of which it defines:

- the elements and attributes that can appear in a XML document.
- datatypes for elements and attributes
- the number of (and order of) child elements

The listing 2.1 shows a simple example of an XML document.

Listing 2.1: Example of an XML document

```
1 <?xml version="1.0"?>
2   <note>
3         <to>Mary</to>
4         <from>John</from>
5         <heading>Reminder</heading>
6         <body>There is a speed limit of 60kmph</body>
7   </note>
```

A well-formed XML document is a document that conforms to the XML syntax rules. In addition to being well-formed, an XML document may be valid. This means that it contains a reference to a Document Type Definition (DTD), and that its elements and attributes are declared in that DTD and follow the grammatical rules for them that the DTD specifies[1].

[1] https://www.w3.org/TR/REC-xml/

2.2.2 RDF and RDFS

RDF is the foundation layer of the Semantic Web, similar to what we have in natural language sentences, the RDF semantics are encoded in sets of triples, where each triple consists of *subject*, a *predicate* or *property* and an *object*. RDF graph is a set of RDF triples and to store the information represented in an RDF graph online, we translate it into RDF/XML format. RDF Schema (RDFS) is a vocabulary language which provides an opportunity to define the terms to be used in a RDF document. It sets the domain and range of properties and relates the RDF classes and properties into taxonomies using RDFS vocabulary. Properties in RDFS are relation between subjects and objects in RDF triples i.e. predicates. Classes are also resources which can be described using properties.

The Listing 2.2 and Fig 2.2 shows a simple RDF graph and its corresponding RDF/XML code.

Listing 2.2: RDF/XML

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <rdf:RDF
3 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
4 xmlns:Law="http://www.xyz.com/patentLaw#">
5 <rdf:Description rdf:about="http://www.xyz.com/doc#
      patentAppl'n">
6 <feature:claim rdf:resource="http://www.xyz.com/claim#
      dependentClaim"/>
7 </rdf:Description>
8 </rdf:RDF>
```

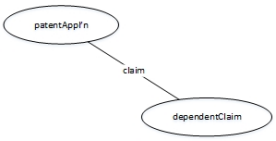


Figure 2.2: RDF Graph.

2.2.3 OWL

An ontology is a specification of a conceptualization. In other words, an ontology represents knowledge as a hierarchy of concepts within a domain, using a shared vocabulary to denote the types, properties and interrelationships of those concepts [13]. The W3C Web Ontology Language (OWL) is a family of KR languages for authoring ontologies, it provides a greater expressivity to represent the web contents than that provided by XML, RDF and RDFS. OWL makes use of notions such as *'classes and subclasses* to classify things in terms of their semantics. Few such notions defined in OWL are as shown below:

- Classes: a class is a collection of objects, all classes are subclasses of `owl:Thing`, the root class.
- Individuals: instances of classes and subclasses.
- Taxonomy: hierarchy of terms is called a taxonomy.
- Properties: a property is a characteristic of a class. It acts as data values, or links to other instances.
    - Object property: relates individuals (instances) of two OWL classes.
    - Datatype property: relates individuals (instances) of OWL classes to literal values.

OWL is further divided into different versions and subversions based on its increasing expressivity factor. Listing 2.3and Fig 2.3 shows a simple OWL ontology as an example.

Listing 2.3: OWL/XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
  <owl:ObjectProperty rdf:about="&PatLaw;performs">
    <rdfs:domain rdf:resource="&PatLaw;examiner"/>
    <rdfs:range rdf:resource="&USPatLaw;
        patent_examination"/>
  </owl:ObjectProperty>
    <owl:Class rdf:about="&PatLaw;examiner">
     <rdfs:subClassOf rdf:resource="&PatLaw;
         patent_office"/>
    </owl:Class>
    <owl:Class rdf:about="&PatLaw;
        patent_examination"/>
    <owl:Class rdf:about="&PatLaw;patent_office"/>
  <owl:NamedIndividual rdf:about="&PatLaw;USPTO">
        <rdf:type rdf:resource="&PatLaw;
            patent_office"/>
  </owl:NamedIndividual>
</rdf:RDF>
```

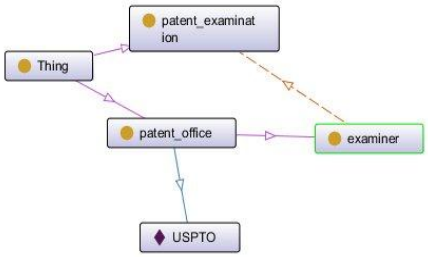


Figure 2.3: OWL ontology Visualization.

2.3 OMG's Model Driven Architecture

Object Management Group (OMG) is a Not-for-profit international standardization body responsible for computer related standards. The OMGs Model Driven Architecture (MDA) [9] provides a basis for representing information on different layers of KR models. The basic concept of MDA is the seperation of the operations of the system from the details of the way that system uses the capabilities of its platform [14]. MDA addresses portability, interoperability and re-usability through architectural separation of concerns. MDA proposes three viewpoints on a system, a computational independent viewpoint, a platform independent and a platform specific viewpoint. These viewpoints correspond to the three models:

(a) Computational Independent Model (CIM) specifies the function (or external behavior) of a system without showing constructional details. I.e. representation of knowledge as structured natural language supporting the domain (legal) experts (e.g. judges, lawyers, patentees, examiners).

(b) Platform Independent Model (PIM) describes the construction of a system on an ontological level, meaning that the construction of the system is specified without implementation details. I.e. representation of the legal information as XML-based legal metadata mark-up and legal XML rule serializations supporting machine processing and interchange (e.g. LegalDocML, LegalRuleML, LKIF).

(c) Platform Specific Model (PSM) A PSM combines the specifications in the PIM with the details that specify how that system uses a particular type of platform.

I.e. representation of formal logical rules in a logic-based rule (+ontology) language supporting automated logical reasoning.

Compared to the traditional approaches on the portability, interoperability and re-usability aspects, the MDA provides:

- Higher level of abstraction than other design processes.
- PSM being a complete description of the application in the form of metadata, it allows design enhancement with technology specific features [15].
- Code generated from the PSM is close to a complete application.

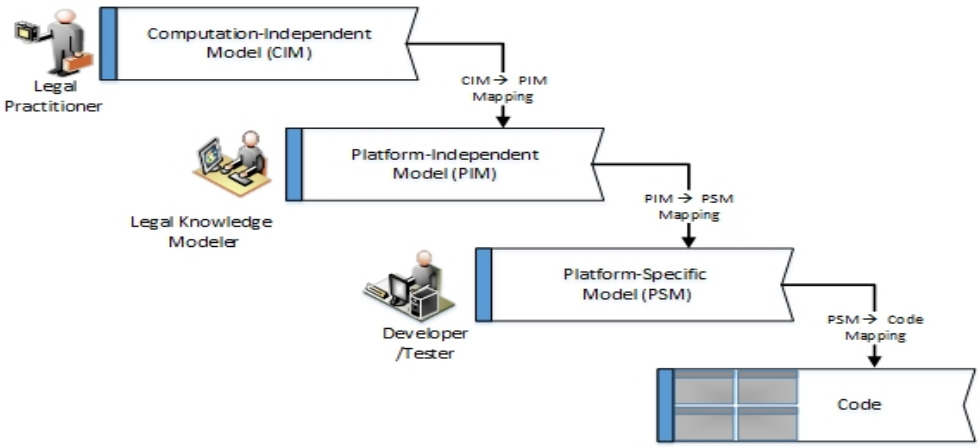


Figure 2.4: Model Driven Architecture.

Fig 2.4 shows the three KR layers of a MDA as seen from a domain expert and Knowledge modelers perspective. In a process of moving between the layers, principal knowledge is added at each step by different professionals. Unlike the traditional approaches, the approach of using formal models allows to define a formal transformation for moving between different models.

The concept of MDA can also be extended to complex systems [2] by allowing multiple horizontal transformation of a model on a layer thereafter, be fed to the model from a different layer (vertical transformations).

2.4 Legal Rules on Web

Web rules are broadly classified into two types; rules that influence the operational/decision processes and constraints that define organizations structure, behavior or information. Rules influencing operational/decision processes of can be further subdivided into two types:

[2] A complex system in this context refers to a system, wherein the transformation from a CIM layer to PSM layer is complex. In other words, models' gaps does not allow to perform a direct transformation

(a) Derivation rules: rules which establishes or derives information that is is used e.g. in a decision process.

(b) Reaction rules: rules that establish when certain activities should take place e.g. Condition-Action rules or Event-Condition-Action rules.

Constraints may be of structural (deontic assignments), state or process/flow constraints. The example below shows an simple rule, both in its natural language format (in *italics*) and prolog style representation format (in `courier`).

An invention is granted a patent, if the invention is novel and nonobvious.
`patent(invention):- novel(invention), nonobvious(invention).`

There exists several formal rule representation languages such as KRIP/L [16], RuleML [17], 2APL [18], NRKL [19], SD-RuleMarkup Language [20], LegalRuleML [21] [22] etc.. Such languages support for rule- modelling, markup, translation, publication and archiving of rules in several formats like RDF, XML, ASCII etc..

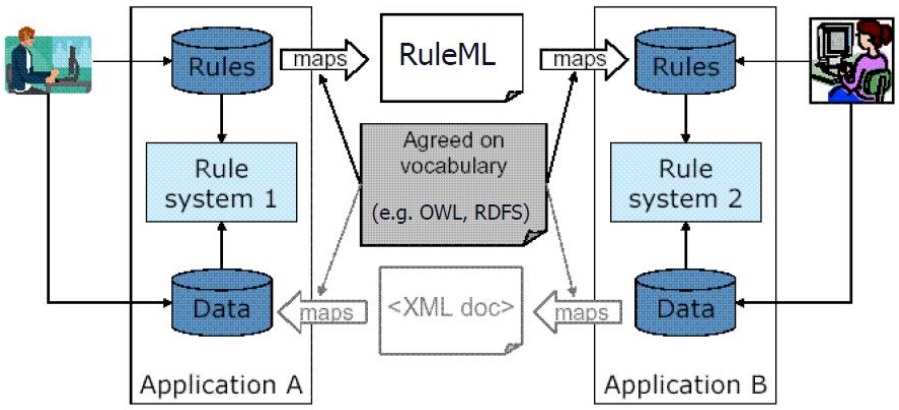


Figure 2.5: Rule Interchange on web

Fig 2.5 shows general rule interchange scenario on the Web. Generally, every application is defined by a set of rules, and two or more applications may share a set of rules. Modularization and interchange capabilities become two important factor that cannot be neglected when building/constructing rules for any said application. From Fig 2.5, it can be seen that two applications 'A' and 'B' have rules built on a agreed vocabulary. Its shown that rules used in 'A' is mapped using RuleML, an rule interchange language to fit to the rules that are needed for the application 'B'. As discussed in earlier subsection, data between different applications can be mapped. We make use of the XML format discussed earlier for the above said purpose.

Later sections of this thesis describes the above rule representation languages and its family of web rule languages in detail.

2.5 Legal Reasoning on Web

The task evident after constructing legal rules is to reason with them against the available facts/data. Any rule can be written (and executed) using imperative or declarative programming approaches. Imperative programming is characterized by a state and commands which modify the state. Declarative programming, on the other hand, describes a proof that some truth holds [23]. The imperative programming can be further divided into procedural programming (e.g., FORTRAN, C) and object oriented programming (e.g., C++, Java) and the declarative programming can be divided into logic programming (e.g., Prolog, Prova) and functional programming (e.g., Haskell, Erlang).

Leith [24] and Subirana et.al [25] provide an overview on legal logic programming and its fundamental errors. This thesis makes use of the declarative programming approaches to construct legal rules and reason them. The main advantage of using a declarative approach is that a declarative program expresses the logic of a computation without explicitly describing its control flow. Making a same legal rule to be used under different contexts as originally intended by the law Maker. Further, AI approaches to legal reasoning have been extensively discussed in [26].

Listing 2.4 depicts an example for constructing rules using Prova (an, declarative programming language).

Consider a company that offers some cloud services and has to comply with the following (legal) rules set by official internet standardization authority:

- If quality of service is high then cost for service can be +5% more.
- If quality of service is low then cost for service has to be -5% less.
- If availability of service is 100% then quality of service is high.
- If availability of service is lower then 99.98% then quality of service is high.

Listing 2.4: Declarative programming example

```
1   Rules:
2      cost ( Service , 0.05) :-  qos( Service , high ).
3      cost ( Service , -0.05) :- qos( Service ,low ).
4      qos( Service , high ):- availability ( Service )=1.
5      qos( Service ,low ):- availability ( Service ) <0.98.
6
7   Queries :
8      cost ( Service ,X)? All discounts for all services
9      cost (s1 ,X)? cost for "s1"
10     cost (s1 ,5\%)? Service "s1" -> cost 5\%?
11     cost ( Service ,5\%)? All services with cost 5\%
12     qos( Service ,Y)? All service levels for all services ?
13     qos(s1 ,Y)? Service level for "s1␣"?
14  ....
```

Here, it says "Give services which have cost associated at 5%", The use of declarative approach provides the several advantages over an imperative approach. Few such advantages of using Declarative Programming Language (DPL) (e.g. Prova) over an Imperative Programming Language (IPL) (e.g. Java) are as described below;

(a) DPL uses fewer coding efforts to implement more queries than an IPL like Java requires for implementing same queries.

(b) DPLs syntax is more closer to natural language and is more understandable to domain (legal) experts than that of an IPL like Java.

(c) An DPL can accommodate any changes to the logic of the program, the rules can be easily modified without extending the inference machine which is difficult when using an IPL, as their logic is deeply buried in the source program [23].

2.6 Summary

The chapter introduced the basic stepping stones/foundations to the domain of law (esp. patent) and the Semantic Web. Firstly, it introduced a general classification structure for any NPSs, later in the domain of semantic web, this thesis introduced, the basic notions XML, RDF, OWL etc.. The chapter also provided a brief overview into the OMGs MDA, which is loosely coupled with the framework introduced in further chapters. Lastly, the chapter provided an birds eye view on the interaction of law and semantic web using (legal) rules and its reasoning on the Web.

Chapter 3

Pragmatics and its role in Legal Norm Representation

Contents

The three major building block of a language are its *syntax, semantics* and *pragmatics*. While, the *syntax* defines the grammar and punctuation of the language, *Semantics* provides the meaning of constructs in a language and *pragmatics* has to do with how well the language connects to the 'real world'.

The role of *syntax, semantics* and *pragmatics* in a language can be easily understood with an example. Consider a "Traffic-Light Situation Awareness" scenario, where the lights displayed are of standard colors (red, yellow, and green) following a universal color code. Wherein the;

- **Syntax:** Bottom light - Green, Middle light - Yellow and Top Light - Red
- **Semantics:** Green light allows traffic to proceed in the direction denoted, Yellow light provides warning that the signal will be changing from green to red and red light prohibits any traffic from proceeding.
- **Pragmatics:** You are obliged to stop at a green light and make way for an approaching emergency service vehicle.

This chapter discusses the notion of pragmatics and how it enriches the law, as the importance of adherence to syntax and semantics in the legal context. It focuses on the statutory law - especially the patent law -, however its premises can apply *mutatis mutandis* to other kinds of legal discourse.

3.1 The Pragmatic Web

Weigand et.al [27] defines The Pragmatic Web [28] as a particular view on the internet as a platform of communication and coordination [29]. Pragmatic Web consists of the tools, practices and theories describing *Why* and *How* people use information. Compared to the *Syntactic Web* (e.g. Webpages linked by HTML references) which is about form or syntactic information resources and the *Semantic Web* (e.g. Ontologies) which is about the semantic resources (meaning) concerning the Syntactic Web, the *Pragmatic Web* is about the interaction which brings about e.g. understanding or commitments [30]. The *Pragmatic Web* enriches the *Semantic Web* by considering the necessary context information and lets this context information evolve.

Norms are represented and reasoned using rule-based systems. Such systems encompass both rule representation languages specifically designed to handle legal rules and rule reasoners. Rules as such provide a mechanism to control and reuse the manifold semantically linked meaning representations published on the semantic web. With the inclusion of the underlying pragmatics (e.g. in the form of micro-ontologies) allows for automated intelligence, which goes beyond simple deductive reasoning. According to Weigand and Paschke [30], an envisioned ubiquitous Pragmatic Web 4.0 - an extension of current Semantic Web 3.0 - is as depicted in Fig 3.1

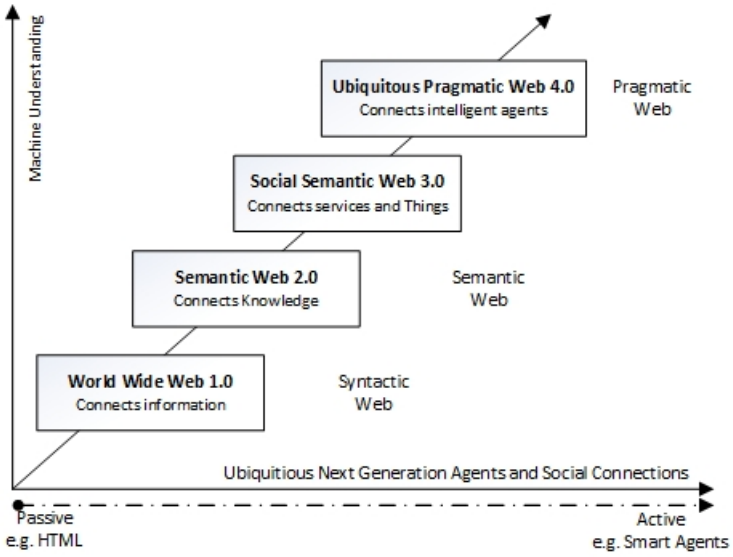


Figure 3.1: Envisioned Pragmatic Web 4.0 (adapted from [30])

3.2 Pragmatics in Law

The distinction of syntax, semantics and pragmatics applies to law as well. Though the linguistic analysis of the legal language has been studied extensively [31][32][33][34], little efforts have been made to combine this with informatics and web technologies. Number of researchers have already suggested that linguistic analysis of the legal language could

be viewed using the conceptual framework offered, *inter alia* by the speech act theory. In this context, Habermas' theory of communicative action is a shining example [35][36].

Induction of speech act theory as an philosophical approach to legal language was proposed by Austin [37] and Searle [38]. They proposed how the utterance of a speaker can be related to the surrounding world.

The basic elements of linguistic conceptual framework of syntax, semantics and pragmatics apply to law as well as any other act of communication [39]. All three facets of the language gain importance, especially in the context of legal argumentation [40].

Despite the apparent clarity of a given legal provision, its application to a particular case may result in an outcome that does not exactly conform to the semantic level of a statute. There may be other conflicting laws in place, and it is a matter of interpretation to choose the prevailing one. Additionally, application of a particular provision may result in an outcome that is absurd or unjust, so the provision has to be interpreted in a way that yields more acceptable results [41]. One should also account for the fact that legislature usually passes a law with a purpose, and that it may be taken into account when interpreting statutes. Finally, also Holmes pointed out the importance of the context: "If Congress has been accustomed to use a certain phrase with a more limited meaning that might be attributed to it by common practice, it would be arbitrary to refuse to consider that fact when we come to interpret the statute" [39]. Similar assumption was put forth by Marmor: "[T]he law typically speaks to a legal community, not to lay people. Legislatures in a modern society mostly address their laws to legal experts, such as judges, lawyers, administrative agencies, etc. There is typically a legal culture that would share a great deal of information and contextual knowledge that is much greater than the relative contextual knowledge of the population at large." [41]

3.2.1 Speech Theory and Patent Law

Speech-act theory is based on conversations to understand the content and the context behind a utterance. Statutes are a way of passing information (e.g. requirements or constraints) between a person/body responsible for creating such statutes and its intended subjects (generally bound by a jurisdiction). This analogy provides us with a conceptual framework for to applying the concepts of speech-act theory to legal linguistics theory dealing with the interpretations of these statutes.

This relation between contextual information and language has already been carefully studied in the field of linguistics, most famously by Grice [8][39][42]. He viewed the conversation as the cooperative endeavor, where not only what has been explicitly said, but also what has been tacitly assumed and is known to both conversation parties matter. Such conversation is guided by the following conversational maxims, which - while obeyed or deliberately flouted - allow to decode what the utterance means on pragmatic level:

- **Maxim of Relevance** - the conversational utterance has to make a contribution to the conversation somehow; i.e. when a conversation party answers that "weather is bad" to a suggestion for going outside, the relevancy maxim says that most probably he meant to decline the proposal and not only comment on a weather.

- **Maxim of Quantity** - the conversational contribution has to be precisely as informative, as required; it should not state more nor less than the speaker intends to; such maxim would be violated in the following sample exchange: A: Can you tell me the time? B: Yes.

- **Maxim of Quality** - don't say something you don't believe to be true; sample exchange that violates this maxim: A: What are major building blocks of a language? B: Only semantics and syntactics.

- **Maxim of Manner** - be brief and orderly; any overly long and complicated utterance violates this maxim.

In the domain of intellectual property laws (including patent laws), as described by Osenga [42], the process of applying the substantive laws on inventions/patent applications can be viewed as a non-verbal conversation between the legislature and patent officer/examiner, and further- between the inventor and the patent office/examiner. Such conversation can be seen in similar lines to utterances as defined by the speech-act theory. According to Marmor, the norms of the statutory interpretation that are adopted by the courts over time can be understood as a sort of Gricean maxims of conversation in legal context. However, it is more problematic to define their scope and exact content [41]. The following examples show, how the pragmatics enriches the semantics of legal provision and how the courts and patent office seek to look for a context that allows to understand the legal provisions regarding the patent law.

Such enrichment can be seen, for example, in the court-mandated doctrine of equivalents (see, for example, *Graver Tank and Manufacturing Co. v. Linde Air Products Co.* [43] or *Winans v. Denmead* [44]). In those cases the courts have found that "In determining whether an accused device or composition infringes a valid patent, resort must be had in the first instance to the words of the claim. If accused matter falls clearly within the claim, infringement is made out, and that is the end of it" [43]. However, taking into account only the literal sense of the patent would be void as it would allow "unscrupulous copyist to make unimportant and insubstantial changes and substitutions in the patent which, though adding nothing, would be enough to take the copied matter outside the claim, and hence outside the reach of law" [43]. Hence, such interpretation would allow to upkeep the patent law's main purpose - protection of inventor and his monopoly over the invention. Doctrine of equivalents is not codified in statutory law, therefore can not be constructed on the basis of the semantics of legal provisions related to patents - it constitutes the pragmatic aspect of patent law.

In its decision in *Ex parte Lundgren* USPTO implicitly recalled the maxim of quantity. In its interpretation of 35 U.S.C. 101 it stated that does not pertain only to "technological arts", as no such reservation was made therein. Obviously, such analysis was also supported by invoking the other aspects of pragmatics, as USPTO inspected a number of prior judgments pertaining to 35 U.S.C. 101 to assure that "technological art test" has not been invoked by the court before.

3.3 Interpretation of Pragmatics

As discussed before, statues can be seen as an non-verbal textual unilateral communication between the legislative body responsible for creating such statutes and its intended subjects (generally bound by a jurisdiction). In the domain of patent laws (Title 35 of the United States Code), the legislative/congress frames the substantive patent laws for its use by the USPTO - inturn by an patent examiner- for examining a patent application for it satisfying the patent norms [1].

Due to the obvious differences between everyday's conversation and the "legislature speech" Grice's maxims do not always apply to legal provisions in their original form. Sinclair [39] extended these maxims to connect with well-established canons of statutory interpretation. We further adapt the maxims to the patent law domain (as shown in Fig 3.2), thereby, providing an qualitative approach for non-ambiguous interpretation of substantive patent norms, which further can be modeled for (semi-/)-automated reasoning.

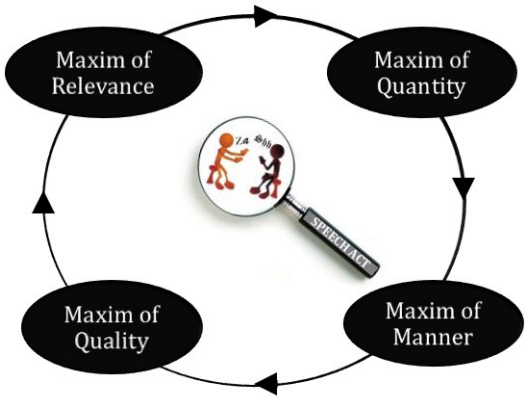


Figure 3.2: Framework on Interpretations

(a) **Maxim of Relevance** - The scope of the statute defines the boundary of each provision and its interpretation; i.e. while US copyright law has provisions that pertain to the works made under the employment contract ("works made for hire"), the patent law does not and copyright law provisions shouldn't be extended to patents.

(b) **Maxim of Quantity** - A statutory provision does not apply to entities or behavior not in its specific domain and does not place controls on entities or behaviors beyond those specified; the *Ex parte Lundgren* mentioned hereinbefore is of an example.

(c) **Maxim of Quality** - Every provision should be interpreted in the light of its objective; the already *Graver Tank and Manufacturing Co. v. Linde Air Products Co.* case can be invoked as an example in this respect.

[1] The other node of the patent examination communication (textual & argumentative) between the applicant and the USPTO, in the form of 'Office Action Response', is out of scope for this paper

(d) **Maxim of Manner** - The statute should be interpreted according to plain, ordinary meaning of its provisions (i.e. in line with the doctrine of plain ordinary meaning), unless explicitly stated otherwise; 35 USC 100 enumerates a number of definitions that alter the plain meaning of a number of terms - i.e. it stats that patentee includes not only the patentee to whom the patent was issued but also the successors in title to the patentee.

Capturing the pragmatics associated to a norm in-line to the maxims is an important step in the direction of modeling these norms. An approach of legal knowledge modularization provides us with the possibility to segregate the three building blocks. A simplest form of such modularization is as shown in the Fig 3.3.

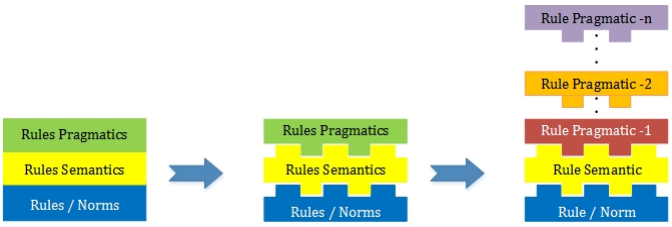


Figure 3.3: Legal Knowledge Modularization

Modularization approach provides us with the capability to apply a norm under different contexts while keeping the syntax (logical part) and its semantics intact.

3.4 Definitions

Different school of thoughts have proposed different definitions to legal terminologies. There exists no common consensus on definitions between school of thoughts. A subset of legal terminologies relevant to this thesis are defined below [2].

3.4.1 Legal Norm

Definition 1. *A legal norm is a rule of conduct of people that may contain a rule on the application of a sanction in the case of its violation [45]. A legal norm α_{legal}, derived out of a legal text/paragraph may be defined as a 'triple' (or 3-tuple) comprising of*

$$\alpha_{legal} \stackrel{def}{=} \langle \mathcal{P}(\alpha), \mathcal{S}(\alpha), \mathcal{L}(\alpha) \rangle$$

Wherein;

- α_{legal} is a considered norm/rule in natural language text.
- $\mathcal{L}(\alpha)$ defines the syntax/logic behind a considered rule.
- $\mathcal{S}(\alpha)$ defines the semantics for α_{legal}.
- $\mathcal{P}(\alpha)$ defines pragmatics applicable for α_{legal}.

[2] Definitions to terminologies pertaining to knowledge representation requirements are covered in 4.1.

3.4.2 Pragmatics

If $\mathcal{D}$ is a domain model of the language under discussion, α a legal text/paragraph from the domain $\mathcal{D}$, a set of questions on α can be written as:

$$q(\alpha)_{n,m} = \{[(p(\alpha)_1, wh_1).....(p(\alpha)_1, wh_m)].....[(p(\alpha)_n, wh_1).....(p(\alpha)_n, wh_m)]\}$$

Wherein, $p(\alpha)_i$, $1 \leqslant i \leqslant n$ are the set of propositions on α and wh_j, $1 \leqslant j \leqslant m$ are the set of *wh*-questions i.e. *'who', 'what' etc....* The validity of a *wh*-question depends on their acceptance by the interlocutors.

An interpretation of a legal text/paragraph α, $\mathcal{I}(\alpha)$ may be defined as a set of $\mathcal{A}(\alpha, q)$, where $\mathcal{A}$ is an answer to a *wh*-question on the legal text/paragraph α.

Definition 2. ***Pragmatics,*** *$\mathcal{P}$ associated with a legal text/norm, $\mathcal{P}(\alpha)$ may be defined as a set of interpretations or utterances (according to speech-act theory) about a norm.*

$$\mathcal{P}(\alpha) = \{\mathcal{I}_1(\alpha), \mathcal{I}_2(\alpha)...., \mathcal{I}_n(\alpha)\} \,|\mathcal{I}_i(\alpha) \mapsto \mathcal{D} \; and \; 1 \leqslant i \leqslant n.$$

Wherein each interpretation, $\mathcal{I}_i(\alpha)$ provides a part of the context information associated to the content i.e. legal norm.

3.4.3 Elementary Pragmatics

Definition 3. ***Elementary Pragmatics,*** *$\mathcal{EP}(\alpha)$ is a part of the context information (either implicit, explicit, intrinsic or extrinsic) associated with a legal norm α. $\mathcal{EP}(\alpha)$ may be defined as below:*

$$\mathcal{EP}(\alpha) = \mathcal{A}((p(\alpha)_i, wh_j))| \; 1 \leqslant i \leqslant n.and \; 1 \leqslant j \leqslant m.$$

3.4.4 Elementary (Legal) Norm

Definition 4. *A legal norm,α_{legal} with a mono-modal formulation (i.e. having one modality) and having $\mathcal{EP}$ (i.e. No-Uncertainty) is referred to as an 'Elementary Legal Norm'.*

3.4.5 Normalized (Legal) Norm

Definition 5. *An 'Elementary Legal Norm' is defined as 'Normalized' if its logical formulation has its modal-operator in front of the formula.*

A non-normalized 'Elementary Legal Norm' can be converted into normalized form using:

(a) 'Barcan Formulae'($\forall x \Box Fx \equiv \Box \forall x Fx$) [46] or/and

(b) Alethic & Deontic Squares

i) $\Box x \rightarrow \boldsymbol{O}x$ i.e. Everything which is Necessary is also Obligatory

ii) $\boldsymbol{O}x \rightarrow \diamond x$ i.e. Everything which is Obligatory is a Possibility

3.4.6 Person Skilled in Art

Person Skilled in Art is a central concept throughout the lifetime of a patent, irrespective of any NPSs. In the domain of KR, this serves as a baseline when considering who should be able to understand the knowledge that is being represented.

Several NPSs have different definitions for the concept *Person Skilled in Art*. In US patent Law, this fictional person is referred as Person Having Ordinary Skill In The Art (PHOSITA) more specifically, *A person of ordinary skill is also a person of ordinary creativity, not an automaton* [47]. According to German patent law, the same fictional person is referred as *Fachmann* - a specialist with average knowledge and talent whom one would ordinarily ask to seek a solution for the (objective) problem the invention deals with. A general definition for the considered fictional person, as applicable to any NPS may be given as

Definition 6. *A 'reasonable man' [48] who has ordinary skill to understand the 'object' of the law and pragmatics referring to the meaning in which the 'object' (patents-norms/ precedents/ guidelines) is understood and applied.*

3.4.7 Substantive Patent Laws

SPL deals with those areas of patent laws which establishes the right and obligation on a patent, on what individual (examiner/inventor) may or may not do. SPLs have the independent power to decide the fate of a patent application.

Definition 7. *The part of the patent law that creates, defines, and regulates rights. For example, the rules for what is patentable.*

3.4.8 Procedural Patent laws

Contrary to SPL, PPL have no independent powers. It simply deals with and lays down the ways and means by which substantive laws can be enforced [49]. PPLs are described by NPSs through their patent examination manuals.

Definition 8. *Procedural law is the body of legal rules that govern the process for determining the rights of parties.*

3.4.9 Prior Art

Prior-art or state-of-the-art are the known information about a topic from all sources. In the domain of patents, prior art does not need to exist physically or be commercially available. It is enough that someone, somewhere, sometime previously has described or shown or made something that contains a use of technology that is very similar to your invention [50].

Definition 9. *Everything which has been made available to the public before the relevant date anywhere in the world by means of written disclosure and which can be of assistance in determining whether the claimed invention is new and involves an inventive step (i.e. is not obvious) for the purposes of international search and international preliminary examination* [3].

Definitions to terminologies pertaining to knowledge representation requirements, logical formats etc.. are covered in the next few chapters.

3.5 Summary

The chapter dealt with the notion of pragmatics and its role in legal norm representation, thereby grounding all the definitions used further in this thesis. The thesis initially introduced the new direction of Web, the pragmatic Web or simply known as *'active'* web. It also provided an overview on the notion pragmatics, as seen through legal domain perspective.

The chapter further provided a detailed discussion on the basic concepts of speech theory act and showed its interconnections to the domain of law (esp. patent law). Furthermore, the thesis proposed a set of theoretical guidelines grounded based on the legal theories from Grice and Marmor on how one may interpret pragmatics when dealing with the task of legal norm representation.

To the end, this chapter introduced a set of definition, which in the upcoming chapters, are further used to define the knowledge representation framework.

[3] http://www.wipo.int/pct/en/texts/glossary.html

Chapter 4

Related Work

Contents

Designing a new knowledge representation format or adopting an existing one, for its application in the domain of law, requires the fulfillment of certain requirements. Such requirements also encompasses general knowledge representation requirements, which are independent to any domain. This chapter initially introduces to such requirements and later focuses on existing related technologies pertaining to legal knowledge representation. We further discuss their advantages, disadvantages and their applicability in the domain of patent law.

4.1 Requirements

This section initially introduces a set of requirements, independent of its application domain, for designing a knowledge representation language. Later, the additional requirements required for a legal (rule) knowledge representation are presented.

4.1.1 Representational Adequacy

Representational Adequacy refers to the ability to represent all the different kinds of knowledge that might be needed in that domain. This branches out into sub-requirements such as Expressiveness, Naturalness, Simplicity, Precision or Semantic Clarity.

4.1.1.1 Expressiveness

The language should be expressive enough for the knowledge engineer to say most of what he/she wants to say. *'Expressive Power'*, is a metric used to measure the expressivity of a language. Levesque and Brachman defined it as "The expressive power ... determines not so much what can be said, but what can be left unsaid." [51]. A more formal definition of the metric was given by Baader in [52]. Another important yardstick to measure the expressivity, describing the relative expressive power of formalisms, of a formal language was provided by Chomsky [53], also referred to as Chomsky hierarchy [1]. Table 4.1, depicts the Chomsky hierarchy against different types of formal languages encapsulating the grammar within.

Table 4.1: Chomsky hierarchy

Chomsky Hierarchy	Language Type	Grammar
Type-0	Recursively enumerable	Unrestricted
Type-1	Context-sensitive	Context-sensitive
Type-2	Context-free	Context-free
Type-3	Regular/Non-recursive	Regular/Finite

From Table 4.1, we can infer that, the higher we climb the Chomsky-Type ladder, the more restrictive the language is. This provides us with an understanding on the trade-off that exist between the expressiveness and tractability.

4.1.1.2 Naturalness

The language should be as close as possible to its base language of considered problem domain. In field of linguistics Dessler [54], proposed a metric, *'Naturalness Scale'* to evaluate the Naturalness of a language. The end points of this scale being *more natural* and *less natural*. These values are assigned based on:

(a) relations between content (meaning) entities,

(b) relations between expression (form) entities and

(c) mappings between content entities and their respective expressions

Oresnik [55], provides additional criteria for identifying the *'Naturalness Value'* for a given language. Such criteria include *Processing ease, Cognitive simplicity, Relative frequency etc....* As to a conclusion of this requirement, the more natural the representation language is, the better understanding of the existing knowledge and communication with the knowledge engineer.

[1] A deeper discussion on 'Formal Grammar' or on 'Operations on Languages' are out of the scope of this thesis and hence not considered here

4.1.1.3 Simplicity

Simplicity is a key factor for achieving efficiency, usability and maintainability. There exist a trade-off between simplicity and expressivity. A language should be as simple as possible in order to express the concepts of interest.

4.1.1.4 Semantic Clarity

The language constructs should have a clear and well-defined semantics. The semantic should allow for correctly computing the legal effects that follows from a set of legal rules [56]

4.1.2 Inferential Adequacy

The process of deriving new facts from given facts is called *inference*. Inferential Adequacy refers to the ability to manipulate the representational structures to derive new structures (corresponding to new knowledge) from existing structures. In other words, the representational format should be able to allow the system/engine reasoning on top of it to infer answers to a broad range of questions, minimizing the degree of incompleteness in its inference process.

4.1.3 Representational Efficiency

Representational Efficiency refers to ability to manage the represented information. The requirements associated with the representation efficiency are as described below.

4.1.3.1 Knowledge Reuse

In Markus theory of knowledge reuse [57], a knowledge reuse is the process that involves sharing best practices or helping others to solve common technical problems. In order to reuse the represented knowledge, the representation format should support the functionalities such as Repackaging, Decontextualizing, Recontextualization, Support for indexing and searching etc.. Knowledge reuse is tightly linked to Integrability.

4.1.3.2 Integrability

The ability of the representation structure to allow its use in concert with other structures and tools with minimal effort.

4.1.3.3 Knowledge Encapsulation

A structure should support the encapsulation of coherent collections of knowledge together into a single conceptual unit for its use in secnarios such as Multiple Modeling - allow alternative, possibly conflicting representations to co-exist- or for compositional modeling.

4.1.3.4 Extensibility

The ability of the structure to support additional constructs and concepts i.e. to provides constructs to help manage large-scale descriptions.

4.1.3.5 Usability

It should provide an intuitive method for understanding and using the knowledge representation format. Generally, usability demands extensibility with a trade-off with expressivity. The more expressive a knowledge representation structure is, the more difficult it is to understand the knowledge intuitively.

In addition to the requirements discussed above, for a knowledge representation format to be used in the domain of law, it has to fulfill certain additional requirements. A list of such additional requirements has been put together by Gordon et.al in [56]. This thesis, while re-iterating these requirements, extends it from a patent norm representation perspective.

4.1.4 Isomorphism

Bench-Capon and Conen [58] defined Isomorphism as "the term is intended to capture the notion of creating a well defined correspondence between source documents and the representation of the information they contain used in the system". In short, it refers to the well defined correspondence of the (legal) knowledge base to their (legal) source texts. Such a requirement helps in verification and validation of knowledge bases and its maintainence. In addition, it will also be able to record the provenance of all items of knowledge in the intermediate representation.

In order for a representation format to fulfill the isomorphic requirement, Karpf proposed a set of criteria's/sub-requirements that needs to be fulfilled [59] [2].

(a) Each legal source must be represented separately.

(b) The ability of the representation format to preserve the structure of each legal source.

(c) The ability to preserve the traditional mutual relations, references and connections between legal sources.

(d) The ability to separate the legal sources and their relations from queries and facts.

4.1.5 Reification

The rule representation format should be able to manage rule reification in order to have an effective reasoning. Reification in knowledge representation involves the representation of factual assertions, that are referred to by other assertions. Such assertions helps in

[2] There exists no online version of the paper. The Karpf's criteria proposed here have been collected from a series of publications [58][60][61], which in-turn cite Karpf publication [59]

provenance. Provenance is information about entities, activities, and people involved in producing a piece of data or thing, which can be used to form assessments about its quality, reliability or trustworthiness [62]. If rules are viewed as objects then its properties are as shown below:

4.1.5.1 Jurisdiction

Jurisdiction refers to the limits under which an entity can exhibit legal rights. Its one of the important aspect to be considered when reasoning/applying a legal norm. It provides an geographic area or subject-matter over which an Authority applies its legal power. The possibility to associate jurisdiction/'s to one or set of rules is an important aspect.

4.1.5.2 Author & Authority

Similar to Jurisdiction, associating the authority to a rule plays an important aspect when reasoning on top of a the legal knowledge base. It indicates a ranking status of by the role played in framing the rule and indirectly, the hierarchy of the rule (e.g. Statutes vs MEMO). It becomes more evident, when handling case-laws and procedural laws.

4.1.5.3 Temporal Properties

Provides the time of application of rules. The first approaches towards modeling legal rules with temporal aspects was discussed by Governatori et.al [63] and Palmirani et.al [64]. Temporal properties that play an important role towards legal rule modeling are:

(a) Enactment: The time when a norm was passed.

(b) Efficacy: The time when a norm starts to produce a desired or intended effect.

(c) Transition: Defines the transition of a old rule to a new rule.

4.1.6 Rule Validity

Rules can be valid or invalid. Marmor [65] interprets Kelsen's definition on validity [66] as: a norm can only be legally valid if it belongs to a system, a legal order, that is by and large actually practiced by a certain population. This is closely related to its temporal aspects.

4.1.7 Legal Procedures

The representation format should be able to handle procedures not just to define a relation between rules, especially when handling procedural laws, but also to handle legal conflicts and normative effects triggered by a norm violation.

4.1.8 Life Cycle Management

Legal norms continue to evolve after they emerge, while this requirement encapsulates other requirements described above, an ability to track a norm from its creation to its annulment. This also addresses issues pertaining to collaborative legal knowledge engineering and maintenance via versioning.

Further sections of this chapter provide a comprehensive discussion on existing works pertaining to legal knowledge representation, its advantages, its drawbacks based on the requirements discussed in previous Section 4.1.

4.2 Legal Section/Document Markup

Several method have been proposed for annotating a law/legal document. Few amongst them are EnAct [67], EUR-LEX [68], MetaLex [69], Akoma Ntoso [70][71] etc..

4.2.1 EnAct

EnAct is a legislation drafting, management and delivery system. Internally, EnAct made use of a Standard Generalized Markup Language (SGML) format to store the legal information in a structured manner. To accommodate the changes in law over time, The acts were stored in Structured Information Manager (SIM) as fragments with a timestamp marking the time interval over which the fragment or table of contents is valid. The EnAct system also incorporated an SGML parser which allowed indexing on the logical structure of the Acts represented as SGML fragment, which enabled time point searching. To accommodate cross-referencing between legal documents e.g. between acts and by-laws, all cross-references were activated as hypertext links, which were activated using queries to the database. As an example, whenever history notes are displayed, those that refer to amending Acts on the system are also displayed as hyper text links on those Acts. Periodical changes to the law need to be stored to generate amendment wordings, which are appended to a stub-or substantive bill. EnAct uses a Change Description Document (CDD), to capture all the changes done on a Act represented using a SGML format. Fig 4.1, shows a general architecture of EnAct including its CDD feature. While EnAct provides the very basic support to markup the legal documents, its fails to provide the necessary modularization mechanism, interchange mechanism, multilingualism features necessary for the legal document management in a global perspective as seen in patent laws.

4.2.2 EUR-Lex FORMEX

Formalized Exchange of Electronic Publications (FORMEX), currently is an XML standard used for management of legal information under EUR-lex repository. It provides an huge collection of European legal documents considered to be public. The use of FORMEX supports jurisdiction specific language representation, as a result EUR-lex currently provides EU-law, preparatory acts, EU case-law and international agreements in

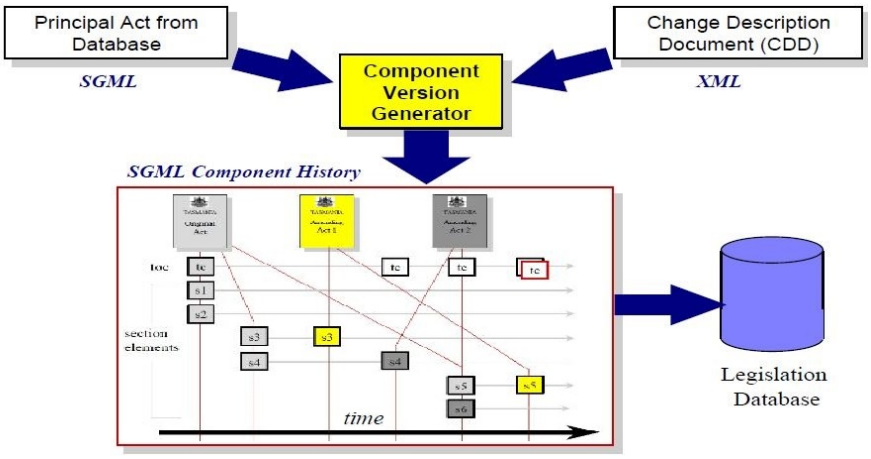


Figure 4.1: EnAct Architecture (adapted from [67])

24 official EU languages. FORMEX supports analytical information such as publication, reference, dates, keywords in the form of meta data. It also supports the lifecycle management of the legal document. Eventhough, FORMEX has been been widely accepted under EURlex community with more than 249 million pages consulted, it currently aim at providing only a legal document management option and not a legal rule representation option as required by our process in the next steps.

4.2.3 MetaLex

Metalex is an XML standard for the markup of legal sources. Metalex enables to represent the sources of law and references to sources of law in an XML format. Main idea of Metalex is to provide an interchange format providing a standardization view on legal documents for the purpose of information exchange and interoperability. It defines, a mechanism for schema extension, adding metadata, cross referencing, constructing compound documents and a basic naming convention. It opens up an possibility to include not only legislation and case laws, but also written public decisions, internal and external business regulations, and contracts. The main features supported by Metalex are:

- Version Management and Maintenance: It proposes to use four attributes, *date-enacted, date-repeated, date-publication and date-effective* which support automatic generation of current and past versions of a legal text.
- Presentation: Supports different representation formats, such as XHTML, PDF, RTF etc.
- Extensibility: It is designed to be embedded in technologies for legal knowledge representation, code generation, rule generation and verification of legally relevant "contents".
- Filtering: Supports searching and filtering on legal documents

- Multiple Language: Supports the same content written down in one Metalex document using multiple languages for its use in jurisdiction-specific language scenario.

Metalex incorporates most of the missing features from *EnAct* necessary for the legal document markup, it lacks an hierarchical structuring, authority definition, modularization features.

4.2.4 Akoma Ntoso

Akoma Ntoso[3] is an standardization effort under OASIS, It defines a common format for documents of parliamentary activities such as primary legislation, parliamentary debates, amendments, briefs, journals. It also support documentation of judgments and opinions. Fig 4.2 shows the possible structure of attributes associated with markup of any legislative Act using legalDocML. The metadata associated with it supports the long-term preser-

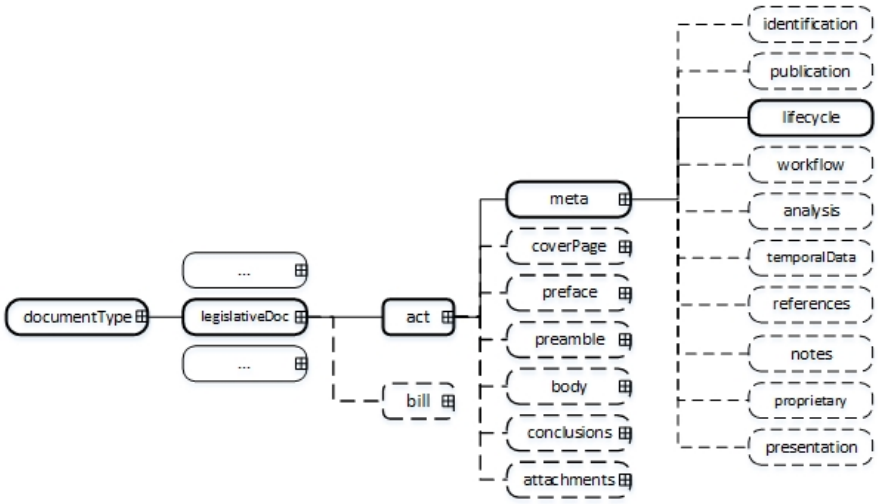


Figure 4.2: Akoma Ntoso attribute structure for legislative Act markup (adapted from [70])

vation (i.e. evolution of laws over time) of legal documents with their intended meanings, by creating a common data and metadata models. The models use a common resource naming and linking mechanism to be easy enough to be cited and cross-referenced when dealing different section of law. LegalDocML addresses all the problems discussed under different legal document markup schemes, its sharing of the metamodel with legal rule representation language, LegalRuleML makes it as the best option for legal section/document markup step. A detailed discussion on each attributes of LegalDocML, its use and adaption to suit patent section mark up is explained in detail under chapter 5.

[3]LegalDocML is an affiliated LegalXML Member Section Technical Committee under OASIS for Akoma Ntoso.

4.3 Semi-Formal Legal Knowledge Representation

Controlled Natural Language is a subset of natural language that can be accurately and effectively processed by a computer but is expressive enough to allow natural usage by a non-specialist. There exists a wide variety of CNL, amongst them we consider a subset of CNLs to study its applicability to our problem domain:

4.3.1 Attempto Controlled English

Attempto Controlled English (ACE) is a CNL defined precisely by a subset of full English that can automatically and unambigiously be translated into Discourse Representation Structures (DRSs), a variant of First Order Logic (FOL). It includes restricted syntax and a restricted semantics (of base language) described by a small set of construction and interpretation rules [72]. The vocabulary of ACE comprises of predefined words (e.g. *determiners, conjunctions, prepositions*) or user-defined, domain specific content words (e.g. *nouns, verbs, adverbs*). ACEs grammar is expressed as a small set of construction rules and an ACE sentence constructed using the built vocabulary may be viewed as a sequence of anaphorically interrelated sentences.

ACE texts may be broadly divided into three types

4.3.1.1 Simple Sentences

Simple sentences follows general structure as shown below;

$$\text{subject} + \text{verb} + \text{complements} + \text{adjuncts}.$$

Wherein, *complements* being the direct or indirect objects which are necessary for defining transitive/ditransitive verbs and *adjuncts* are optional parameter which is used when defining prepositional phrases. Below shown are some examples of simple sentences

(a) An invention must be novel.

(b) An invention should be non-obvious.

4.3.1.2 Composite Sentences

Composite sentences are recursively built from simpler sentences through *coordination, subordination, quantification* and *negation*. Below shown are examples of such constructed composite sentences.

(a) An invention must be novel and non-obvious.

(b) Every Invention must be novel and non-obvious.

4.3.1.3 Query Sentences

Query sentences are built to query the contents of an ACE text. A query sentence may be of two type; *yes/no* query, for establishing the existence or non-existence of a specified situation and a *wh*-query, for querying an ACE texts for details of the specified situation. Examples for both query types are as shown below.

(a) Yes/no-query: Is the invention novel?

(b) Wh-query: How should the invention be?

ACE handles modality through fixed modal phrases and sentence reification. The introduction of fixed modal phrases only modifies respective embedded sentences, thus limiting ACE to FOL even with the addition of fixed modal phrases.

4.3.2 Drafters Language

Drafters Language is a CNL originally developed for DRAFTER system. It works on a conceptual authoring approach which provided a relatively simple pseudo-text to specify a complex configuration of action and object entities and the relations between them [73][74].

Underlying processing component of this system is a *domain Knowledge base*, which acts as a central repository for all the information that might potentially be expressed in the texts regardless of the language or style used. The knowledge is derived from a multilingual corpus (useful when dealing with multilingual translations) and stored as different entities. These entities include *actions, states, objects* and *set of relations between them*. It also allowed authors to include information other than those found in the knowledge base. The system provided an Controlled Natural Language (CNL) interface allowing the authors/users to work in terms of pseudo-texts through *define_action* and *select_a_pattern* functionality. The CNL grammar is maintained as a part of the domain knowledge base, and reflects the concepts and relations in the domain hierarchy. A CNL construction of a legal text is as shown below,

(a) **[** person **]** examines **[** application **]**

Wherein, the words in the brackets could be further specified by the list of available patterns (using the *select_a_pattern* functionality) leading to the following

(a) **[** examiner**]** examines **[** patent application **]**

While the structural ambiguity is resolved with the use of *modification* actions, the users have no capability to edit the CNL texts directly.

4.3.3 Massachusetts Legislative Drafting Language

Is a CNL developed for describing legal texts (originally for Massachusetts Senate). The core idea behind such a approach was to provide a uniformity in drafting style by specifying a restricted syntax, restricted semantics and restricted document structure [75]. It provided pointer for drafting (in turn representing) a legal text. Few important classes of it are as descibed below.

4.3.3.1 Simplicity

(a) Selecting short familiar words and phrases and to avoid *Legalese*.

(b) Do not use both a word and its synonym. Etc.

4.3.3.2 Conciseness

(a) Omit needless language.

(b) If a word has the same meaning as a phrase, use the word. Etc.

4.3.3.3 Consistency

(a) keep the semantics of each word intact

(b) Being consistent in the arrangement of comparable paragraphs. Etc.

4.3.3.4 Sentence structure

(a) Defines the use of *Subject, Verbs, Finite Verbs* and *Modifiers* within a legal text.

(b) Defines the use of *Numbers, Organizations etc..*within a legal text.

Along with the forementioned pointers, it also provided a small set of around 100 keywords, to be used in the process of drafting a legal text. Few such keywords are *void, beacuse, compute, start, begin, give, 'before this takes effect'* etc..

Eventhough, this CNL effort provided an appoach to keep the legal language simple and easy to understand but it lacked the needed expressiveness to represent a range of legal norm with its restricted syntax and semantics.

4.3.4 SBVR Structured English

SSE is a CNL originally developed for representing business rules. It is more reliable for automatic interpretation due to its high syntax restrictions. It ignores the grammatical structure followed by its peer base language when representing the same rule/statement [76].

The OMGs Model Driven Architecture, MDA [9] provides a basis for representing information on different layers of knowledge representation models (CIM, PIM and PSM). Semantic Business Vocabulary and Business Rules, SBVR [76], is an ISO terminological dictionary (vocabulary) for defining business concepts and rules. SBVR works on the Computational Independent Model (CIM) layer of the OMGs MDA. It supports the use of Structured English (SE), a computational-independent English (natural) with a structured syntax for representing business vocabularies and business rules. SBVR captures the structural and behavioral aspects of business processes, as well as the policies that

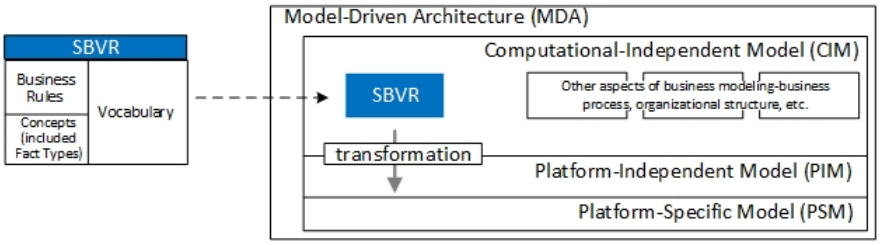


Figure 4.3: SBVR position in MDA (adapted from [76])

should guide the business behavior in certain situations. Fig 4.3 depicts the relation of SBVR and OMGs MDA. A core idea of business rules formally supported by SBVR is the following: Rules build on facts, and facts build on concepts as expressed by terms. Terms express business concepts; facts make assertions about these concepts; rules constrain and support these facts [76]. SBVR is built based on three important elements:

4.3.4.1 Noun concept

Noun concept is a concept that is the meaning of noun or noun phrase. An unitary noun concept whose extensions are necessarily invariant across all possible worlds are referred to as individual noun concept. Noun concepts defined in green and individual noun concepts are defined in dark-green color starting with capital letters [4].

E.g.

(a) examiner, claim

(b) **USA**.

4.3.4.2 Fact type

They denotes some type of relationship between two or more noun concept or a characteristics of a noun concept using a *verb concept*. Verb concepts are defined in blue color with italics font style. E.g.

(a) claims *are_mandatory*.

(b) examiner *rejects* the claim.

4.3.4.3 Rules

Rules defined under business jurisdiction, they may be formulated as *necessities, possibilities, permissions, prohibitions* or *obligations* related to facts types. E.g.

(a) It is obligatory that examiner *rejects* the claim, if claim *is_about* living_organisms.

[4] [76] differentiates the noun and individual noun concepts using single and double underline in addition to green and dark green colors respectively

(b) It is necessary that a patent_application *includes* atleast one claim.

Logical rule formulations in SBVR are equivalent to formulae in 2-valued, first-order predicate calculus with identity [76]. In addition to standard universal ($\forall$) and existential ($\exists$) quantifiers, for the sake of convenience, SBVR standard allows logical formulation to use some pre-defined [77] numeric quantifiers, such as at-most-one ($\exists^{0..1}$), exactly-n ($\exists^{n}, n \geqslant 1$) and others.

In order to express the structural or operational nature of a business rule, the corresponding rule formulation use any of the basic alethic or deontic modalities. Structural rule formulations use alethic operators: $\Box$ = it is necessary that and $\Diamond$ = it is possible that; while operative rule formulations use deontic modal operators $\boldsymbol{O}$ = it is obligatory that, $\boldsymbol{P}$ = it is permitted that, as well as $\boldsymbol{F}$ = it is forbidden that. A detailed description of the use of SBVR in legal domain is described in chapter 5.

4.3.5 Comparision of CNL's

To compare the efficiency of different CNL's we use the evaluation methodology as proposed by Kuhn [78] [79]. The evaluation is done based on the four parameters, also combinedly referred as PENS classification [79]. The parameters themselves are based on works from Mitamura et al. [80], Boyd et al. [81], Pool [82] and Wyner et al. [83].

- Precision: Shows the degree to which the meaning of text can be directly retrieved from its textual form.
- Expressivity: Describes the range of propositions that a certain language is able to express.
- Naturalness: Describes how close the language is to its base English (base language of considered problem domain) language.
- Simplicity: Describes simplicity/complexity of exact and comprehensive language description.

The grading of CNL's, based on these parameters is done on a scale of 1-5. As an example to the proposed metric system, when dealing with parameters such as *precision* and *simplicity*, plain English language may be considered as 1 i.e. being more precise and simple against a pure propositional logic being less precise and complicated i.e. 5 on the metric system.

Fig 4.4, compares the four CNL's based on the four parameters discussed above. From the Figure we see that two out of four CNL's, i.e. SBVR-SE and ACE seem to fulfill the requirements required to represent our problem domain. Legal practitioners being both the authors and end-users of CNL based systems, we need to add another evaluation parameter 'learning curve'. From a legal practitioners point of view, the learning curve involved in ACE seemed to be higher than that involved in SBVR-SE.

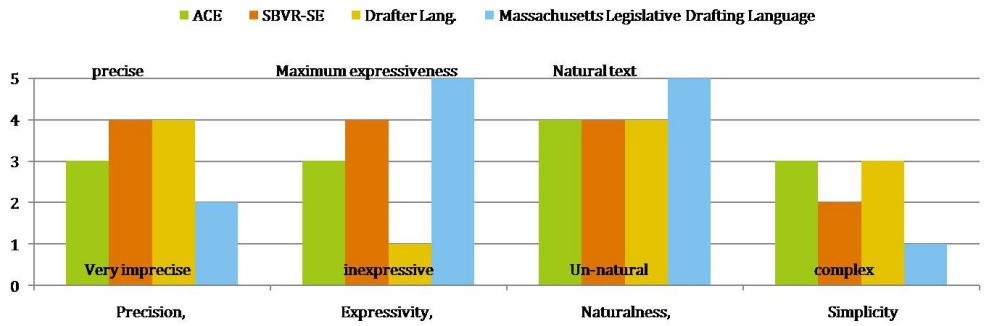


Figure 4.4: Comparision of CNL's

4.4 Legal Rule Logical Formalisms

Despite the arguments made by many leading legal philosophers that formal logic rarely controls the outcome of a decision, it is still the critical element in making a legal decision. Law and logic are intertwined. This section provides a detailed discussion on the existing logic used for legal reasoning.

4.4.1 Deductive Logic

A deductive logic is one wherein, a conclusion is proved by means of two other propositions. Instead of a result being compelled by two facts its is instead inferred from two premises. It provides a grounded foundations for conclusions. A generalized syntax for decudtive logic may be *"if A and B are true so must C"*. wherein, A is refereed as major premise, B as minor premise and C is the conclusion. Different form of decudtive reasoning is as shown in Listing 4.1.

Listing 4.1: Deductive logic

```
(a)   Modus Ponens
      If p then q.
      p.
      Therefore q.

(b) Negative form: Modus Tollens
      If p then q.
      not q.
      Therefore not p.

(c) Disjunctive form: Disjunctive Syllogism
      p or q.
      Not p.
      Therefore, q.
```

Eventhough, the use of deductive logic has some advantages in legal reasoning due to its simplicity, it also has some downsides. As for any legal reasoning, you need an excellent volume of knowledge, which is otherwise not the case of deductive reasoning which works on abstraction. The task of constructing both the major and minor premise accurately becomes a challenging task.

4.4.2 Inductive Logic

Inductive generalization works opposite to deductive reasoning. The idea is by arriving at generalizations based upon a small and more discrete event. Inductive reasoning is used when there are less or insufficient precedents or basically no major premise. Inductive reasoning may also be based on analogy, wherein, a comparison of two things is made in order to draw a conclusion about both. A generalized (Enumerative) induction logic is of the form as shown in Listing 4.2.

Listing 4.2: Inductive logic

```
1   Some As are B
2   Therefore, All As are B
```

A conclusion obtained in legal reasoning using inductive logic is not considered a truth, but as a proposition that is most probably true than not. Based on the definitions from Carnap [84] and Fitelson [85], an inductive logic can be formally defined as follows,

$$c\left(C, \{P_1, \ldots, P_n\}\right) \text{ is high iff}$$
$$P\left((P_1 \wedge, \ldots, \wedge P_n) \rightarrow C\right) \text{ is high.}$$

Wherein, $\{P_1, ..., P_n\}$ are set of propositions constituting the premise of an argument, C its conclusion and c being the inductive strength of the argument.

4.4.3 Default Logic

Default logic is an approach designed to handle non-monotonic legal reasoning. While classical logic approaches rely on complete information to obtain a conclusion, default logic approaches obtain a conclusion even when we have incomplete information. The incompleteness may be due to certain decisions that are to be made. Default logic approach uses rules of thumb, called *defaults* to make some plausible conjectures [86].

Defaults are drawn based upon assumptions, and can be naturally used to model *closed world assumption* [87]. A *default* representation of a closed world assumption is as shown below

$$\frac{true : \neg \varphi}{\neg \varphi}$$

Wherein, each ground atom φ is assumed to be false if it does not follow from the axioms on which the application domain is based on. Priorities are assigned to *default*, whenever dealing with one or more defaults for a particular situation. Priority assignment may be static or dynamic [88].

The syntax of a *default*, δ is as shown below,

$$\delta = \frac{\varphi : \psi_1, ..., \psi_n}{\chi}$$

Wherein, φ, $\psi_1, ..., \psi_n$ and χ are closed predicate logic formulae with $n > 0$, φ is called a prerequisite, $\psi_1, ..., \psi_n$ the justifications, and χ being the consequent of the *default* δ.

An example to the application of *default* in the legal domain for an example pertaining to US patent law stating *"An application is granted a patent if, it is novel, non-obvious and useful'* one exception to the above rule is that *" the best mode should not be concealed in the written description of the application"*

$$\frac{novel(X) \wedge nonObvious(X) \wedge useful(X) : grantPatent(X)}{grantPatent(X)}$$

In combination with the rule.

$$concealBestMode(X) \rightarrow \neg grantPatent(X).$$

4.4.4 STIT logic

Sees To It That (STIT) logic or simply 'STIT' is a logic used with logical systems dealings with *agency*. *Agency*, may be defined as a relationship between an *agent* (or group of *agents*) and the actions performed. Both *actions* and *agency* are important notions in the domain of law. Eventhough, STIT is developed for addressing the issues related to *agency*, it is more expressive than few known logics like Alternating Time Temporal Logic (ATL) [89] or Coalition Logic (CL) [90] developed solely for addressing multi-agent systems. The proven 'decidability property' [91] of STIT logic gives it an edge to be used for legal reasoning over other logics.

When handling legal situations like contracts, agreements, negotiations, dialogue or argumentation, normative concepts such as commitment and achievement obligation has to be considered but logically dealing with commitment and achievement obligation is not possible without addressing the notion of *action*, *time* and their relationship. In here, we focus on a branch of STIT known as Temporal - STIT. It is a variant of STIT with time interpreted in standard *Kripke* Semantics and which is capable of handling *action-time* relationships. The notion 'commitment' may be viewed as two different aspects, firstly the propositional commitment, dealing with *'what is true'* and a pragmatic commitment, dealing with *'what is to be done'*. Commitment is related to the notion to achievement obligation (specifically as a *directed achievement obligation*) [92]. A commitment between two agents i and j about a proposition φ may be defined as shown below

$$C_{i:j}\varphi \stackrel{\text{def}}{=} \Box(\neg F^*[i]\varphi \rightarrow G^* v_{i,j}) \wedge \neg[i]\varphi$$

i.e. an agent i is committed to agent j to ensure an action φ iff all historic alternatives in which i will never see to it that φ are histories in which agent i will never fulfill his commitment to agent j and i does not see to it that φ [92].

Wherein, i and j are agents, $[Agt]$. $[i]\varphi$ denotes a formula that captures that fact that φ is guaranteed by a present action of agent i to be read as *'agent i sees to it that φ regardless of what the other agents do'*. $\Box\varphi$, stands for 'φ is necessarily *true*'. G and F are tense operators, $G\varphi$ means 'φ will always be *true* in the future' and F denotes the

dual of the future tense operator G i.e. '$F\varphi$, means φ will be *true* at some point in the future'. Finally, G^* and F^* stands for 'φ is *true* in the present and will always be *true*' and 'φ is *true* in the present and will be *true* at some point in the future' respectively.

The next section introduces to a different non-monotonic logic scheme called as defeasible logic used in defeasible legal reasoning.

4.4.5 Defeasible Logic

Defeasible logic is a non-monotonic logic with proven success in formalizing legal knowledge. The main intuition of the logic is to be able to derive lausible conclusions from partial and sometimes conflicting information. conclusions are tentative conclusions, in the sense that conclusions can be withdrawn when we have new pieces of information. It permits the expression of regulations with an almost one-to-one correspondence between plain language expressions and its encoded form. As it is in our case, having a domain expert annotate the rules of defeasible logic theory with plain English expression during validation phase can further enhance the clarity of a defeasible logic [93].

A reasoning schema for a defeasible reasoning as stated in [94] is as defined below.

"A reasoning schema is defeasible if one should, under certain conditions, refrain from adopting its conclusions through endorsing its premises."

which means, a defeasible reasoning schema can be used when one has no prevailing beliefs against the schema or against adopting its conclusions and when one endorses the premises of a defeasible schema but has such prevailing beliefs to the contrary then one should withdraw the conclusions made by instantiating the schema.

Listing 4.3 shows an example for possible defeasible inferences. From the example we can see that with the addition of more information, the second inference is contrary to the first inference i.e we may say that the second inference is stronger than the first inference. However, in such situations we may not withdraw any premises but we should refrain from deriving conclusions until we have more premises about it. Hence, we also call the defeasible reasoning schema as non-truth preserving [5].

Listing 4.3: Defeasible Inferences

```
1  (a) 'X' is a novel invention.
2  (b)   novel inventions are generally patentable.
3  ---------------------------------------------------
4  (c)   Therefore 'X' is patentable.
5
6  (a) 'X' is a non-patentable subject-matter invention.
7  (b)   non-patentable subject-matter invention are normally not-patentable.
8  -------------------------------------------------------
9  (c)   Therefore 'X' is not-patentable.
```

Defeasible logic theory is a collection of rules to reason about a set of facts/premises to reach to a set of defeasible conclusions [95]. Defeasible logic uses strict rules, defeasible

[5] A reasoning schema, S is truth preserving if all $S's$ instances are truth preserving i.e. if necessarily, whenever $S's$ premises are *true* then also $S's$ conclusions are *true* [94]

rules and undercutting defeaters. Strict rules are those which cannot be defeated or in other words, rules which do not have exceptions. Defeasible rules represent weaker connections which can be defeated and Undercutting defeaters are too weak to support and inference, they specify exceptions to defeasible rules. A formal definition for the defeasible theory and the set of rules used within it are as defined below [95]:

A defeasible logic theory may be viewed as a quadruple $\langle F, R, C, \prec \rangle$ such that

(a) F is a set of formulas,

(b) R is a set of rules,

(c) C is a set of finite set of formulas such that for every formula ϕ

i) $\{\phi, \neg\phi\} \in C$, and

ii) for every $S \in C$ and $A \rightarrow \phi$ in R, if $\phi \in S$, then $A \cup (S - \{\phi\}) \in C$, and

(d) $\prec$ is an binary relation over the set of rules.

Wherein, ϕ is an atomic formula, $\neg\phi$ its complement, A the antecedent of the rule and ϕ (i.e. atomic formula) its consequent. If, $A \in \{\phi\}$ is a set of formulas, we may then define the *strict, defeasible* and *undercutting defeater* rules as:

(a) *strict rule*: $A \rightarrow \phi$.

(b) *defeasible rule*: $A \Rightarrow \phi$.

(c) *undercutting defeater*: $A \rightsquigarrow \phi$.

Another reason for legal rules being formalized in defeasible logic is the defeasibility feature embedded in a legal language (in its conceptual constructions). Defeasibility is introduced in a legal language on purpose to accommodate diverse domain on which such laws are applied (generally done with the use of *unless clause, Explicit exceptions or presumptions*). The big challenge of the use of defeasible logic is its vast knowledge creation to come to a reasonable conclusion when using it in a decision support system.

The next section deals with another type of logic known as *Modal logic*, it is a well suited logic capable of handling legal knowledge to be used in legal reasoning.

4.4.6 Modal Logic

Modal logic is a logic that extends classical propositional logic and predicate logic to include modalities such as alethic, deontic, epistemic and doxastic. The use of modal logic to formalize legal knowledge for legal reasoning was done by philosophers like *Von Wright* and *Kanger*, they showed the use of modal logics to make a formal theory about central legal concepts.

Listing 4.4, shows the general structure of the above mentioned modalities.

Listing 4.4: Modalities

```
Alethic Modalities
   (a)  It is possible that 'p'
   (b)  It is necessary that 'p'. etc..

Deontic Modalities
   (a)  It is obligatory that 'p'.
   (b)  It is permissible that 'p'. etc..

Temporal Modalities
   (a)  It was the case that 'p'.
   (b)  It will always be that 'p'. etc..

Doxastic Modality
   (a)  It is believed that 'p'.

Epistemic Modality
   (a)  It is known that 'p'.
```

The semantics of modal logics are as defined below:

If $\mathcal{G}$ is a non-empty set in a possible world w and $\mathcal{R}$ a binary relation, we define $\mathcal{R}$ as *accessibility* relation (a relationship between two possible worlds), $w\mathcal{R}v$, if non-empty set $\mathcal{G}$ in world v can be accessed from world w using a relation $\mathcal{R}$.

A set of properties can be derived based on the *accessibility* relation.

(a). *Reflexivity*: *An* accessibility *relation is* reflexive *iff, $w\mathcal{R}w$, for every w in $\mathcal{G}$ i.e. for every $\mathcal{G}$ in w, it can be accessed by itself.*

(b). *Symmetry*: *An* accessibility *relation is* symmetric *iff, $w\mathcal{R}u$ implies $u\mathcal{R}w$.*

(c). *Transitivity*: *An* accessibility *relation is* transitive *iff, $w\mathcal{R}u$ and $u\mathcal{R}q$ together imply $w\mathcal{R}q$.*

(d). *Serial*: *An* accessibility *relation is* serial *iff, for each w in $\mathcal{G}$, there is some u in $\mathcal{G}$ such that $w\mathcal{R}u$.*

Further subections discuss in detail the *Alethic* and the *Deontic* modalities of the modal logic

4.4.6.1 Alethic Logic

Alethic Modal Logic or simply Alethic logic is logic of necessary truth and related notions, alethic logic is based on five basic alethic modal operators,

(a) Necessary '$\Box$'

(b) Possible '$\diamond$'

(c) Impossible

(d) Non-Necessary

(e) Contingent

Table 4.2: Defining *'alethic'* modal operators using the *'necessity'* ($\Box$) operator. ('$\sim$' denotes negation)

It is possible that p ($\diamond\, p$)	$\stackrel{\text{def}}{=}$	$\sim \Box \sim p$
It is impossible that p	$\stackrel{\text{def}}{=}$	$\Box \sim p$
It is non-necessary that p	$\stackrel{\text{def}}{=}$	$\sim \Box p$
It is contingent that p	$\stackrel{\text{def}}{=}$	$\sim \Box p \;\&\; \sim \Box \sim p$

A *Contingent* proposition is neither necessarily *true* nor necessarily *false*. It is possible to define (& transform) one *alethic* modal operator using any another operator. Such a possible transformation can be seen in table 4.2, where all the above operators are defined using *necessity* operator ($\Box$). The 'Necessary', 'Impossible' and 'Contingent' modal operators are mutually exclusive operators i.e. every proposition is either one of these operators.

Fig 4.5, depicts these relations using a modal square of opposition [96]. Wherein,

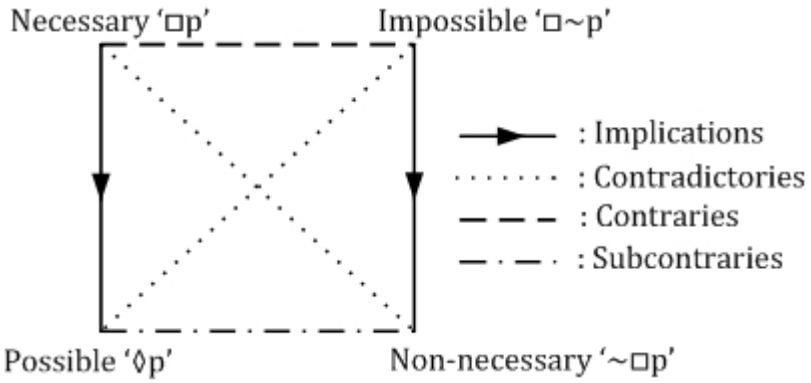


Figure 4.5: Modal Square of opposition (adapted from [96])

propositions are '*contraries*' is they cant both be *true*, '*sub-contraries*' if they cant both be *false* and '*contradictories*' is they always have oposing *truth-values*.

4.4.6.2 Deontic Logic

Deontic Logic is a logic that is concerned with *Obligation*($\boldsymbol{O}$), *Permission*($\boldsymbol{P}$) and related concepts. As like in alethic logic, deontic logic is also built on five normative operators as shown below

(a) Obligation '$\boldsymbol{O}$'

(b) Permission '$\boldsymbol{P}$'

(c) Forbidden '$\boldsymbol{F}$'

(d) Omission 'OM'

(e) Optional 'OP'

Similar to 4.2, it is possible to define (& transform) one *deontic* modal operator using any another operator. Such a definition can be seen in Table 4.3. A Deontic square similar

Table 4.3: Defining *'deontic'* modal operators using the *'Obligation'* ($\boldsymbol{O}$) operator

It is permitted that p	$\leftrightarrow$	$\sim \boldsymbol{O} \sim p$
It is forbidden that p	$\leftrightarrow$	$\boldsymbol{O} \sim p$
It is omissible that p	$\leftrightarrow$	$\sim \boldsymbol{O}\, p$
It is optional that p	$\leftrightarrow$	$\sim \boldsymbol{O}\, p \;\&\; \sim \boldsymbol{O} \sim p$

to the modal square of opposition as shown in Fig 4.5 can be constructed as shown in Fig 4.6. Wherin, the deontic operators '$\boldsymbol{O}$', OP' and '$\boldsymbol{F}$' are mutually exclusive i.e. every proposition is either *Obligatory, Optional* or *Forbidden*. The schema is analogous to the schema seen in alethic-modal operators.

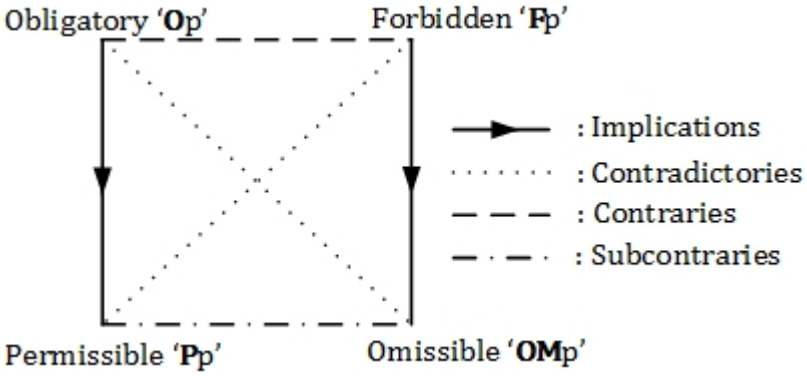


Figure 4.6: Deontic Square (adapted from [96])

Based on the properties described in 4.4.6, there exist several modal classes/systems. Few such systems are as shown below:

(a) **K**: no conditions i.e. $\Box(\mathcal{A} \rightarrow \mathcal{B}) \rightarrow \Box\mathcal{A} \rightarrow \Box\mathcal{B}$

(b) **M/T**: K + Reflexivity i.e. $\Box\mathcal{A} \rightarrow \mathcal{A}$

(c) **B**: K + Reflexivity + Symmetry i.e. $\mathcal{A} \rightarrow \Box \diamond \mathcal{A}$

(d) **S4**: K + Reflexivity + Transitivity i.e. $\Box\mathcal{A} \rightarrow \Box\Box\mathcal{A}$

(e) **S5**: K + Reflexivity + Symmetry i.e. $\diamond\mathcal{A} \rightarrow \Box \diamond \mathcal{A}$

System **S5** simply makes all modal truths necessary, thus making it not suitable for patent norm representation. Further subsections discuss a benchmark and most studied system of deontic logic, Standard Deontic Logic (SDL).

4.4.6.3 Standard Deontic Logic

SDL, is a type of modal logic/system also referred as *KD* or D. A standard *Kripke*-style possible world semantics based *accessibility* relation is assumed. It builds upon the existing propositional logic framework. If p, q and r are propositional variable of a simple propositional language, $\sim$ and $\rightarrow$ are its truth functional operators and $\boldsymbol{O}$ is a deontic modality operator *'obligation'*, then an SDL, can be axiomatized as follows:

Axiom 1. ***TAUT****: All tautologous wffs of the language - Defined by three axiom schemes of propositional calculus,*

Axiom 1.1: $\mathrm{p} \rightarrow (\mathrm{q} \rightarrow \mathrm{p})$.

Axiom 1.2: $(\mathrm{p} \rightarrow (\mathrm{q} \rightarrow \mathrm{r})) \rightarrow ((\mathrm{p} \rightarrow \mathrm{q}) \rightarrow (\mathrm{p} \rightarrow \mathrm{r}))$.

Axiom 1.3: $(\sim \mathrm{p} \rightarrow \sim \mathrm{q}) \rightarrow (\mathrm{q} \rightarrow \mathrm{p})$.

Axiom 2. ***K****:* $\boldsymbol{O}(p \rightarrow q) \rightarrow (\boldsymbol{O}p \rightarrow \boldsymbol{O}q)$ *- If a material condition is obligatory and its antecedent is obligatory, then so its consequent.*

Axiom 3. $\boldsymbol{O}p \rightarrow \sim \boldsymbol{O} \sim p$ *- If p is 'obligatory' only if its negation isn't.*

Rule 1. ***MP****: If* $\vdash p$ *and* $\vdash p \rightarrow q$ *then* $\vdash q$ *- If a 'material condition' and its antecedent are theorems, then so is the consequent.*

Rule 2. ***NEC****: If* $\vdash p$ *then* $\vdash \boldsymbol{O}q$ *- If anything is a theorem, then the claim that the thing is 'obligatory' is also a theorem.*

Based on the kripe-style semantics assumed for a SDL, we can conclude that any theorem of SDL is valid per this semantics (soundness), and any formula valid per this semantics is a theorem of SDL (completeness) [97].

Formal semantics of SDL cannot capture the important legal norm procedural aspect of dependencies and contradictions between rules of similar nature. Semantic Business Vocabulary and Rules (SBVR) uses *alethic* and *deontic* modalities to define legal rules. As to the deontic modality, SBVR, defines rules based only on Obligation modality. This proves to be an inconsistent when using the model proposed by SDL. This can be seen using a trivial example as proposed by Solomakhin in [98].

If:
$\mathcal{S}$ is a SBVR conceptual schema,
$\mathcal{I}$ is the fact population of $\mathcal{S}$,
$\boldsymbol{O}$ is deontic modality for Obigation and
$\boldsymbol{P}$ is deontic modality for Permission
Then,
$\{\boldsymbol{O}p, \boldsymbol{P}(\neg p)\}$ is a set of valid rules under SBVR (conceptual) permitted model.
However,
$\boldsymbol{O}p \equiv \neg \boldsymbol{P}(\neg p)$ is a valid deontic rule
Hence,

> $\mathcal{S}$, the SBVR conceptual schema is inconsistent i.e. it is counter-intuitive to have permitted model under SDL semantics.

Thus, a direct use of semantics from modal logics to formulate legal norms is not sufficient. To handle the alethic and deontic rule defined in SBVR, a new logical flavor, the First Order Deontic Alethic Logic (FODAL), was proposed by Solomakhin in [98].

4.4.6.4 FODAL Logic

FODAL is a multimodal logic- logics whose language contains more than one primitive modal operator and whose axioms define the logical properties of each one of them along with their interaction [99]- as a first-order extension of a combining quantified SDL and S4. It provides the formal semantics for defining the alethic and deontic rules defined in SBVR.

In addition to the axioms defined in 4.4.6.3, the FODAL axiomatization is obtained by combining the axiom systems of S4 and KD and extending it with the additional axioms defining the relations between alethic and deontic modalities. The syntax and semantics assumed to define the FODAL axioms are as shown below:

Language:
The alphabet Σ consists of the following class of symbols:

- $\overline{P}$ an infinite set of predicate symbols $\langle P_1, .., P_n \rangle$.
- $\overline{F}$ a infinite set of function symbols $\langle F_1, .., F_m \rangle$.
- For each P_i respectively each F_j, $arity(P_i)$ resp. $arity(F_j)$ is a non-zero natural number denoting the arity of P_i resp. F_i.
- $\overline{c} = \langle c1, .., c_o \rangle$ is a finite or infinite sequence of constant symbols.
- A collection of variables V which will be denoted by identifiers starting with a capital letter like U,V,X
- Logical connectives / operators: $\neg$, $\wedge$, $\vee$,
- Modal connectives / operators: $\Box$ (alethic necessity), $\diamond$ (alethic possibility), $\boldsymbol{O}$ (deontic obligation), $\boldsymbol{P}$ (deontic permission), $\boldsymbol{F}$ (deontic forbidden).
- Quantifier: $\forall$ (forall), $\exists$ (exists).

A formula ϕ is defined as in FOL with the extension of a set of modal formulas ϕ^{Mod} ($\Box\phi$, $\diamond\phi$, $\boldsymbol{O}\phi$, $\boldsymbol{P}\phi$, $\boldsymbol{F}\phi$) with the additional modal operators ($\diamond$, $\boldsymbol{P}$, $\boldsymbol{F}$) definable in terms of the others:

- $\diamond\phi \equiv \neg\Box\neg\phi$
- $\boldsymbol{P}\phi \equiv \neg\boldsymbol{O}\neg\phi$

- $\boldsymbol{F}\phi \equiv \boldsymbol{O}\neg\phi$

Semantics:
The semantics is defined by a two layered Kripke semantics with augmented bimodal frames consisting of two accessibility relations, R_O and $R_\Box$ between possible worlds.

Definition 1. ***Augmented frame****: A varying domain augmented bimodal frame* $A = \langle W, R_\mathrm{O}, R_\Box, d\rangle$ *consists of a non-empty set,* W*, whose members are possible worlds, two binary accessibility relations,* R_O *and* $R_\Box$*, that hold (or not) between the possible worlds of* W*, a domain function 'd' mapping possible worlds 'w' to a non-empty set 'P' such that if* $d(w, P)$*, then 'P' is true at 'w'.*

As in S4 the alethic accessibility relation is reflexive and transitive [100] and the deontic accessibility relation is serial as in KD [101]. The FODAL semantics additionally defines the bi-modal FODAL frame with the modal formula for the interaction between alethic and deontic logic.

- $\Box\phi \rightarrow O\phi$ (Everything which is necessary is also obligatory)

Definition 2. ***Interpretation and model****: An interpretation* I *in an augmented frame* A *is an interpretation function which assigns to each possible world* w *and each predicate symbol* p *some n-ary relation to the domain* $D(w)$ *of that world. A model* M *is an interpretation of an augmented FODAL frame* A*, if* A *is true wrt to* I*.*

The satisfiability relation between FODAL models and formulae is then defined as shown in definition 3.

Definition 3. ***Satisfiability relation****:* ϕ *is a FODAL formula and* σ *is an assignment to the interpretation* I*, then the relation* $I \models \phi[\sigma]$ *means that* ϕ *is true in* I *when there is a substitute for each free variable* V *of* ϕ *with the value of* $\sigma(V)$*. We omit the definition of the inductive requirements of "*$\models$*" here and refer to [102]. Accordingly, a formula* ϕ *is satisfied by an interpretation* I *(*$I \models \phi$ *) iff* $I \models_\sigma \phi$ *for all variable assignments* σ*.*

Axiomatization:
Following [103] the formalization is given as an axiomatic system in the typical way for a normal modal logic.

Axiom 1. *All S4 and KD tautologies and axioms.*

Axiom 2. *All instances of the Kripke schema:* $\Box(p \rightarrow q) \rightarrow (\Box p \rightarrow \Box q)$ *and* $\boldsymbol{O}(p \rightarrow q) \rightarrow (\boldsymbol{O}p \rightarrow \mathrm{O}q)$

Axiom 3. ***Vacuous*** $\forall$*:* $\forall x\phi \equiv \phi$ *with* x *not being free in* ϕ

Axiom 4. $\forall$ ***Distributivity****:* $\forall x(\phi \rightarrow \psi) \rightarrow (\forall x\phi \rightarrow \forall x\psi)$.

Axiom 5. $\forall$ ***Permutation****:* $\forall x\ \forall y\phi \rightarrow \forall y\ \forall x\phi$.

Axiom 6. $\forall$ ***Elimination****:* $\forall y(\forall x\phi(x) \rightarrow \phi(y))$.

Axiom 7. ***Necessary O:*** $\Box\phi \rightarrow \boldsymbol{O}\phi$

and additionally inference rules

Rule 1. ***Detachment:*** $\frac{\phi \quad \phi \rightarrow \psi}{\psi}$

Rule 2. ***Necessiation:*** $\frac{\phi}{\Box\phi}$ *and* $\frac{\phi}{\boldsymbol{O}\phi}$

Rule 3. $\forall$ ***Generalization:*** $\frac{\phi}{\forall x \phi}$

The FODAL axiom system is *complete* and *sound* with respect to the class of FODAL frames defined above. For proof of soundness and completeness see [102].

Chapter 5, provides a detailed discussion on the extension of FODAL logic for its use in the context of legal rule representation and reasoning.

4.5 Formal Legal Rule Representation

As discussed earlier in 2.4, there exists several formal rule representation languages such as KRIP/L [16], RuleML [17], 2APL [18], NRKL [19], SD-RuleMarkup Language [20], LKIF [104], SWRL [105], LegalRuleML [21] [22] etc.. In this section we discuss a subset of these languages suitable for legal norm representation. In the end, we select one (or a small subset /of) suitable formal rule representation languages for its use in our process defined in 5.2.

4.5.1 KRIP/L

Knowledge Representation and Inference for Procedural Laws (KRIP) is a language specifically built to describe the knowledge for procedural laws [16]. The core idea of KRIP/L (where L stands for Language) is as defined below:

$$\text{Knowledge of Law} = \text{Provisions} + \text{Interpretations} + \text{Control}.$$

Wherein, *Interpretations* = strict analysis of each provision and *control* defines the relation between provisions and selection of proper provisions at a given situation. Provisions were defined using logical natural sentences i.e. logic was used to describe the law. KRIP employs the integration of object concept and extended prolog. It also included convenient functions to describe the relation between provisions. Description of features of a group of objects are called a 'class'. The object which stood for an action occurring in a certain time is called an 'event'. The concepts such as proceedings, inventions, rights are expressed as classes, and provisions are expressed as clauses of extended prolog. For each clause, the identification mark is attached in order to control the inference. The syntax of a class is as shown in the listing 4.5.An example as to depict the use of KRIP/L language is as shown in Listing 4.6[6].

[6] '*self* is bound to the object whose method is being executed.

Listing 4.5: KRIP/L Syntax

```
class <class name>.
   super: <upper class>.
      <slot name>: <value> OR {<class name>}.
            ...
      <message>  - - > <Goal Sequence of Prolog>.
end_class
```

Listing 4.6: KRIP/L Example

```
(a) A person filing_an_application must be a
    person_registered_for_competing_proceedings_before_the_office.
(b) Prerequisites of 'application' are provided in sections 102 and 103 of patent law.
(c) A claim of an application must have a technical_idea.
(d) The effect of the 'application' is provided in the cabinet_order.

      class proceedings.
            super: event
            subject: {person_registered_for_competing_proceedings_before_the_office}.
      end_class

      class filing_an_application.
            super: proceedings.
            claim: {technical_idea}.
            ...
            Legal_check - - > legal(self, section(patent_102)).
            Legal_check - - > legal(self, section(patent_103)).
            effect(X) - - > exec(self, section(cabinet_order)).
      end_class.
```

KRIP/L worked more or less on the PSM layer, which meant there was less opportunity to address the functionalities such as modularization, Context Associations, Isomorphism etc.. Later formal rule representation languages were built addressing the drawbacks of KRIP/L.

4.5.2 SWRL

Semantic Web Rule Language (SWRL), is a web rule language for the semantic web [105]. It embeds sub-languages such OWL (OWL DL and OWL Lite) with those of the RuleML [17]. The model-theoretic semantics of SWRL is a straightforward extension of the semantics for OWL given in the OWL semantics. The basic idea is that we define bindings, extensions of OWL interpretations that also map variables to elements of the domain. A rule is satisfied by an interpretation iff every binding that satisfies the antecedent also satisfies the consequent. It extends the set of OWL axioms to include Horn-like rules, thus enabling the use to horn-like rules to be combined with an OWL knowledge base. SWRL provides a large set of built-in functions addressing the modularization functionality.

Eventhough, SWRL addressed the issue of interoperability, re-usability, extensibility, computational scalability or rules, it puts a restriction on expressiveness provided. SWRL can be translated to FOL, but providing inference service with SWRL is hard, as when supporting the full specification the reasoning becomes undecidable. Approaches to handle the reasoning tasks with a FOL theorem prover has been proposed in [106].

Listing 4.7, shows the use of SWRL for a US patent rule on infringement *"If inventor A infringes an invention B invented by an inventor B, then inventor A has to buy license from inventor B"*.

Listing 4.7: SWRL Example

```
<ruleml:imp>
   <ruleml:_rlab ruleml:href="#example1"/>
   <ruleml:_body>
      <swrlx:individualPropertyAtom  swrlx:property="Infringes">
         <ruleml:var>x1</ruleml:var>
         <ruleml:var>x2</ruleml:var>
      </swrlx:individualPropertyAtom>
      <swrlx:individualPropertyAtom  swrlx:property="Was_Invented_By">
         <ruleml:var>x2</ruleml:var>
         <ruleml:var>x3</ruleml:var>
      </swrlx:individualPropertyAtom>
   </ruleml:_body>
   <ruleml:_head>
      <swrlx:individualPropertyAtom  swrlx:property="Buy_License_From">
         <ruleml:var>x1</ruleml:var>
         <ruleml:var>x3</ruleml:var>
      </swrlx:individualPropertyAtom>
   </ruleml:_head>
</ruleml:imp>
```

4.5.3 LKIF

Unlike others rule representation formats, Legal Knowledge Interchange Format (LKIF) [7], is an XML schema built bottom up to represent legal theories and arguments constructed from theories. LKIF was developed under the EU project ESTRELLA. The main purpose of it is to serve as an interchange format which is easy to translate to and from other formats. Legal rules in LKIF are modeled as defeasible inference rules. LKIF format can be used to represent both propositional and first-order logic formulas. *Terms, Atomic formulas* and *compound formulas* form the building blocks of a legal rule in LKIF format.

LKIF not only supports for the representation of legal rules but it tries to accommodate how to reason legally about the situations and actions covering such a legal rule. Rule conflicts are solved by the use of rule priorities using meta-level principles such as **lex-superior** and **lex posterior**. It also supports isomorphic modeling of legislation [107]. The syntax of a legal rule in the LKIF format is as shown below in Listing 4.8.

Listing 4.8: LKIF rule syntax

```
   Rule = element rule {
      attribute id {xsd:ID},
      attribute strict {xsd:boolean}?,
      Head, Body?
      }

      Head = element head {Wff+}
      Body = element body {Wff+}
```

Figure 4.7, shows the schema for the LKIF representation format. Wherein,

- **Sources**: Is used to associate major elements used in the LKIF model to its XML resources on the World Wide Web. This enables to links the legislation sources to its constructed legal rules.

[7] LKIF is composed by two parts: ontology and rule language. In here, only the rule part is considered for the discussion

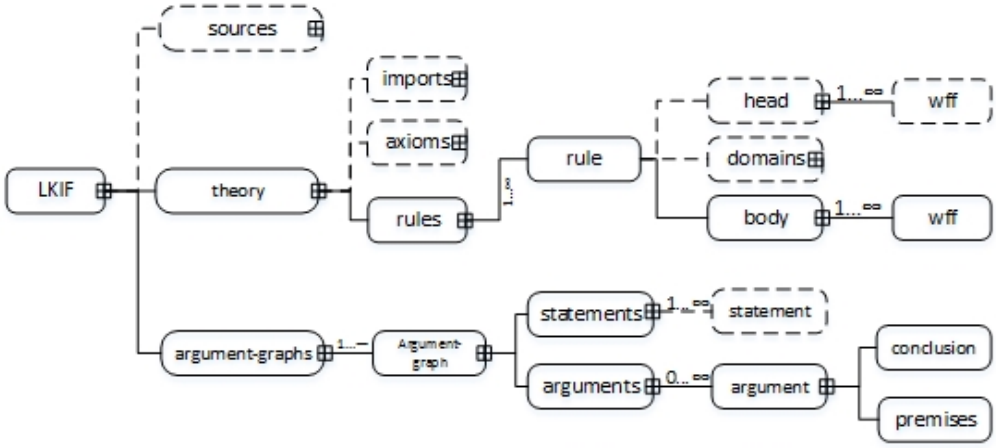


Figure 4.7: LKIF Schema

- **Theory**: Theory refers to a set of propositions. *Axiomtization* is used to represent an infinite number of propositions in a finite text. A *theory* may import propositions from other XML files on the web and a *theory* is completed by a set of inference rules with Well formed formulas.

- **Argument Graphs**: Conceptual model of Argument Interchange Format (AIF) [108] is used as a basis for the LKIF argument model. The argument-graph further consists of one or more *statement* elements and zero or more *argument* elements. Arguments by default are considered as pro arguments. The *premise* and *conclusion* element of the arguments reference a *statement* element defined in the same argument graph.

4.5.4 LegalRuleML

LegalRuleML [21] [22], is a new standardization effort under OASIS with an aim to produce a rule language representation format for representing legal norms. It supports multiple semantic annotations, wherein each legal annotation can represent a different pragmatic involved with it. As a part of provenance information, legalRuleML provides the necessary information for identifying the relationship between the fragments of the legal rules to its creators. RuleML is an XML based language for representing rules. The hierarchical structure of RuleML comprises of reaction rules, transformation rules, derivation rules, facts, queries and integrity constraints. RuleML's XML specification is built based on XML schema modules, which provides both *flexibility* and *extensibility* functionality to RuleML. The *extensibility* functionality of RuleML provides with an option to embed/re-use modules of RuleML inside LegalRuleML. LegalRuleML is positioned between the deliberation rules and the reaction rules facilitating the modeling of either norms or business rules. Figure 4.8 shows the positioning of LegalRuleML inside the RuleML architecture. LegalRuleML addresses the all the drawbacks of the previous rule representation formats in the domain of law. The possibility to link and re-use modules of LegalDocML and RuleML inside a legalRuleML makes it an appropriate choice for

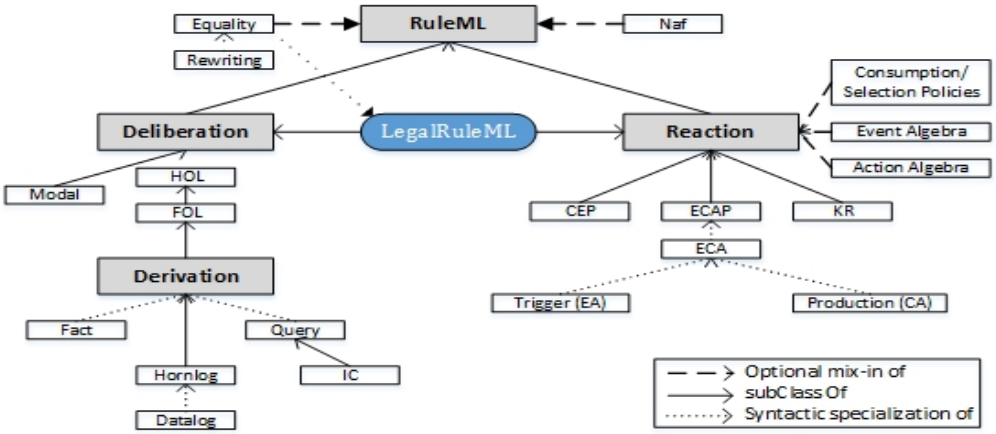


Figure 4.8: LegalRuleML position inside RuleML architecture (adapted from [])

legal rule representation format. A detailed description of the LegalRuleML, legal rule representation format is described in chapter 5.

4.6 Logic Programs

Generally, a logic program is a finite set of rules. Each rule r is of the form:

$$\alpha_1 \vee \cdots \vee \alpha_k :- \beta_1, \ldots, \beta_m, not\ \beta_{m+1}, \ldots, not\ \beta_n.$$

where $\alpha_1 \vee \cdots \vee \alpha_k$ are atoms, and $k \geqslant 1, n \geqslant m \geqslant 0$. The disjunction of $\alpha_1 \vee \cdots \vee \alpha_k$ is the head of r, and the conjunction of $\beta_1, \ldots, \beta_m, not\ \beta_{m+1}, \ldots, not\ \beta_n$ is the body of r. Let the set of the head literals be denoted by $H(r)$ and the body of r be denoted by $B(r)$. Also, let the set of positive and negative body literals be denoted by $B+(r)$ and $B(r)$, respectively.

Furthermore, an *atom* is a formula $p(t_1, ..., t_n)$, where p is a predicate symbol of arity n (n $\geqslant$ 0). A literal is either an atom or of the form *not* where is an atom. Each argument of an atom t_i is a term, which is either a variable or a function term with a form of $f(t_1, ..., t_k)$. f is a function symbol of arity k (k $\geqslant$ 0), and each t_i is a term. A functional term with arity zero is a *constant*.

Based on different classes of rules, there are different types of logic programs:propositional logic programs, Datalog logic programs, definite logic programs, stratified logic programs, normal logic programs, extended logic programs, disjunctive logic programs and combinations of classes, as shown in Figure 7.2

Propositional Logic Program

A propositional logic program consists of simple propositional clauses. All literals of a clause are propositional ones without no variables, quantifiers and functions, and thus propositional logic programs provide a limited expressiveness.

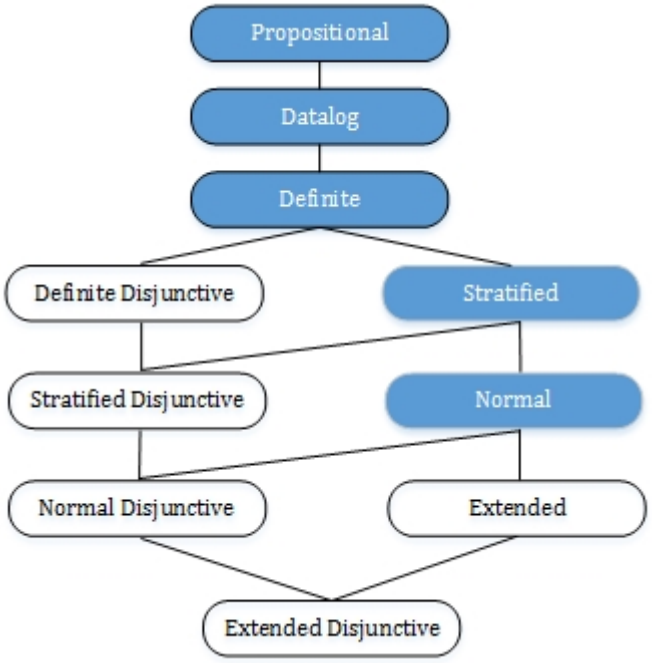


Figure 4.9: Classes of Logic Programs

DataLog

Datalog is a light weight declarative logic programming language. Its is more often Datalog is often used as a query language for deductive databases. Moreover, Datalog is function-free, i.e., all functional terms appearing in a Datalog program are constants. In addition, variables in the head of a rule must appear in its body.

There are three main kinds complexity connected to plain Datalog [109]

(a) The data complexity is the complexity of checking whether $D_{in} \bigcup P \mid = A$ when Datalog programs P are fixed, whereas input databases D_{in} and ground atoms A are an input

(b) The program complexity (also called expression complexity) is the complexity of checking whether $D_{in} \bigcup P \mid = A$ when input Databases D_{in} are fixed, whereas Datalog program P and ground atoms A are an input.

(c) The combined complexity if the complexity of checking whether $D_{in} \bigcup P \mid = A$ when input databases D_{in}, Datalog programs P, and ground atoms A are an input

Definite Logic Program

A definite logic program is a set of rules of the form:

$$\alpha :- \beta_1, \ldots, \beta_n. \qquad (n \geqslant 0)$$

where α and $\beta_1, \ldots, \beta_n$ are all positive atoms.

A finite logic program goes beyond Datalog programs by allowing functions, which provide the ability of handling finite sets of constants [110].

Normal Logic Program

As seen before, a literal is either an atom or a negated atom. A normal clause is a rule of the form as shown in 4.6. A normal logic program is a finite set of normal clauses. I.o.w, a normal logic program has a similar form with definite logic program, but each body literal can be either positive or negative:

$$\alpha :- \beta_1, \ldots, \beta_m, not\ \beta_{m+1}, \ldots, not\ \beta_n. \qquad (n \geqslant m \geqslant 0).$$

Stratified Logic Program

Stratification is a constraint usually placed on logic programs to rule out negation wrapped inside recursion. Sergot [111], defines stratified logic program as follows:

A normal logic program $\mathcal{P}$ is stratified when there is a partition

$$\mathcal{P} = \mathcal{P}_0 \bigcup \mathcal{P}_1 \bigcup \ldots \mathcal{P}_n \qquad (\mathcal{P}_i\ and\ \mathcal{P}_j\ disjoint\ \forall\ i \neq\ j).$$

Such that, for every predicate p

(a) the definition of p is contained in one of the partitions/strata $\mathcal{P}_i$

and, for each $1 \leqslant$ i $\leqslant$ n;

(b) If a predicate occurs *positively* in a clause of $\mathcal{P}_i$, then its definition is contained within

$$\bigcup_{j \leqslant i} \mathcal{P}_j$$

(c) If a predicate occurs *negatively* in a clause of $\mathcal{P}_i$, then its definition is contained within

$$\bigcup_{j < i} \mathcal{P}_j$$

While there exists several other flavor of logic programing such as extended logic program, disjunctive logic program, stratified disjunctive, definite disjunctive etc.. this falls out of the scope of this thesis.

4.7 Summary

The chapter provided a detailed discussion on the existing works in the domain of legal knowledge representation and reasoning. The related works were discussed based on a set of requirements.

The chapter first introduced, to a set of general requirements for any knowledge representation structure grouped based on representational adequecy, inferential adequecy, representational efficiency criteria's. In addition, the chapter introduced to a set of requirements specific for legal knowledge representation.

The chapter provided a detailed discussion on the existing document markup languages, such as EnAct, EUR-Lex, MetaLex and Akoma Ntoso. Wherein, each markup language was discussed with its pros and cons based on the requirements presented previously.

Later, the chapter discussed, a set of semi-formal legal representation structures, which were based on controlled natural language approach. In here, the chapter discussed, various CNL's such as ACE, Drafters language, Massachusetts legislative drafting language etc.. with its advantages and disadvantages over others from a legal knowledge representation perspective.

The chapter introduced to a series of existing legal rule logical formalisms. Different logics such as Deductive, Inductive, Default, Defeasible and Modal logics. The thesis further discussed a subset of modal logics, the Deontic, Alethic and FODAL logic for representing legal rules.

Based on the requirements that were described at the beginning of the chapter, formal legal rule representation languages such as KRIP/L, SWRL, LKIF and LegalRuleML were discussed. A discussion on a subset of logic programming languages, which were used in the domain of legal rule representation was presented. As a whole, this chapter provided a detailed discussion on the existing technologies/approaches for representing legal rules.

Chapter 5

Patent Information System KR Framework

Contents

Reasoning elementary patent norms requires representing them on different knowledge representation layers. Eventhough, as seen in previous Chapter 4, there exist few minor related works in this area, none of them deal directly with using it in the form of a process, starting from a natural language legal text to formal rule representation with a key foresight that its a patent/legal practitioner who is the end user and not a knowledge modeler.

5.1 Conceptual patent information system KR framework

A conceptual patent information system KR framework envisioned is as shown in Fig 5.1. The framework may be divided into three interdependent main modules, *'Environment'*, *'Representation'* and *'Foundation'* modules. Each of the main modules are further divided

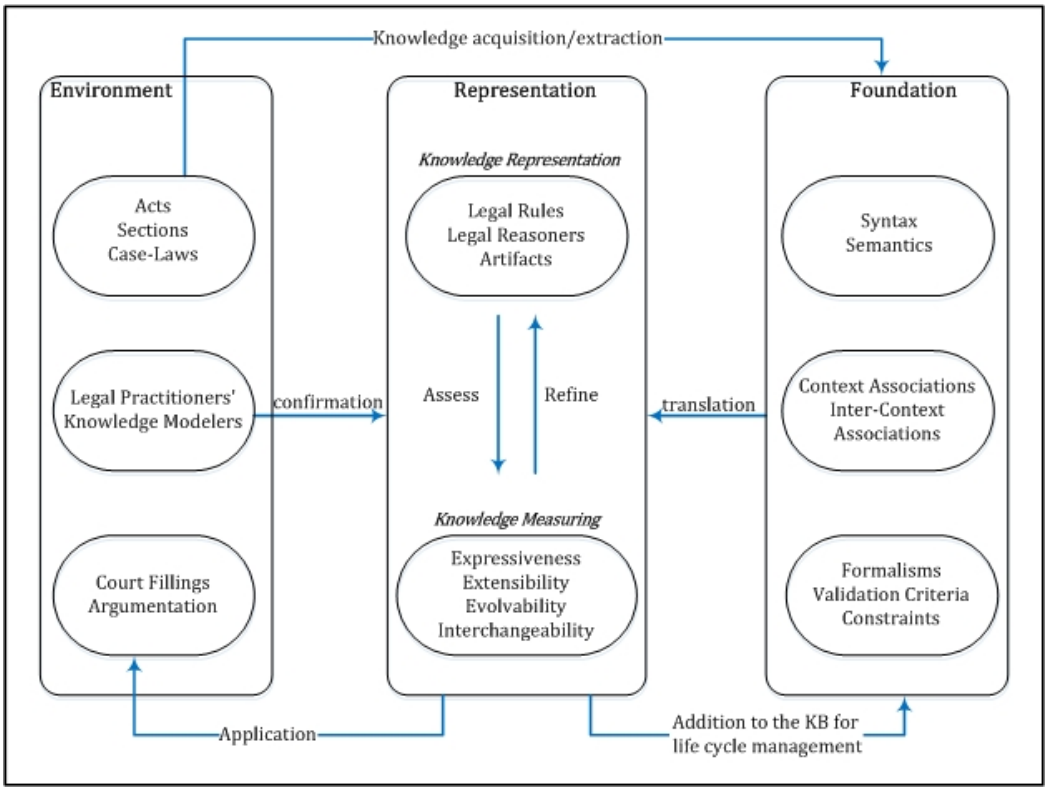


Figure 5.1: Conceptual patent information system KR framework

into sub-modules. The *Environment* module is conceptually divided into modules pertaining to *'source', 'user'* and *'use-case'*. The *'Source'* sub-module comprises of all possible legal sources e.g. Acts, Sections or Case-laws pertaining to a legal norm. The *'user'* sub-module refers to the users, user may be an end user - *legal practitioners'* or technical users -*Knowledge Modeler*. *Court Fillings, In-court Argumentation, etc..* constitute the *'use-case'* sub-module.

Similar to the *'Environment'* module, the *'Foundation'* module also comprises three conceptual sub-modules, Knowledge building-block, pragmatics and methodologies sub-modules. *'Syntax' and 'Semantics'* for the basic raw data required for any knowledge building process. The context is provided by the pragmatic sub-module. They provide the association rules /relations between the raw data collected. Such Associations may be inter-context associations. Several methodologies for validating such stored associations between the raw-data is provided by the *Methodologies* sub-module.

The third module is the *'Representation'* module, this is responsible for representation and evaluation of such representation. The knowledge from the *'Foundation'* module is translated into artifacts such as rules (simple / complex), which are then reasoned using a legal reasoner. The knowledge in the form of rules, are measured for its aspects such as *'Expressiveness', 'Extensibility' etc...* The directional arrow show the conceptual information flow between modules and sub-modules. The rules and/or results of reasoning such rules are confirmed by its user depicted by an arrow from the *'user'* sub-module to the *'representation'* module. Later such confirmed rules or results of reasoning on such rules are then applied onto different application, the information from the *'Representation'*

modules is also sent back to the *'Foundation'* module for its life-cycle management.

Using this conceptual framework as a background, this thesis proposes a process for knowledge representation, transformation and reasoning of patent norms. The process proposed (with the help of the system) helps in reducing such gaps concerning the understanding of legal knowledge. The process provides efficient platforms through which a legal practitioner can provide the required legal information pertaining to a section in a simpler form as required by the modeler.

This chapter initially explains such a process using a generalized process modeling technique, independent of any associated technologies. The use of a generalized process model provides an overview on the layers, tools and technologies associated in disaggregation of patent norms and their pragmatics, and reasoning on top of them.

The key concept in this process is a modularization approach, which can be easily understood, adapted and finally used by a patent/legal practitioner. Fig 3.3, in Chapter 3 depicted such a key concept in its simplest form. Every patent norm/rule constructed by an authorized body like *congress* has three basic building blocks, the rule, the meaning of the rule and the context for which such a rule was formed.

5.2 Generalized Process Model

Fig 5.2, shows a generalized process model for knowledge representation and reasoning of elementary patent norms with elementary patent pragmatics. The process is open ended, such that, the input to the process may be a legal section from any NPS and the output from the process can be used as an input into any existing legal argumentation system.

As discussed earlier, a legal section from any NPS e.g. § 112 of US patent law or § 69 of EPC etc. can be considered as an input to the process model. Such legal sections are then annotated using well known legal document markup standards. A comparison of such standards and its adaption to fit this process is discussed in chapters 4 and 5. In addition to annotating the legal sections, additional information pertaining to the considered legal section in the form of judgments, opinions, amicus briefs etc. are annotated. Such additional information only provide the additional pragmatics needed to understand the considered legal section and does not change the legal section itself. Legal sections are changed through regulations or acts. Hence, the legal section with its annotated metadata form the core and the additional related forms the cladding. Identification of all associated information through annotation helps in the life-cycle management of legal information i.e. the legal section considered.

Even with the annotation of legal section and its related information, automated reasoning on top of it is very difficult due to the vagueness involved in it. Generally, laws/legal sections are designed to be vague and its vagueness is to accommodate different possible scenarios under which such a law can be applied. Pragmatics is an important aspect in legal domain, it explains the context in which such laws are being applied.

The next step in the process is to deal with the disaggregation of a law/legal sections into elementary norms with Elementary Pragmatics, 'EP' which provides the required degree of separation (i.e. elementary concern) sufficient enough to minimize the vagueness and

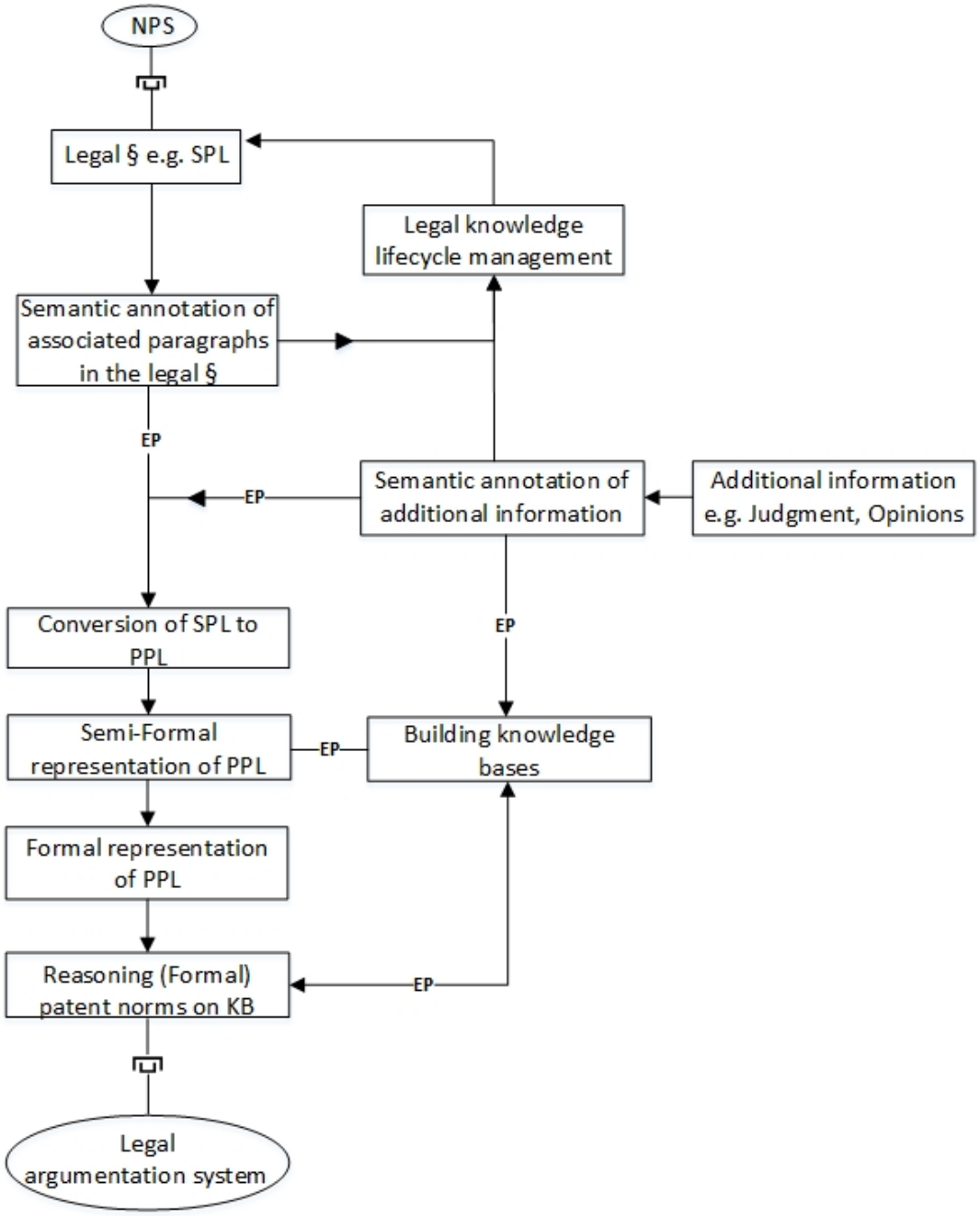


Figure 5.2: Generalized process for obtaining elementary norms with EPs.

thereby making the legal language simple. For the purpose of this disaggregation, legal sections under SPL are converted into PPL using decision models. Such a process of converting an SPL into a decision model requires interaction by a person skilled in legal knowledge, like a legal practitioner.

Minimizing such vagueness directly results in a legal language which is more simple than before. Thereby, providing enough information in an elementary and/or simpler form. The simple legal norms are still far away from it to be used in a legal reasoner. The

next step is to transform such PPL norms into a semi-formal knowledge representation format. We make used of an CNL approach for the semi-formal representation. A CNLs is a subset of natural language that can be accurately and effectively processed by a computer but is expressive enough to allow natural usage by a non-specialist.

CNL based semi-formal patent norms on a CIM layer are further transformed into a PIM formal rule representation format. A formal rule representation format on the PIM layer provides the necessary rule modularization and interchange feature for its re-use in other contexts.

Parallel to the process, a Knowledge Base (KB) is built and updated based on the EPs obtained during each step of annotation and transformation. For the purpose of reasoning, the formal elementary patent norms transformed into a PSM format to be (semi-/) automatically reasoned on against the elementary patent pragmatics within the knowledge base.

Such elementary patent norms with its elementary pragmatics can be further used as an input into any existing legal argumentation system.

In the following sections, we will instantiate the proposed process of obtaining and formally representing elementary norms in the context of elementary pragmatics by means of a running example, which is Paragraph 1 of Section 112, of the United States Patent Law, dealing with the patent enablement.

5.3 Example

35 U.S. Code § 112 - Specification

(a) In General –The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same, and shall set forth the best mode contemplated by the inventor or joint inventor of carrying out the invention.

5.4 Legal Document Annotation

The first step in the process is to semantically annotate the legal documents for contextual information. As seen in subsection 4.2.4, we use Akoma Ntoso or Legal Document Markup Language (LegalDocML), a standard under OASIS for managing legal documents. It provides a systematic mechanism for referencing documents based on URIs through sound ontological approach by designing metadata and relationships between documents and different versions of documents. The possibility to define a common format for most parliamentary activities. Such possible document type handled by LegalDocML can be seen in Figure 5.3.

Figure 5.3: Document-types handled by LegalDocML

Wherein; <**documentType**> defines the type of document under consideration. Some of the possible types of document of our interest in patent domain are, <**legislativeDocs**>, comprising of *'acts' and 'bills'*, <**debateDocs**>, comprising of *'reports', debateRecords*, etc.., <**amendmentDocs**> related to ammendments to any acts/bills/sections etc.., <**judgementDocs**>, pertaining to judgements, amicus briefs, cert petitions etc.. and other documents pertaining to any parliamentary activities concerning a law.

To understand the structure of LegalDocML, this thesis focuses on <**act**> and <**judgement**> elements. The <**act**> element, used for describing the structure and content of act can be further hierarchical subdivided into several classes as shown in Figure 5.4. The <**meta**> element, is used to capture all the meta-information concerning a *'act' or 'judgement'*, the <**preface**>, defines document type (<docType>), document title (<docTitle>), document number (<docNumber>), date of assent (<docDate>), table of contents (<toc>). The <**preamble**> element, defines *'citations'* i.e. list of other legislation resources that empower the current document and *recitals*, that provide the justification and motivation. Finally the <**Conclusions**> element describes the *'Signature', 'date of signature', 'place of signature'* etc...The <**judgement**> element, comprises of self descriptive sub-elements such as meta-information (<**meta**>), content of the judgement (<**judgementBody**>), verdict (<**conclusions**>), attachments pertaining tot he judgment (<**attachments**>) etc.. Figure 5.4 depicts the structure of the <**judgement**>.

Further, the <**meta**> element is divided into several subclasses for defining identification (<identification>) using <**FRBR**> vocabulary, publication, classification (<classification>), lifecycle management, workflow, temporal data (<temporalData>), references (<references>), proprietary and presentation information pertaining to a legal document under consideration. The structure of the <**meta**> element is as shown in Figure 5.5 We use LegalDocML, to annotate the legal section described above. We split the annotation into two parts, the context and the content. The context comprises meta-data dealing with the context

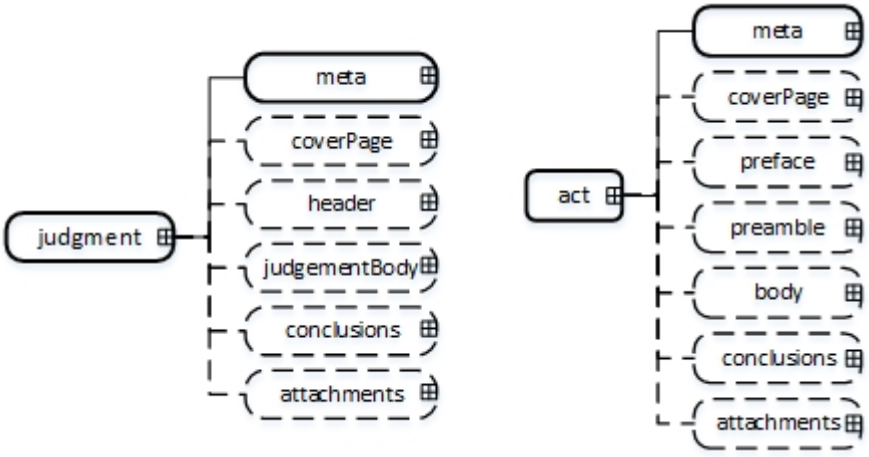


Figure 5.4: 'Judgement' and 'Act' element hierarchical structure in LegalDocML

information of the legal section and the content deals with the actually legal paragraph itself. The XML listing 5.1 and 5.2, shows the annotation of Section 112 1^{st} Paragraph [1].

Listing 5.1: Context annotation of first paragraph of § 112

```
1 <?xml-model href="schema.xsd" type="application/xml" schematypens="http://purl.oclc.org/
    dsdl/schematron"?>
2 <akomaNtoso xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
3   <act contains="originalVersion">...</act>
4    <!--Meta data describing the source, author, country etc.. of the legal section-->
5     <meta>
6         <identification source="#LII">
7             <FRBRWork>
8                 <FRBRthis value="akn/us/codes;us/main"/>
9                 <FRBRuri value="akn/us/codes;us/main"/>
10                <FRBRdate date="1946" name="creation"/>
11                <FRBRauthor href="#congress" as="#author"/>
12                <FRBRcountry value="us"/>
13                <FRBRlanguage language="eng"/>
14            </FRBRWork>
15            <FRBRExpression>
16                <FRBRthis value="akn/us/codes;us/patentlaw/main#title35/"/>
17                <FRBRuri value="akn/us/codes;us/patentlaw/main#title35/"/>
18                <FRBRdate value="2014-03-26" name="Generation"/>
19                <FRBRauthor href="#LII" as="editor"/>
20                <FRBRlanguage language="eng"/>
21            </FRBRExpression>
22            <FRBRManifestation>
23                <FRBRthis value="akn/us/codes;us/patentlaw/main#title35/␣.akn-html"/>
24                <FRBRuri value="akn/us/codes;us/patentlaw/main#title35/␣.akn"/>
25                <FRBRdate value="2014-03-26" name="Generation"/>
26                <FRBRauthor href="#FUB" as="author"/>
27            </FRBRManifestation>
28        </identification>
29        <lifecycle source="#FUB">
30            <eventRef source="#ref1" id="e1" type="generation" date="1946"/>
31                                .....
32                                .....
33            <eventRef source="#ref8" id="e8" type="enforcement" date="2012-09-16"/>
34        </lifecycle>
35        <temporalData source="#FUB">
36            <temporalGroup is="#t1">
```

[1]Only important aspects of the annotation is highlighted here. Complete annotation of the legal section can be found under Annex A

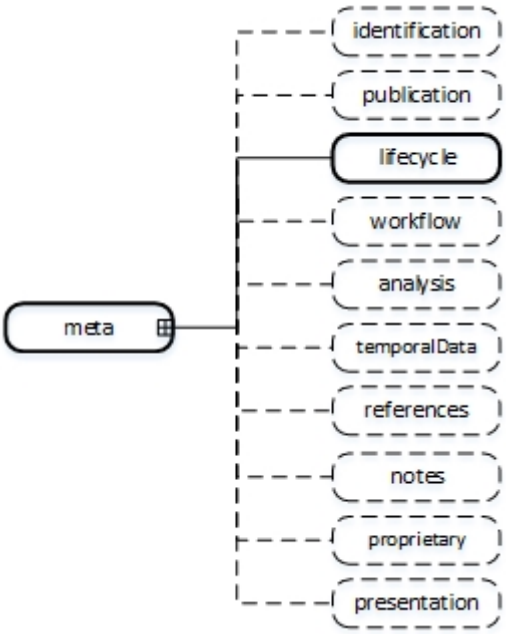


Figure 5.5: 'Meta' element hierarchical structure in LegalDocML

```
37              <temporalInterval refersto="#inforce" start="e2"/>
38              <temporalInterval refersto="#efficacy" start="e2"/>
39          </temporalGroup>
40                      .....
41                      .....
42          <temporalGroup is="#t4">
43              <temporalInterval refersto="#inforce" start="e8"/>
44              <temporalInterval refersto="#efficacy" start="e8"/>
45          </temporalGroup>
46      </temporalData>
47      <references source="#FUB">
48      <original href="/us/codes;us/eng/patentlaw/#title35" showAs="Title␣35,␣US␣Code"
            id="ref1"/>
49      <passiveRef href="/us/codes/1965-10-24/main" showAs="Pub.␣L.␣89  8 3 " id="ref4"/>
50      <passiveRef href="/us/codes/1978-01-24/main"showAs="Pub.␣L.␣94  1 3 1 " id="ref6"/>
51      <passiveRef href="/us/codes/2012-09-16/main"showAs="Pub.␣L.␣112  2 9 " id="ref8"/>
52      <TLCOrganization id="congress" href="/ontology/organizations/congress/"
53              showAs="US␣Congress"/>
54       <TLCOrganization id="LII" href="/ontology/organizations/LII/"
55              showAs="Cornell␣University"/>
56       <TLCRole id="editor" href="/ontology/roles/editor"showAs="Editor"/>
57       <TLCRole id="author" href="/ontology/roles/author"showAs="Author"/>
58       <TLCPerson id="FUB" href="/ontology/person/editors/FUB"
59              showAs="Free␣University␣of␣Berlin"/>
60      </references>
61    <notes source="#FUB">
62      <note id="#n1">
63          <p>
64              Leahy-Smith America Invents Act: First to file policy.
65          </p>
66      </note>
67    </notes>
68  </meta>
```

Listing 5.2: Content annotation of first paragraph of § 112

```
1    <preface>
2       <block name="preface">
```

```
3          <docTitle id="title">United States Code </docTitle>
4       </block>
5    </preface>
6    <body>
7       <title id="tit35">
8          <num>Title 35</num>
9          <heading>PATENTS</heading>
10         <section id="tit35-112" period="#t4">
11            <num> 112 </num>
12            <heading>Specification</heading>
13         </section>
14         <clause id="112-a">
15            <num>(a)</num>
16            <noteRef href="#n1"/>
17            <heading>In General.</heading>
18            <list id="tit35-sec112-par1">
19               <content>
20                  <p> - The specification shall contain a written description of the
21                     invention, and of the manner and process of making and using it,
22                     in such full, clear, concise, and exact terms as to enable any
23                     person skilled in the art to which it pertains, or with which it
24                     is most nearly connected, to make and use the same, and shall
25                     set forth the best mode contemplated by the inventor or joint
26                     inventor of carrying out the invention. - </p>
27               </content>
28            </list>
29         </clause>
30      </title>
31   </body>
32 </akomaNtoso>
```

In addition to capturing (by annotation) the legal information from legal sections/paragraphs, we also annotate the landmark decisions such as, *In re Ruschig Fed Cir and Pfizer Inc. v. Teva Pharmaceuticals Inc* pertaining to this legal section. These additionally annotations capture the pragmatic context in which such a law section has to be applied. I.o.w, it defines new pragmatics, in terms of understanding and commitment, encompassing the legal section. Annotation of *'In re Ruschig Fed Cir'* is as shown in Listing 5.3 [2].

Listing 5.3: Annotation of judgment pertaining to first paragraph of § 112

```
1 <?xml-model href="schema.xsd" type="application/xml" schematypens="http://purl.oclc.org/
     dsdl/schematron"?>
2 <akomaNtoso xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
3    <judgement contains="singleVersion">
4         <meta>
5             <identification source="#SKGF">
6                 <FRBRWork>
7                     <FRBRthis value="/us/judgement;us/main"/>
8                     <FRBRuri value="/us/judgement;us/main"/>
9                     <FRBRdate date="1965" name="Hearing"/>
10                    <FRBRauthor href="#rich" as="#author"/>
11                    <FRBRcountry value="us"/>
12                    <FRBRlanguage language="eng"/>
13                </FRBRWork>
14                <FRBRExpression>
15                    <FRBRthis value="us/judgement;us/patentlaw/main#title35/"/>
16                                ...
17                    <FRBRauthor href="#author1" as="editor"/>
18                    <FRBRlanguage language="eng"/>
19                </FRBRExpression>
20                <FRBRManifestation>
```

[2]Only important aspects of the annotation is highlighted here. Complete annotation of the legal section can be found under Annex A

```
21                  <FRBRthis value="us/codes;us/patentlaw/main#title35/␣.akn-html"/>
22                                  ...
23                  <FRBRauthor href="#FUB" as="author"/>
24              </FRBRManifestation>
25            </identification>
26            <publication date="1965" name="Judgement" showAs="Fer␣Cir␣Judgment" number="
                  910"/>
27            <lifecycle source="#SKGF">
28                <eventRef date="1965" is="e1" source="ro1" type="Hearing"/>
29            </lifecycle>
30            <analysis source="#SKGF">
31                <judical> <result type="approve"/> </judical>
32            </analysis>
33            <references source="#SKGF">
34             <original id="ro1" href="ak/judgement/1965/eng /main" showAS="Original"/>
35                  ...
36         <TLCRole id="Respondent" href="/../Respondent" showAs="Respondent"/>
37            </references>
38            <notes source="#SKGF">
39                <note id="#n1">
40                    <p> - Judge Rich concurred, believing that ...  was subject to a ...
                          non-complex cases - </p>
41                </note>
42                <note id="#n2">  ... </note>
43            </notes>
44        </meta>
45        <header>
46             <p class="judgementNumber">
47                <span class="preface">
48                   <docketNumber> 379 F.2d 990 </docketNumber>
49                </span>
50                 <neutralCitation>
51                  In re Ruschig, 379 F.2d 900, 15 USPQ 118 (CCPA 1967)
52                 </neutralCitation>  </p>
53            <ref id="ref01" href="ak//us/codes/1965/main">
54                 <title id="tit35">
55                      ...
56                    <clause id="112-a">
57                        <noteRef href="#n1"/>
58                        <heading>Enablement</heading>
59                    </clause>
60                </title>
61            </ref>
62         <p class ="judges">
63            <judge id="jud01" refersTo="#rich"> Judge Rich </judge>
64            <judge id="jud02" refersTo="#Almond"> Judge Almond </judge>
65         </p>
66            <courtType id="#court1" refersTo="#FerCir"/>
67         <p class ="parties">
68            <party id="p1" refersTo="#ruschig" as="#Appellant"/>
69            <party id="p1" refersTo="#FedCir" as="#Respondent"/>
70         </p>
71        <summary>
72            <noteRef href="#n1"/>
73            <noteRef href="#n2"/>
74            <p> - While we have no doubt a person . . . the specification discloses the
                   .. actually invented     - </p>
75        </summary>
76        </header>
77         <conclusion>
78            <p class="signature">
79            <judge id="jud01" refersTo="#rich">Judge Rich</judge><eol/>
80            ...
81            <span class="signature"> CHIEF JUSTICE </span> </p>
82         </conclusion>
83    </judgement>
84 </akomaNtoso>
```

For ease of understanding, few instantiations from the annotations shown in List-

ing 5.1, 5.2 and 5.3 relevant to first paragraph of § 112 and judgment respectively are shown in table 5.1.

Table 5.1

Act			
	↳ source	→	LII
	author	→	Congress
	lifecycle	→	1946 - 2011.09.16
	note	→	AIA: First to file policy
	docTitle	→	35 United States Code
	etc...		
Judgment			
	↳ source	→	Sterne kessler Goldstein Fox
	docTitle	→	In re Ruschig
	docNumber	→	379 F.2d 990
	#Appellant	→	Ruschig
	#Respondent	→	Judge Rich
	judge	→	Judge Almond
	signature	→	Chief Justice
	etc...		
etc...			

5.5 Legal Decision Model

The process of disaggregation extracts elementary concerns from the compound concerns of the statutory and transforms them from their vague SPL semantics into a concrete PPL semantics, i.e., a legal norm representation (syntax) which has a concrete meaning (semantics) in PPL is understood and evaluated (pragmatics) within the context of case law. In the US patent law, the USPTO, uses a standard patent evaluation procedure provided in the MPEP. We transform such procedures into legal decision models, wherein, each decision point is a single procedure or a set of procedures to be carried out. As discussed earlier, whenever there is a landmark decision pertaining to a particular paragraph, like, as in our case, with the *In re Ruschig Fed Cir and Pfizer Inc. v. Teva Pharmaceuticals Inc* decision, it defines a new way to understand and apply that specific part of the 1^{st} paragraph of legal section 112 (i.e., it defines a new branch/decision point or re-branches from a decision point in a manner different than before).

Figure B.1, such a knowledge transformation process of disaggregation. The landmark decisions pertaining to the legal section under consideration are integrated into the decision model to obtain the latest understanding of the decision model through PPL. This allows capturing the different interpretations of the same section in different case laws. The context information for the legal section and its related judgments, which have been annotated before, are added as meta information to the decision model.

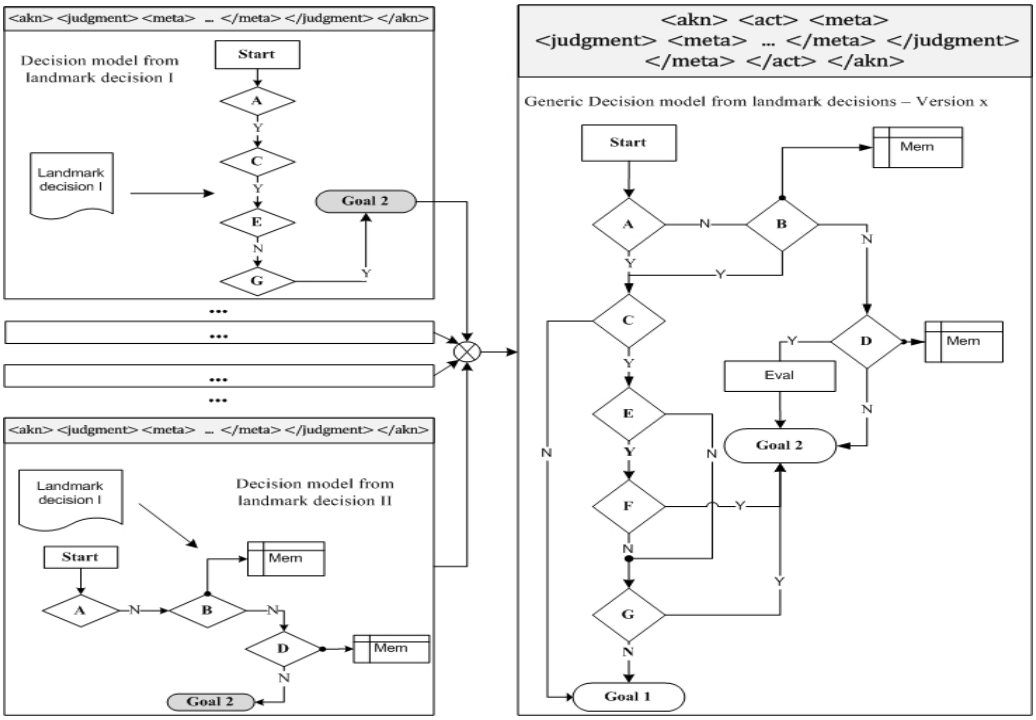


Figure 5.6: Decision model with the <meta> information about legal section as overhead(adapted from [112]).

Further, we use the easy to understand decision models as basis for writing the legal norms and their elementary concerns in terms of constitutive vocabulary definitions and prescriptive behavioural legal rules in SBVR's Structured English. We can classify the mapping relationships as 1:1- wherein each decision is mapped into a single SBVR rule, 1:M- where, a single decision is mapped into many SBVR rules or an N:M relationship. The core idea of SBVR to work on a conceptual level provides the required platform for legal domain experts and trained formal knowledge engineers to work together in a formalization process using Structured English as common computational independent knowledge representation language.

The generalized decision model pertaining to first paragraph of § 112 including the landmark decisions *In re Ruschig Fed Cir and Pfizer Inc. v. Teva Pharmaceuticals Inc* is as shown in Figure 5.7.

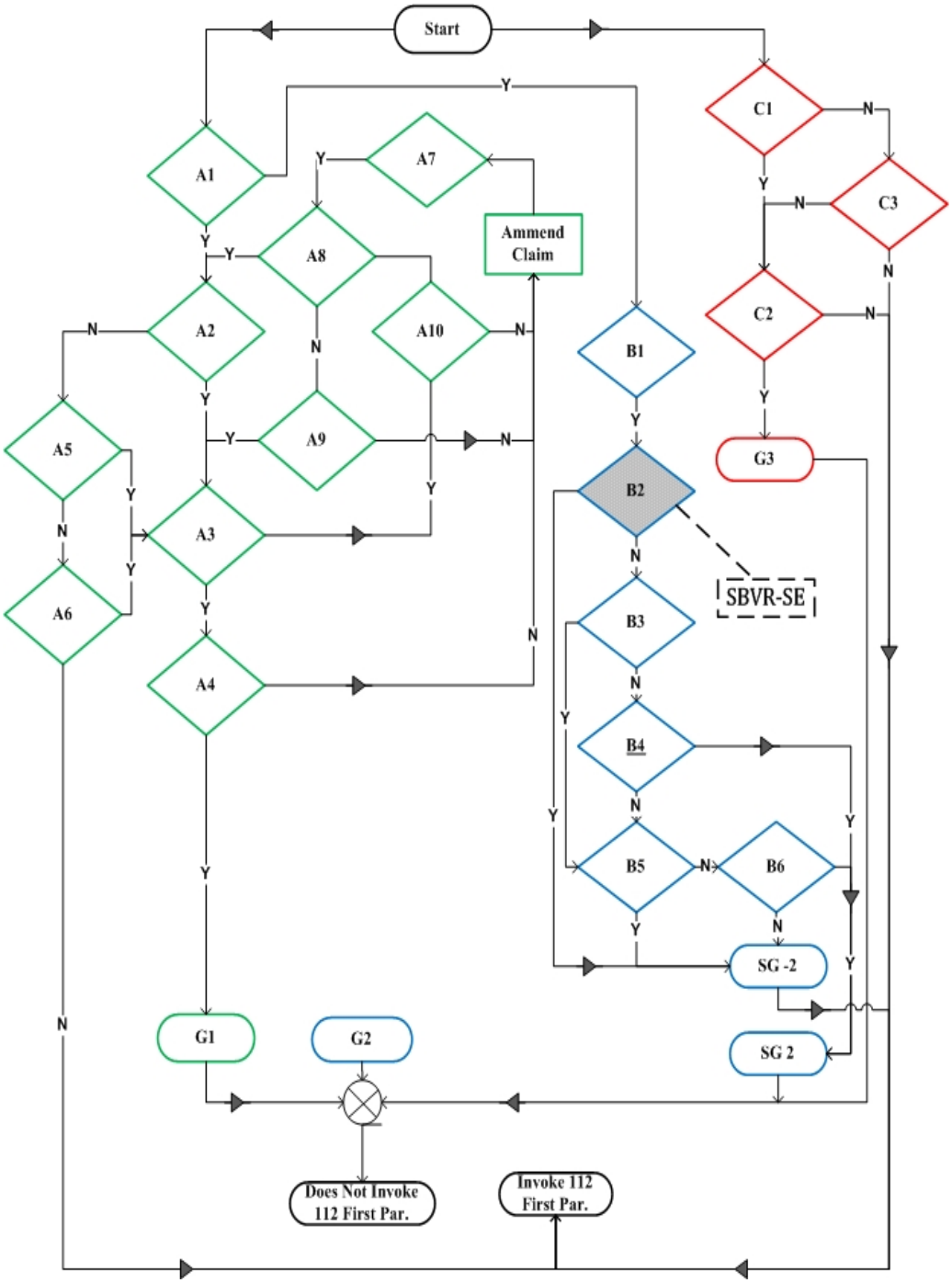


Figure 5.7: Decision model of legal section 112, 1^{st} paragraph [a].

[a]The textual content inside the decision model is left out on purpose to handle the space restrictions, a complete decision model with textual content can be seen under Annex B

5.6 Structured Legal English

The use Structured Legal English (SLE) - a knowledge representation format is coupled with SBVR Structured English (SSE) - as a semi-formal representation format for rep-

resenting legal (procedural) norms was proposed in [113]. The core idea of SLE in the domain of law is: **legal rules** build on legal facts, and **legal facts** build on **legal concepts** which are expressed by **legal terms**. Terms express legal concepts; facts make assertions about these concepts; rules constrain and support these facts. Using the resulting semi-formal legal vocabulary and rules the decision models can be semantically enriched, giving them an underlying formal semantics. Such an adaption of the OMG Semantic Business Vocabulary and Business Rules [76] (OMG SBVR) standard to the legal domain, is as illustrated in Figure 5.8.

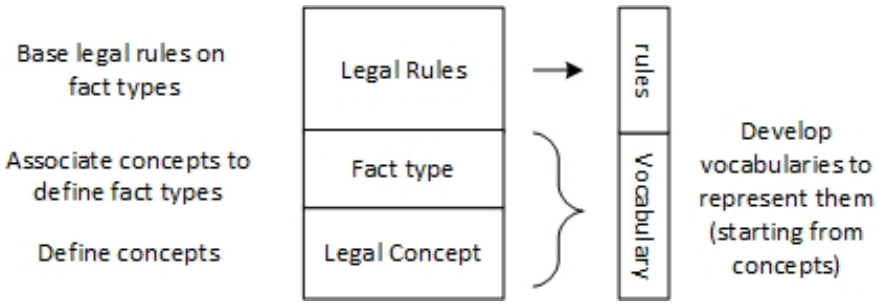


Figure 5.8: Building legal vocabulary(adapted from [113])

Legal (procedural) rules defined in SLE, using predefined **legal vocabularies**, consisting of legal concepts (concepts which have a meaning in the legal tradition, e.g. claim construction vocabulary) in template-based **legal rules**. The mapping of SBVRs standard building blocks into SLE is as shown below:

- **Noun concepts**, which correspond to legal concepts.
- **Verb concepts**, which correspond to relationships between legal concepts.
- **Definitional rules**, which constrain these relationships so that they can be used to define consistent and complete arguments.

Legal concepts represented by noun concepts must be explicitly defined with the intended semantics given in an authoritative source or otherwise acknowledge by implicit pragmatic understanding (the ordinary natural language meaning of the term used). Verb concepts can only use such recognized noun concepts as their terms.

The legal rules can then be constructed using "*if ... then ...*", "*at least*", "*each*" as well as definitional alethic and behavioral deontic legal norm modalities (*"obliged", "permitted"* ...), etc. The following example in the next section illustrates its use.

Along with the definitional alethic and behavioral deontic legal norm modalities, SLE support construction of legal norm with a a wide range of keywords for logical rule formulations. Assuming *'p'* and *'q'* being propositions, the set of keywords broadly divided into *Quantification, Logical and modal Operators* are as shown in Table 5.2. Additional 'modal' mapping schemes are similar to the mapping schemes discussed under Table 4.2 and 4.3 of Chapter 4.

As previously discussed, legal (procedural) rules are defined in Structured Legal English (SLE), using predefined **legal vocabularies**. The aim of a legal vocabulary is to provide a

Table 5.2: Keywords for Logical Forumlations (Adapted from [76])

Quantification			
each	→	$\forall p$	universal quantification
some	→	$\exists p$	existential quantification
at-least n	→	$\exists^{n}, n \geqslant 1$	at-least-n- quantification
at most n	→	$\exists^{0..n}, n \geqslant 1$	at-most-n quantification
at-least n and at-most m	→	$\exists^{n..m}, n \geqslant 1 \& m \geqslant 2$	numeric range quantification
etc...			
Logical Operators			
It is not the case that p	→	$\neg p$	logical negation
p or q	→	$p \vee q$	disjunction
if p then q	→	$p \rightarrow q$	implication
p or q but not both	→	$p \oplus q$	exclusive disjunction
etc...			
Modal Operators			
It is obligatory that p	→	$\boldsymbol{O}p$	obligation formulation
It is necessary that p	→	$\Box p$	necessity formulation
etc...			

glossary like entry for the legal concepts used. A general structure of the legal vocabulary is as shown in Figure 5.9.

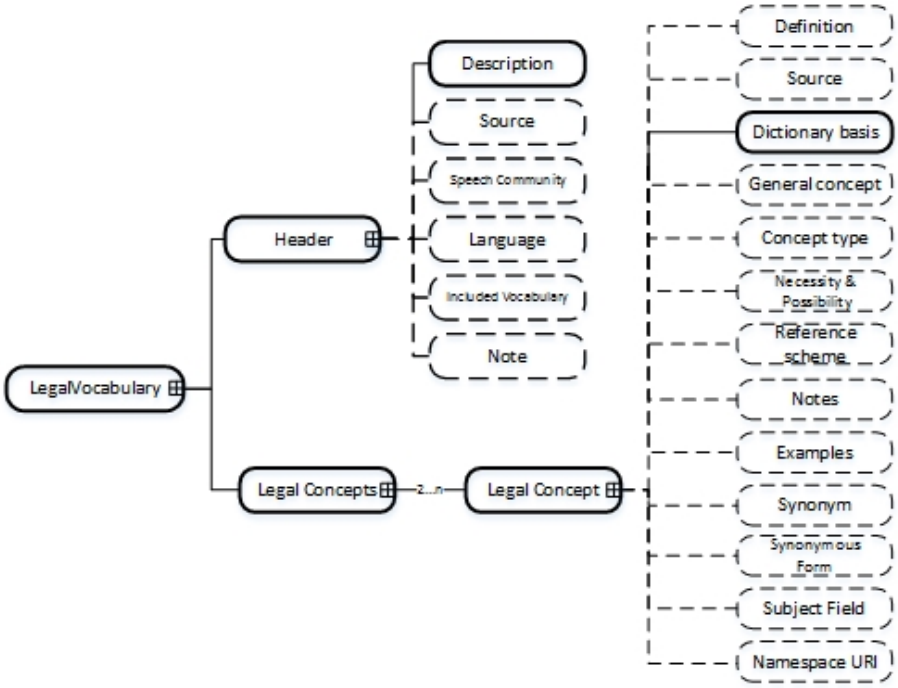


Figure 5.9: Legal Vocabulary Structure

The structure consists of a ***Header*** and ***Legal Concepts***. A ***Header*** is used to define the signature of a vocabulary. It comprises of: *'Description'*, introducing the scope and

the purpose of the vocabulary, *'Source'*, identifier to describe if the current vocabulary is based on a formally-defined work, *'Speech Community'*, identifying the community that controls and is responsible and for the vocabulary, *'Language'*, defining the base-language of the legal vocabulary, *'Included Vocabulary'*, to identify if another vocabulary is fully incorporated into the current legal vocabulary and finally *'note'*, to define user comments on the vocabulary.

A ***Legal Concept*** under a legal vocabulary is defined using several meta-information pertaining to that concept. A list of such meta-information is as enumerated below,

(a) *Definition*: A formal or informal expression that can be logically subsituted for the primary representation

(b) *Source*: The source of the legal concept

(c) *Dictionary Basis*: Definition from a common dictionary that supports the use of the primary representation

(d) *General Concept*: A concept that generalizes the considered legal concept

(e) *Concept Type*: To identify the concept type

(f) *Necessity and Possibility*: To address any modalities of the legal concept

(g) *Reference Scheme*: Denotes how things denoted by the term can be distinguished from each other based on one or more facts about the things

(h) *Note*: User defined comments on the legal concept

(i) *Example*: examples to the legal concept

(j) *Synonym*: defines synonyms to the legal concept

(k) *Synonymous Form*: Used when addressing a verb concept

(l) *Subject Field*: Used when the legal concept is not unique across the domain

(m) *Namespace URI*: To indicate URI of a vocabulary namespace

5.6.1 Structured-Legal-English and Pragmatics

Fig 5.10 depicts a relationship between the framework for interpretations proposed in 3.3 and the structured legal english. The semantic triangle is based on a linguistic triangle, the triangle of reference/meaning, as proposed by Ogden and Richards [114]. The triangle, in its simplest form describes a relationship between the speaker as subject, a concept as object, and its designation.

As described before in Section 3.2.1, the process of applying the law (e.g.inventions/patent applications) as a unilateral, non-verbal conversation between the legislature and members of judicature and executive (e.g.patent officer/examiner), and further- between a citizen and executive (e.g.the inventor and the patent office/examiner).

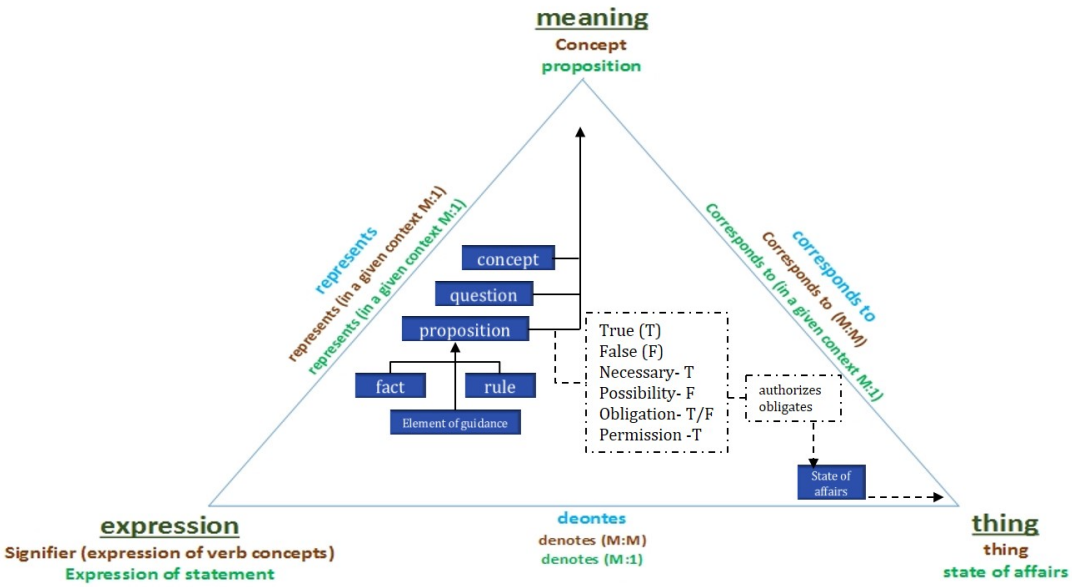


Figure 5.10: Mapping SLE to framework-for-interpretations

The edges of the triangle are expression, meaning and thing. While *expression* and *meaning* are abstract notions, their instances are depicted by *things* in the universe of discourse. Attributes such as *represents, corresponds-to and denotes*, provide the necessary relationship between the edges. A possible (conceptual) mapping between the notions from SLE and pragmatic using the semantic triangle of meaning may be defined as: the notion *meaning* is mapped to a legal concept or a proposition from SLE. The notion *expression* is mapped to expressions defined via verb concepts about statements in SLE. The *thing* is mapped to concrete instances of concepts (i.e. individuals) defined in SLE.

For eg: `expression-of-statement` *represents* a `proposition`, which *corresponds-to* a `state-of-affairs` and `expression-of-statement` *denotes* `state-of-affairs` through `element-of-guidance`.

To illustrate the use of SLE as a semi-formal representation format, A decision point 'B2' is chosen from the decision model (i.e. from Figure 5.7). Decision 'B2' comprises of a procedural rule as stated below:

Essential Subject Matter Missing From Claims

"A claim rejected under 35 U.S.C. 112, first paragraph, as based on a disclosure which is not enabling, critical or essential to the practice of the invention, but not included in the claim(s) is not enabled by the disclosure"

Structured Legal English:

Legal Concepts: Noun concepts defined in green (& underlined) and Individual noun concepts are defined in dark-green(& underlined) starting with capital letters.

Legal Vocabulary:

Table 5.3: Legal Vocabulary

Header	
Description-	35 United States Code Patent Law
Source-	USPTO Glossary
Speech Community-	Patent Practitioners
Language-	English
Included Vocabulary-	Claim-construction vocabulary
Note-	Manual for Patent Examination Procedure
Legal Concept	
Concept-	claim
Definition-	Define the invention and are what aspects are legally enforceable
Source-	USPTO Glossary
Dictionary Basis-	USPTO Glossary
General Concept-	patent_application
Concept Type-	General Concept
Necessity & Possibility-	It is necessary that a claim must be *written_in* English
Reference Scheme-	-NA-
Note-	A patent claim is to read in the light of the specification.
Example-	A self-propelled vehicle, comprising: An engine mounted in said carriage for producing rotational energy. Whereby said carriage can be self-propelled along said surface
Synonym-	-NA-
Synonymous Form-	product_claim, process_claim, method_claim
Subject Field-	Patent Examination
Namespace URI-	http://www.uspto.gov/main/glossary/#c

building on the same lines, we obtain other legal concepts like:
examiner, office_action, paragraph, argument, applicant, patent_application, invention and essential_subject_matter_requirement

Paragraph_7_33_01, **US**

Legal Facts: A legal fact denotes some type of relationship between two or more legal concepts or a characteristic of legal concept, *Verb* concepts are used to define such relationships. *Verb* concepts are defined in blue. In-line to our considered example, below shown are some legal facts constructed using Legal Concepts defined before.

office_action *includes* **Paragraph_7_33_01**
claim *is_rejected_under* essential_subject_matter_requirement

patent_application *is_filed_in* **US**
patent_application *includes* at least 1 claim
examiner *rejects* the claim

Legal (procedural) rules: As discussed before, SLE handles *Definitional/Structural* and *Behavioral* form of rules, both common in the legal domain. SLE uses both the alethic and deontic modalities to express such rules. Legal procedural rules pertaining to ¶ 7.33.01 may be defined as below

(a) It is obligatory that examiner *rejects* the claim and office_action *includes* **Paragraph_7_33_01**, if claim *is_rejected_under* essential_subject_matter_requirement.

(b) It is necessary that a patent_application *includes* at least 1 claim, if patent_application *is_filed_in* **US**

With the procedural legal rules represented in semi-formal representation format, the next step is to transform these semi-formal rules from their computational independent layer to a platform independent layer and thereafter to a platform specific layer. Further sections of this thesis, provided a detailed description of such semantic and logical transformations.

5.7 Formal Representation

This section describes the transformation of legal knowledge from a semi-formal representation to formal representation. As seen in the previous section 5.6, The use of SLE for a semi-formal representation of legal (procedural) rules comprises of two building blocks, the *legal vocabulary* and *legal rules*.

First, this thesis addresses the formal transformation of legal vocabulary and later the formal transformation of the legal rules is addressed.

5.7.1 Legal Vocabulary

The legal vocabulary represented in SLE are transformed into a platform independent formal representation format. The legal concepts and facts from the vocabulary are transformed into an OWL 2 ontology. OWL 2 is the latest version of an ontology language under the World Wide Web Consortium (W3C) for the development of semantic web.

Apart from the fact that such a formal transformation helps in building a knowledge base, using which legal reasoners reason it also helps to check the consistency of legal information itself (using legal ontological reasoners).

The mapping scheme is defined on three knowledge representation layers of a legal vocabulary. An abstract level mapping between the legal vocabulary structure and OWL2 structure is as shown in Table 5.4. Legal concepts are mapped to *OWL-Entities*, logical (legal) operators are mapped to *OWL-Expressions* and finally the legal facts are mapped to *OWL-Axioms*.

Table 5.4: Abstract mapping between SLE Vocabulary structure to OWL 2 structure

Legal concepts	↔	OWL- Entities
Legal (logical) operators	↔	OWL - Expressions
Legal facts	↔	OWL - Axioms

Several mapping schemes have been proposed for transforming an SBVR vocabulary transformation into an OWL 2 ontology [115] [116] [117]. This thesis adapts (and re-iterates some) the mappings as proposed by 'Kendall' and 'Karpovic' to the context of SLE vocabulary transformation. The core-mappings are as follows:

(a) Legal *Noun, Verb* and *Individual* concepts may be expressed as a *'class'*, *'Object-Property'* and as *'NamedIndividual'* respectively under OWL2 notation.

Noun Concept	↔	Declaration(Class(owl:Thing))
Verb_Concept	↔	Declaration(ObjectProperty(owl:ObjectProperty))
Individual Concept	↔	Declaration(NamedIndividual(owl:Individual)

(b) Existential and Universal quantifications may be expressed as a *Object Property Expression (OPE)* and a *Class Expression (CE)*, and it contains all those individuals that are connected by OPE and by only *OPE* respectively, to an individual that is an instance of CE under OWL2 notations.

some	↔	ObjectSomeValuesFrom:='ObjectSomeValuesFrom''('*OPE CE*')'
each	↔	ObjectAllValuesFrom:='ObjectAllValuesFrom''('*OPE CE*')'

(c) The maximum and minimum cardinality of the existential quantifications i.e. the at-least n and at-most n may be expressed as a non-negative integer n, an *OPE*, and a *CE*, and it contains all those individuals that are connected by *OPE* to at least n and to at-most n different individuals that are instances of *CE* respectively under OWL2 notation.

at-least n	↔	ObjectMinCardinality:='ObjectMinCardinality"('nonNegativeInteger *OPE*[*CE*]')'
at-most n	↔	ObjectMaxCardinality:='ObjectMaxCardinality"('nonNegativeInteger *OPE*[*CE*]')'

(d) Negation may be expressed as a *ObjectComplementOf* under OWL 2 notation.

It is not the case that	↔	ObjectComplementOf:='ObjectComplementOf"('*CE*')'

(e) A *'Unary'* and a *'Binary'* legal fact may be expressed under OWL 2 notations as shown below:

Unary Fact	↔	Declaration(DataProperty(owl:DataProperty)) DataPropertyDomain := 'DataPropertyDomain' '('axiomAnnotations DataPropertyExpression *CE*')' DataPropertyRange :='DataPropertyRange' '(' axiomAnnotations DataPropertyExpression DataRange')'.
Binary Fact	↔	Declaration(ObjectProperty(owl:ObjectProperty)) ObjectPropertyDomain := 'ObjectPropertyDomain' '('axiomAnnotations ObjectPropertyExpression *CE*')' ObjectPropertyRange := 'ObjectPropertyRange' '('axiomAnnotations ObjectPropertyExpression *CE*')'.

The entries as shown in Table 5.3, that define a legal concept are also mapped onto OWL 2. One possible mapping scheme is as shown below in Table 5.5[3].

Table 5.5: SLE vocabulary to formal OWL-2 mapping schema

Header		
Description	↔	skos:definition
Source	↔	awol:src
Speech Community	↔	foaf:group
Language	↔	owl: languageTags
Included Vocabulary	↔	owl: imports
Note	↔	rdfs: comment
Legal Concept		
Concept	↔	owl: Class owl: SubClassOf
Definition	↔	skos:definition
Source	↔	awol:src
Dictionary Basis	↔	—
General Concept	↔	owl: Class
Concept Type	↔	skos:ConceptScheme
Necessity & Possibility	↔	—
Reference Scheme	↔	owl: HasKey
Note	↔	rdfs: comment
Example	↔	skos:example
Synonym	↔	owl: sameAs owl: EquivalentClass
Synonymous Form	↔	owl: equivalentProperty
Subject Field	↔	—
See	↔	rdfs: seeAlso
Namespace URI	↔	xsd: anyURI

[3]Formal OWL 2 mapping notations are dropped for simplicity

5.7.2 Legal Rules

The second part involves the transformation of legal norms in their elementary form as represented in semi-formal SLE into a formal rule representation language. Although a direct translation into a platform-specific language of a logic reasoner is possible, for the purpose of machine processing, reusability, interchange and structured Web publication, we propose to use KR4IPLaw- Schema (KR4IPLaw-s), an ad-hoc patent norm representation format, which seamlessly integrates into the existing rule representation standards like RuleML [17], ReactionRuleML [118] and LegalRuleML [119].

The core idea behind proposing this schema is to depict the modularization approach, as proposed in thought this thesis. The structure of such a schema is as shown in Figure 5.11 [4].

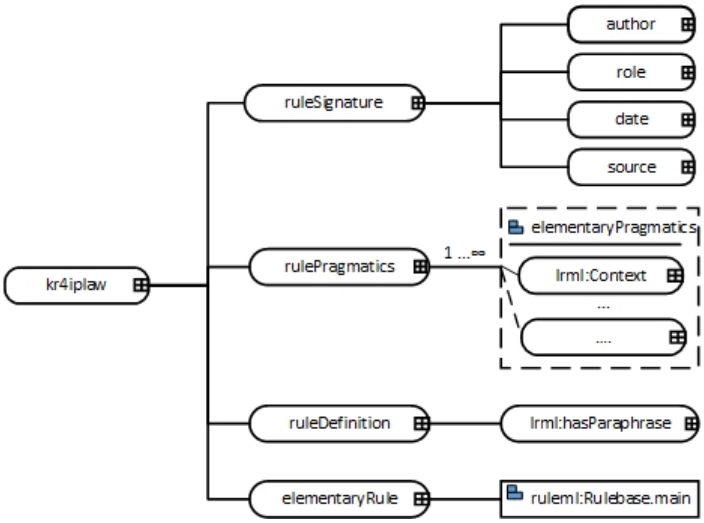


Figure 5.11: KR4IPLaw Schema Structure

The KR4IPLaw-s representation structure comprises of four fundamental blocks;

(a) ruleSignature

(b) ruleDefinition

(c) rulePragmatics

  (i) elementaryPragmatics

(d) elementaryRule

The <**ruleSignature**>, defines the *'Rule Signature'*. The <**ruleDefinition**>, provides the description (semi-formal) of a considered legal norm. The <**rulePragmatics**> defines the pragmatic information pertaining to the considered legal norm and the last block the <**elementaryRule**> describes the actual rule using the available legal operators.

[4] A detailed structure of schema can be found under Annex C

Each block defined in Figure 5.11, may be viewed as collection of modules. The *'ruleSignature'* block comprises four modules, which are further hierarchically subdivided into sub-modules to define different parameters. The four main modules define information pertaining to; person responsible for the creation of an elementary norm (<author>), the function/status played by that person/agent (<role>), the time of creation/modification (<date>) and the source of the semi-formal rule (<source>). The <**rulePragmatics**> block consisting of a collection of elementary pragmatics modules, wherein, each *'elementaryPragmatics'* module provide a part of the required *'context'* information for a rule under consideration. As defined before, the <**ruleDefinition**> and <**elementaryRule**> block defines and formally represents the actual rule (i.e. *'ruleContent'*).

KR4IPLaw-s integrates OASIS LegalRuleML, an XML-based expression language on the PIM layer for modeling legal rules. LegalRuleML, supports the modeling of both constitutive (legal concept definitions) and prescriptive (behavioural) rules. Among other expressiveness, it supports the representation of legal meta-data, penalty, reparation and specific deontic operators. Like LegalDocML, LegalRuleML also provides the capability to manage multiple semantic annotations through annotation blocks as internal or external metadata. By that it manages the temporal issues such as, provisions, provenance references, application of rules and their histories in a unambiguous manner. Furthermore, with its flexibility to separate the logical layer and the context layer.

The *'metainfo'* part of LegalRuleML integrated into the KR4IPLaw-s is as depicted in Figure 5.12.

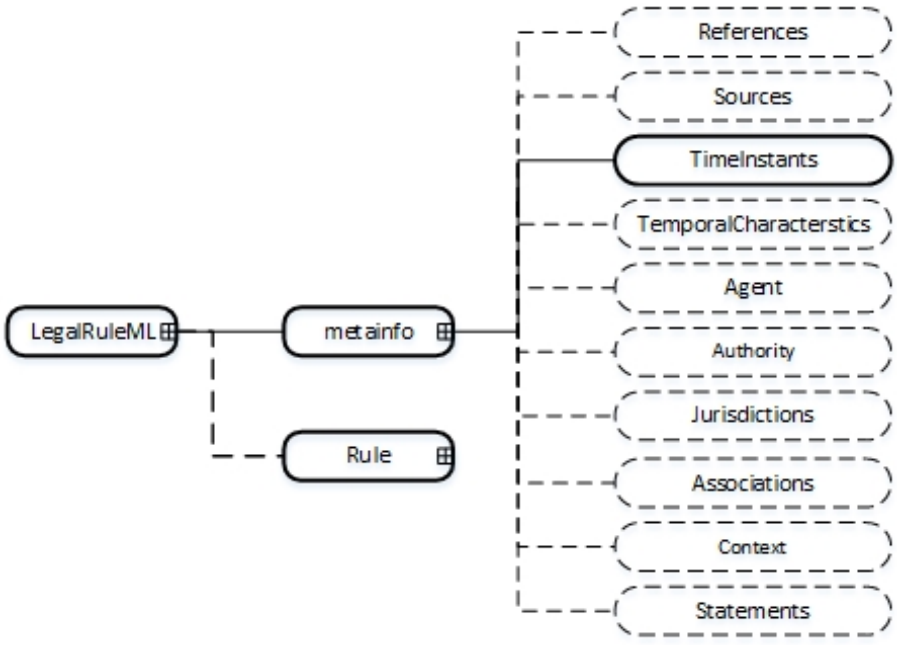


Figure 5.12: legal rule 'metainfo' structure

The <**LegalSource**> module is used to identify the collection of relevant legal resources. The <**References**> module holds the information pertaining to a collec-

tion of references relevant to the rule document. LegalRuleML within the KR4IPLaw-s provides an N:M relationship between textual provisions and the rules itself whereby it addresses the issue of *isomorphism* between formal rule and the legally binding textual statements pertaining to the rule. The <**Agents**> and the <**Authority**> module helps to capture the provenance and the authoritativeness of the rules. The ammendments of the legal text over the time and the events depicting temporal situations such as *'enforceablity, efficacy and applicability* are captured through the <**TimeInstants**> and the <**TemporalCharacteristics**> modules. The next important module in the metainfo block is the <**Context**> module, It captures information pertaining to the jurisdiction, the strength the author of the legal rule under consideration.

To accommodate behavioral aspects within a legal rule, KR4IPLaw-s through LegalRuleML provides with a set of legal operators as shown in Figure 5.13. The detailed descriptions such operators are as shown described below:

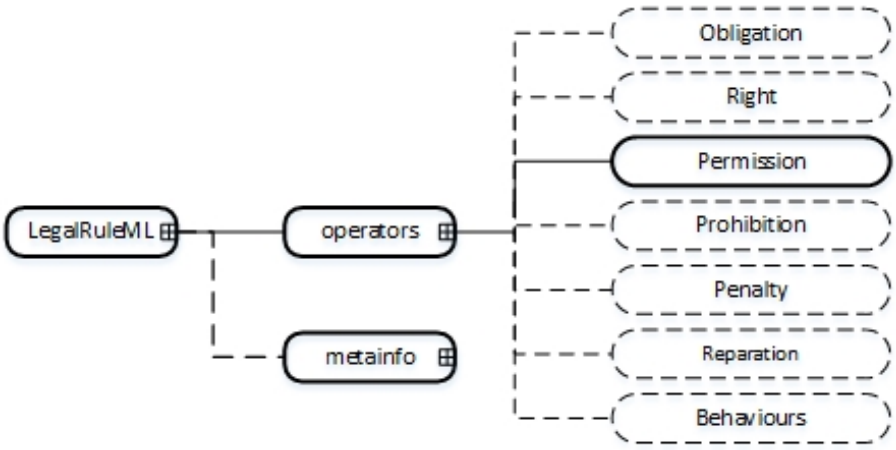


Figure 5.13: Legal (rule) Operators

(a) **Prohibition**: a deontic Specification for a state, an act, or a course of action to which a Bearer is legally bound, and which, if it is achieved or performed, results in a Violation.

(b) **Obligation**: a deontic Specification for a state, an act, or a course of action to which a Bearer is legally bound, and which, if it is not achieved or performed, results in a Violation.

(c) **Permission**: a deontic Specification for a state, an act, or a course of action where the Bearer has no Obligation or Prohibition to the contrary. A weak Permission is the absence of the Obligation or Prohibition to the contrary; a strong Permission is an exception or derogation of the Obligation or Prohibition to the contrary.

(d) **Right**: a deontic Specification that gives a Permission to a party (the Bearer) and implies there are Obligations or Prohibitions on other parties (the Auxiliary Party) such that the Bearer can (eventually) exercise the Right.

(e) **Compliance**: an indication that an Obligation has been fulfilled or a Prohibition has not been violated.

(f) **Violation**: an indication that an Obligation or Prohibition has been violated.

(g) **Penalty**: a Legal Statement of a sanction (e.g. a punishment or a correction).

(h) **Reparation**: an indication that a PenaltyStatement is linked with a PrescriptiveStatement. It indicates that a sanction may apply where the PrescriptiveStatement entails a deontic Specification and when there is a Violation of the deontic Specification.

(i) **Behaviors**: A situation in which a obligation has been fulfilled or a prohibition has not been violated.

(j) **Facts**: Factual statements are modeled using the RuleML elements.

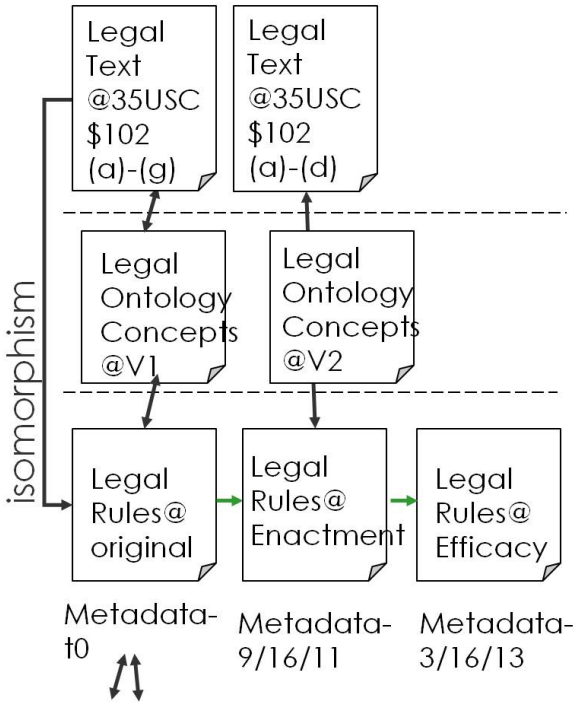


Figure 5.14: Temporal management of legal rules (adapted from [120])

As discussed earlier, KR4IPLaw-s supports the temporal management of legal rules, a necessity for lifecycle management and temporal reasoning. An abstract working procedure with temporal management is as shown in Figure 5.14 and is explained below:

(a) Detect the rules and the ontology classes affected by the changes

(b) Refer to the proper version of the text and of the ontology classes

(c) Take in consideration the evolution of the rules over time with also their metadata fixed in a given time t_x

(d) Sources, Rules and context metadata are valid in a given temporal interval.

Further-on, the KR4IPLaw-s exploits the seamless integration of RuleML (esp. ReactionRuleML) to extend the existing mapping schema to accommodate other deontic and alethic logic formalisms. Modal operators are encapsulated as *'Meta-Knowledge'* in ReactionRuleML and a *'meta-knowledge'* is represented using a meta-knowledge formula (as in our case, the modal formula) $'@\phi'$ wherein, ϕ denotes the modal operator.

Figure 5.15 depicts an abstract structure of the meta-knowledge module embedded withing the ReactionRuleML as required to formally represent its legal norms (in modal form).

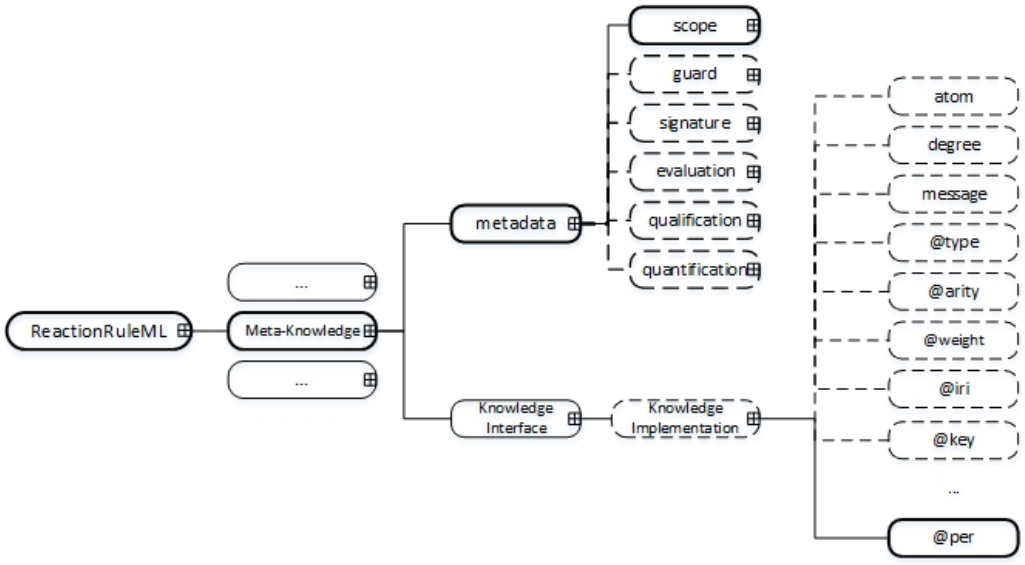


Figure 5.15: ReactionRuleML's 'Meta-Knowledge' module for representing legal norms

The listing below shows the syntax constructed to represent the legal (modal) norms which further can be integrated inside the considered KR4IPLaw-s for formally modeling legal norms.

- Alethic: It is possiblie that p

```
<!-- It is possible that -->
  <Operator type="ruleml:AlethicOperator" iri="ruleml:Possible">
      <Atom>
        <Rel per="modal">p</Rel>
      </Atom>
  </Operator>
```

- Alethic: It is Not-necessary that p

```
<!-- It is not necessary that -->
  <Neg>
      <Operator type="ruleml:AlethicOperator" iri="ruleml:Necessary">
           <Atom>
            <Rel per="modal">p</Rel>
           </Atom>
      </Operator>
  </Neg>
```

- Deontic: It is Obliged that *p(X)*

```
<!-- It is obliged that p(X)-->
    <Operator type="ruleml:DeonticOperator" iri="ruleml:Obliged">
    <Atom>
      <Rel per="modal">p</Rel>
      <Var>X</Var>
    </Atom>
  </Operator>
```

- Deontic: It is permitted that *q(Y)*

```
<!-- It is permitted that q(Y)-->
    <Operator type="ruleml:DeonticOperator" iri="ruleml:Permitted">
      <Atom>
        <Rel per="modal">q</Rel>
        <Var>Y</Var>
      </Atom>
  </Operator>
```

Modal rules can be constructed using the modal operators defined before. As an example, one such modal rule is as shown in the listing below

- If something is *'obliged'* then it is also *'permitted'* (*Conditional permission*).

```
<!-- modal rule: "if obliged(p(X)) then permitted(p(X))" -->
<Rule>
   <if>
      <Operator type="ruleml:DeonticOperator" iri="ruleml:Obliged">
         <Atom>
            <Rel per="modal">p</Rel>
            <Var>X</Var>
         </Atom>
      </Operator>
   </if>
   <then>
      <Operator type="ruleml:DeonticOperator" iri="ruleml:Permitted">
         <Atom>
            <Rel per="modal">p</Rel>
            <Var>X</Var>
         </Atom>
      </Operator>
   </then>
</Rule>
```

- Furthermore, ReactionRuleML provides the capability to accommodate the temporal aspects embedded within a norm (i.e. legal event calculus). One such reified modal rule addressing the *Event, Condition, Action*(*ECA*) can be: If an event $p(X)$ happens at time T and holds the conditions that $p(X)$ is *Forbidden* at the same time T, then initiate the $p(X)$ and *obliged* $q(X)$.

```
<!-- if happens(p(X),T) and holds(forbidden(p(X)),T) then initiates(p(X),
     obliged(q(X)),T)-->
<Rule>
   <if>
      <Operator type="ruleml:Connective" iri="ruleml:And">
         <Atom>
            <Rel per="value" iri="ruleml:Happens"/>
            <Expr><Fun per="value">p</Fun><Var>X</Var></Expr>
            <Time><Var>T</Var></Time>
         </Atom>
         <Atom>
            <Rel per="value" iri="ruleml:Holds"/>
            <Expr>
```

```
13              <Fun per="value" iri="ruleml:Forbidden"/>
14              <Expr><Fun per="modal">p</Fun><Var>X</Var></Expr>
15           </Expr>
16           <Time><Var mode="+">T</Var></Time>
17        </Atom>
18     </Operator>
19  </if>
20  <then>
21     <Atom>
22        <Rel per="value" iri="ruleml:Initiates"/>
23        <Expr><Fun per="value">p</Fun><Var>X</Var></Expr>
24        <Expr>
25           <Fun per="value" iri="ruleml:Obliged"/>
26           <Expr><Fun per="modal">q</Fun><Var>X</Var></Expr>
27        </Expr>
28        <Time><Var mode="+">T</Var></Time>
29     </Atom>
30  </then>
31 </Rule>
```

- Deontic Paradox: Contrary-to-duty Obligation

```
1  <!-- modal rule if obliged (neg(p(X))) and p(X) then obliged(q(X)) -->
2     <Rule>
3       <if>
4          <And>
5            <Operator type="ruleml:DeonticOperator" iri="ruleml:Obliged">
6               <Neg>
7                  <Atom>
8                    <Rel per="modal">p</Rel>
9                    <Var>X</Var>
10                  </Atom>
11              </Neg>
12           </Operator>
13                 <Atom>
14                    <Rel per="value">p</Rel>
15                    <Var>X</Var>
16                 </Atom>
17         </And>
18      </if>
19      <then>
20         <Operator type="ruleml:DeonticOperator" iri="ruleml:Obliged">
21               <Atom>
22                  <Rel per="modal">q</Rel>
23                  <Var>X</Var>
24                </Atom>
25         </Operator>
26      </then>
27    </Rule>
```

Continuing with our example 5.3, the representation is split into two parts. Firstly, defining the context information in terms of its meta-data and secondly, describing the rules content, i.e., the inner legal norms on the logical layer in terms of reaction rules [121].

Listing 5.4, depicts the annotation of the context information pertaining to the semi-formal rules represented in terms if SLE [5]. Inline to the structure shown in Figure 5.12, the <LegalSource> used for describing the source of a legal norm under consideration is instantiated to MPEP, USPTO, the authority concerned with

[5]Only important aspects of the annotation is highlighted here. Complete annotation of the legal section can be found under Annex A

making any changes to the MPEP is associated to the meta-data <Authority>. <TimeInstants> and <TemporalCharacterstic> captures the changes to a legal norm and its temporal aspects respectively. The qualification of Rule 1 overriding Rule 2 is captured using the meta-data <Qualification>. The <Jurisdiction> meta-data defining where such legal norm is applicable is instantiated to US.

Listing 5.4: Formal representation of the context information withing a semi-formal legal rule

```
<!-- Defines the source of the legal text-->
  <lrml:LegalSources>
      <lrml:LegalSource key="ref1" sameAS="http://mpep.patentbargroup.com/html/0700
          _706_03_c.htm"/>
  </lrml:LegalSources>
  <lrml:References>
      <lrml:Reference refersTo="ref2" refID="us/codes;us/patentlaw/main#title35/"
            refIDSystemName="AkomaNtoso3.0"/>
  </lrml:References>
  <lrml:Authorities>
      <lrml:Authority key="USPTO" sameAS="/ontology/organizations/PatentOffice/
          USPTO"/>
      <lrml:type iri="&lrmlv;PatentOffice"/>
  </lrml:Authorities>
  <lrml:Agents>
     <lrml:Agent key="examiner"
            sameAS="/ontology/organizations/PatentOffice/USPTO/Person#Examiner"/>
  </lrml:Agents>
  <lrml:TimeInstants>
      <ruleml:Time Key="#t7"><ruleml:Data Type="xs:Date">2011-09-16</ruleml:Data></
          ruleml:Time>
      <ruleml:Time key="#t8"><ruleml:Data Type="xs:Date">2012-09-16</ruleml:Data></
          ruleml:Time>
  </lrml:TimeInstants>
<!-- Temporal characterstics of the rule -->
  <lrml:TemporalCharacterstic key="tblock1">
      <lrml:forRuleStatus iri="&lrmlv;#Efficacious"/>
      <lrml:hasStatusDevelopment iri="&lrmlv;#End"/>
      <lrml:atTimeInstant keyref="#t8"/>
  </lrml:TemporalCharacterstic>
<!-- Rule Qualification/Override Principle -->
  <lrml:hasQualification><lrml:Overrides over="#ps1" under="#ps2"/></
      lrml:hasQualification>
<!-- Jurisdictions of the rule -->
  <lrml:hasJurisdcition><lrml:Jurisdictions key="#jurisdiction1"/></
      lrml:hasJurisdcition>
<!-- Context Information-->
  <lrml:hasContext>
      <lrml:appliesSelection keyref="#r1"/>
      <lrml:toStatement keyref="#ps1"/>
      <lrml:appliesjurisdiction keyref="#jurisdiction1" iri="&jurisdictions;us"/>
  </lrml:hasContext>
```

Figure 5.17b, shows the representation of the logical part of the norm in Legal-RuleML and Reaction RuleML format. Wherein, the SLE constraints are translated into prescriptive constraint statements. Figure 5.17b(a) and 5.17b(b) exemplifiy the representation of deontic obligations and alethic necessity. The IRI's obtained from the OWL2 ontologies of the SLE vocabulary are used to type the logical parts of the norm represented using RuleML's typing approach (@type attribute) [122].

While we illustrated the transformation and syntactic representation of SLE in KR4IPLaw-s via LegalRuleML and Reaction RuleML in this section, we define a semantic-preserving translation from SLE to KR4IPLaw-s in the next section.

```
<lrml:PrescriptiveStatement key="#ps1">
  <lrml:hasTemplate>
    <rrml:Rule key="#r1">
      <lrml:hasParaphrase> It is obligatory that examiner rejects
            the claim and office_action includes
            Paragarph_7_33_01 if claim is_rejected_under
            essential_subject_matter_requirement
      </lrml:hasParaphrase>
      <rrml:if>
        <rrml:Atom key="sbvrfact1">
<rrml:Rel iri="&voc;is_rejected_under"/>
<rrml: Var type="&voc;claim"/>
<rrml: Var type="&voc;essential_subject_matter_
                    -requirement"/>
        </rrml:Atom>
      </rrml:if>
      <rrml:then>
        <lrml:Obligation>
          <rrml:And>
            <rrml:Atom key="sbvrfact2">
<rrml:Rel per="modal" type="&voc;rejects"/>
<rrml:Var type="&voc;examiner"/>
<rrml:Var type="&voc;claim"/>
            </rrml:Atom>
            <rrml:Atom key="sbvrfact3">
<rrml:Rel per="modal" type="&voc;includes" />
<rrml:Var type="&voc;office_action"/>
<rrml:Ind type="&voc;Paragarph_7_33_01"/>
            </rrml:Atom>
          </rrml:And>
        </lrml:Obligation>
      </rrml:then>
    </rrml:Rule>
  </lrml:hasTemplate>
</lrml:PrescriptiveStatement>
```

(a)

```
<lrml:ConstitutiveStatement key="#ps2">
    <lrml:hasTemplate>
      <rrml:Rule key="#r2">
        <lrml:hasParaphrase>It is necessary that a
          patent_application includes at least 1 claim
          if patent_application is_filed_in US
        </lrml:hasParaphrase>
        <rrml:if>
          <rrml:Atom key="sbvrfact4">
  <rrml:Rel type="&voc;is_filed_in"/>
  <rrml:Var type="&voc:patent_application"/>
  <rrml: Ind iri="&voc:US"/>
          </rrml:Atom>
        </rrml:if>
<rrml:then>
<rrml:Operator type="rrml:AlethicOperator"
iri="rrml:Necessary">
  <rrml:Atom key="sbvrfact5>
    <rrml:quantification>
      <rrml:Exists>
        <rrml:Var type="&voc;claim">claim</rrml:Var>
      </rrml:Exists>
    </rrml:quantification>
    <rrml:Rel per="modal" iri="&voc;includes"/>
    <rrml:Var type="&voc;patent_application"/>
    <rrml:Var type="&voc;claim">claim</rrml:Var>
  </rrml:Atom>
</rrml:Operator>
</rrml:then>
</rrml:Operator>
      </rrml:Rule>
    </lrml:hasTemplate>
  </lrml:ConstitutiveStatement>
```

(b)

Figure 5.16: Legal norms on PIM layer.

(a) Representation of deontic behavior in a legal norm

(b) Representation of alethic behavior in a legal norm

5.8 Semantic Transformation from SLE to KR4IPLaw

To map from Structured Legal English (SLE) into KR4IPLaw (which includes LegalRuleML and Reaction RuleML), we define a semantics preserving translation of both into a First Order Deontic Alethic Logic (FODAL) [102]. As defined before in 4.4.6.4, FODAL is a multi-modal first order extension combining quantified Standard Deontic Logic (SDL) and the quantified Alethic Modal Logic (AML) $S4$. It is used as the underlying formal semantics for both of our patent law representation languages, i.e. SLE and KR4IPLaw-s. This semantics also forms the basis for further transformations from restricted subsets of FODAL into logics for practical reason-

ing, namely description logics (the DL *ALCQI*) and logic programming (quantified extended horn logic). This enables us to map the legal vocabularies, represented in SLE into OWL2 ontologies, and the legal rules in SLE respectively to KR4IPLaw rules into Prova rule bases, on which efficient reasoning can be done.

Based on the FODAL language, its semantics and combined axiomatic system as defined in 4.4.6.4, we propose a transformation function, τ for the mapping between SLE and KR4IPLaw-s via the FODAL semantics. The complete definition can be found in the corresponding FODAL Semantic Profile [6], which has been defined for Reaction RuleML.

```
1       <evaluation>
2             <Profile type="FirstOrderDeonticAlethicLogicProfile"/>
3       </evaluation>
```

Using Reaction RuleML's Semantic Profile mechanism [123], the FODAL semantics can be specified as intended semantics for the interpretation of Legal Reaction RuleML representations and for further transformations.

5.8.1 Translation from SLE to FODAL

For the translation from SLE into FODAL, the following translation $\tau_{SLE}(\cdot)$ from elements of SLE to closed formulas in first-order deontic-alethic logic is used:

(a) For each noun concept N from SLE, $\tau_{SLE}(N)$ is an unary predicate in FODAL.

(b) For each n-ary verb concept V from SLE, $\tau_{SLE}(V)$ is a n-ary predicate in FODAL.

(c) For each rule R from SLE, $\tau_{SLE}(R)$ is defined inductively as follows:

Definition (c) 1: $\tau_{SLE}(\hat{R}) = \phi_{\hat{R}}$, *where* $\hat{R}$ *is a non-modal SLE expression and* $\phi_{\hat{R}}$ *is its non-modal first-order logic translation,*

Definition (c) 2: $\tau_{SLE}(\neg R) = \neg\tau_{SLE}(R)$,

Definition (c) 3: $\tau_{SLE}(R_1 \circ R_2) = \tau_{SLE}(R_1) \circ \tau_{SLE}(R_2)$, *where* R_1 *and* R_2 *are SLE rules and* $\circ \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$.

Definition (c) 4: $\tau_{SLE}(\Box\hat{R}) = \Box\tau_{SLE}(\hat{R})$ *and* $\tau_{SLE}(\boldsymbol{O}\hat{R}) = \boldsymbol{O}\tau_{SLE}(\hat{R})$

5.8.2 Translation from FODAL to KR4IPLaw

For the translation into KR4IPLaw we normalize the FODAL sentences to atomic modal sentences using the typical axioms of normal modal logic, as defined in the previous subsection. Furthermore, we restrict the expressiveness to only mono-modal closed formula, i.e., the original SLE rules are allowed to contain only one modality[7]. We define the translation as the inverse translation function $\tau_{KR4IPLaw}(\cdot)^{-1}$ from normalized mono-modal FODAL formulas to KR4IPLaw as follows:

[6] FODAL Semantic Profile: http://goo.gl/pN58FF

[7] Since our patent law constraints in SLE only have one modality as well as KR4IPLaw statements are mono-modal, mono-modal FODAL provides enough expressiveness

(a) For each constant c (i.e. an individual object/thing in SLE), $\tau^{-1}_{KR4IPLaw}(c)$ maps it

i. To a data term $< Data >$ in $LRML$ if the constant has an interpretation as a data type in the the XML Schema data types.

ii. To an individual term $< Ind >$ in $LRML$ otherwise.

(b) For each variable v, $\tau^{-1}_{KR4IPLaw}(v)$ maps it to a variable $< Var >$ in $LRML$.

(c) For each unary predicate p in $FODAL$, $\tau^{-1}_{KR4IPLaw}(p)$ maps its only argument term (a constant or a variable) into a term in $LRML$ and assigns the predicate relation p_r as type attribute to the $LRML$ term $@type = "p_r"$ [8].

(d) For each n-ary predicate p in $FODAL$, $\tau^{-1}_{KR4IPLaw}(p)$ maps it into an n-ary atom $< Atom >$ in $LRML$ using the predicate relation as relation $< Rel >$ [9] for the atom and each argument term in the $FODAL$ predicate p is mapped into a typed term in the $LRML$ atom, where the type is coming from the previous mapping of a unary predicate which gives the type of the term.

(e) for each formula R in $FODAL$, $\tau^{-1}_{KR4IPLaw}(R)$ is defined inductively as follows:

Definition (e) 1: *$\tau^{-1}_{KR4IPLaw}(\hat{R})$ maps into a corresponding LRML formula $< formula >$, where $\hat{R}$ is a non-modal first-order logic formula and the LRML formula is its non-modal KR4IPLaw translation.*
In particular:

- *if $\hat{R}$ is a conjunction then it is mapped into $< And >$.*
- *if $\hat{R}$ is a disjunction then it is mapped into $< Or >$.*
- *if $\hat{R}$ is an implication (or a formula which logically corresponds to an implication) then it is mapped into $< Rule >$.*
- *if $\hat{R}$ is a universal quantifier or existential quantifier then it is mapped into a quantifier $< Forall >$ (might be left implicit if no further constraints are defined on the quantifier) or $< Exists >$, with the declared variables being typed @type with their type (see unary predicate mapping) and additional quantifier constraints defined in the LRML quantifier ("such that" $< formula >$ and guard constraints $< guard >$).*

Definition (e) 2: *$\tau^{-1}_{KR4IPLaw}(\neg R)$ maps into a LRML negation $< Neg >$ with $\tau^{-1}_{KR4IPLaw}(R)$ being the corresponding LRML formula which is negated.*

Definition (e) 3: *$\tau^{-1}_{KR4IPLaw}(R_1 \circ R_2)$, where R_1 and R_2 are FODAL formulas and $\circ \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ maps into $\tau^{-1}_{KR4IPLaw}(R_1)$ $\tau^{-1}_{KR4IPLaw}(\circ)$ $\tau^{-1}_{KR4IPLaw}(R_2)$, with $\tau^{-1}_{KR4IPLaw}(\circ) = \{\wedge =< And >, \vee =< Or >, \rightarrow < Rule >, \leftrightarrow = < Equivalent >\}$.*

Definition (e) 4: *$\tau^{-1}_{KR4IPLaw}(\Box\hat{R})$ maps into a (definition) constitutive statement $< ConstitutiveStatement >$ in LRML with the alethic necessary operator $\Box$ mapped*

[8] We use webized types coming from the SLE Web vocabulary respectively translated OWL2 ontology, i.e., we use the IRIs of the vocabulary/ontology as sort symbol names

[9] We use the IRI from the SLE vocabulary or its translated OWL2 ontology

into $< rrml : Operatortype = "rrml : AlethicOperator" iri = "rrml : Necessary" >$ *and* $\tau^{-1}_{KR4IPLaw}(\hat{R})$ *mapped into its corresponding LRML formula and* $\tau^{-1}_{KR4IPLaw}(\boldsymbol{O}\hat{R})$ *maps into a (behavioural) prescriptive statement* $< Prescriptive - Statement >$ *with the deontic obligation operator* $\boldsymbol{O}$ *mapped into the obligation* $< Obligation >$ *in LRML and* $\tau^{-1}_{KR4IPLaw}(\hat{R})$ *mapped into its corresponding LRML formula.* [10]

For the other alethic and deontic operators, $\tau^{-1}_{KR4IPLaw}$ gives a similar mapping. Continuing with the example 5.3, the semi-formal legal rule in SLE as shown in 5.17b is transformed into a FODAL sentence as shown below:

[10]The transformation might additionally denormalize the formulas and map into a *LRML* templates *hasTemplate* using the original SLE expressions

Figure 5.18: KR4IPLaw rule representation instance

An SSE expression 'R' is transformed into an FODAL sentence 'F' and further into an '$LRML$' formula by $\tau_{SLE}(R)$.

R= *It is necessary that a patent_application includes at least 1 claim*
F= $\forall X \exists Y (Patent\_Application(X) \wedge Claim(Y) \wedge includes(X, Y))$

This corresponds to the alethic necessity conclusion of the *"ConstitutiveStatement"* rule shown in Figure 5.17b(b). Similarly,

$$\text{R= } \textit{It is obligatory that examiner rejects the claim and office\_action includes Paragraph\_7\_33\_01}$$
$$\text{F= } \boldsymbol{O}(\forall X, Y, Z(Examiner(X) \wedge Claim(Y) \wedge rejects(X,Y) \wedge Office\_Action(Z) \wedge includes(Z, "Paragraph\_7\_33\_01")))$$

which corresponds to the deontic obligation conclusion of the *"PrescriptiveStatement"* rule shown in Figure 5.17b(a).

A re-representation of the deontic legal norm discussed before in Section 5.7 is as shown in Fig 5.18.

5.9 Platform Specific Representation

Further-on, the rules in its elementary form (elementary rules) are transformed from their Platform Independent Model (PIM) layer into a Platform Specific Model (PSM) layer. We make use of the Prova [124] [125] [126] [127], a Semantic Web rule language and a high expressive distributed rule language for the purpose of representing elementary rules for reasoning. It combines benefits of declarative and object oriented programming by combining syntaxes of Prolog and Java. The syntax of prova is as shown below in Listing 5.4

Listing 5.5: Prova syntax for constructing PSM legal rules

```
1   Variables:
2           (upper case)
3   Constants:
4           (lower case)
5   Fact:
6           availability(s1,99%).
7   Rule:
8           qos(S,high):- availability(S,99%).
9   Query:
10          :-solve (not(qos(S,high)).
11          :-eval (not(qos(S,high)).
12  Derive:
13          derive([X|Args]), ...
14  Scoping:
15          scope(p(X),  www.prova.ws  ), ...
16  Memoization:
17          cache(p(X)), ...
18  Lists:
19          [Head|Tail] = [Head,Arg2,...,ArgN] = Head(Arg2,...,ArgN)
20  Module Imports:
21          :- eval(consult("ContractLog/list.prova")).
22          :- eval(consult("http://rule.org/list.prova")).
23  Meta data annotation:
24           @key(value)
25           e.g. @label(rule1) r1(X):-q(X).
26            @qos@(S,medium) :- availability(S,98%).
27  Guards:
28          head_literal(args_h) :- body_literal1(args_1)[guards]
29          e.g. rule(X):- abc(X)[!].
```

As discussed before, an elementary legal norm, in its *normalized* form can be viewed as shown below:

Elementary Legal Norm = [Modal Operator] [Closed Formula].

Wherein, the first part holds the modal operator of the legal rule under discussion and the second part its LRML formula (closed formula). The SLE modalities- alethic necessity and deontic obligation- are handled using metadata annotation, **@obligation(value)** and **@necessity(value)**.

Metadata is a set of annotations, each of which is a pair defined as @*key*(*value*). Both *keys* and *values* can be arbitrary strings. Such annotations can be both on the literals of head and body of a rule. The annotations on literals (as opposed to rules) are a set of constraints on the target rules during unification. Now moving back to our considered example as shown in 5.3 (i.e. 35 U.S. Code § 112 - Specification), A set of SLE rules framed out of its procedural section may be as shown below:

(a) It is obligatory that office_action *includes* **Paragraph_7_30_01** if the claim *conceals* essential_subject_matter_requirement.

(b) It is obligatory that office_action *includes* **Paragraph_7_31_01** if the written_description *conceals* best_mode.

As shown before in section 5.8, an SSE expression 'R' is transformed into an FODAL sentence 'F' and further into an '$LRML$' formula by $\tau_{SLE}(R)$. While, representation of the FODAL sentence 'F' is a straight forward task, the modal operatos within the legal rule, *obligation* is handled as shown in Listing 5.6.

Listing 5.6: handling obligation and neccessity in prova

```
% @obligation(rule1) [office_action includes Paragraph_7.30.01]
                      :- [claim conceals essential_subject_matter]

@obligation(rule1) h1(X):- b(X).

@obligation(rule2) [office_action includes Paragraph_7.31.01]
                      :- [written_description conceals best_mode]

@obligation(rule2) h2(X):- b(X).

p1(X) :- @obligation(rule1) h1(X).
b(1).
b(2).

:- solve(p1(Y1)).
```

As seen in listing 5.6, metadata annotation to handle *obligation* can also be used within the body of a rule. It further can be extended to the literals withing the body of a rule. To show this, additional SLE legal rules, derived out of the legal section are considered.

(c) It is obligatory that examiner *rejects* claim if the office_action *includes* **Paragraph_7_30_01**.

(d) It is obligatory that examiner *rejects* claim if the office_action *includes* **Paragraph_7_31_01**.

The annotations on literals acts as a set of constraints on target rules during unification. It uses metadata matching if metadata is present in the body literal. Listing below provides an extension to the representation format from 5.6. It shows how *obligation* can be extended to the body literal of a rule, wherein it is scoped to *rule1* or *rule2*.

```
% [examiner rejects the claim] :- @obligation(rule1, rule2)
                            [office action includes Paragraph 7.31.01]

p2(X) :- @obligation(rule1, rule2) h1(X).
b(1).
b(2).

:- solve(p2(Y2)).
```

In order to handle the elementary (rule) pragmatics (i.e. the context information associated with the rule) on the platform dependent layer, we make use of prova *'guards'*. A rule is evaluated only if the guard conditions are evaluated to be True. The general syntax of Prova *Guard* is as shown in Listing 5.9.

```
% Prova Guards

    head_literal(args_h) :- body_literal1(args_1)[guards]
    e.g. rule(X):- abc(X)[X > Y].
```

Guards can include arbitrary lists of Prova literals including Java calls, arithmetic expressions or relations. Few operators handled by Prova *Guards* are as shown in Table **??**. Context information such as, *'applicable date'*, *jurisdiction*, *'source'*, *filed_in*, etc... can

Table 5.6: Prova *Guard* Operators

less than	$<$
greater than	$>$
equal	$==$
greater or equal	$\geqslant$
less or equal	$\leqslant$
not equal	$!=$

be transformed into individual constraints (i.e. [filed_in('US')], [jurisdiction('US')], etc..) or may be combined together to form a set of constraints as in, [filed_in('US'), jurisdiction('US'), examiner('USPTO'), includes(patent_application, claims)]. Which further are handled as Prova *Guards*.

Continuing with out example from Listing 5.6, we add the context information 'jurisdiction US' as constraint, which further is represented with the help of Prova *Guards* as shown below:

```

% <!-- <lrml:hasJurisdiction> US </lrml:hasJurisdiction> -->
% [examiner rejects the claim] :- @obligation(rule1, rule2)
                                    [office action includes Paragraph 7.31.01]

    p2(X) :- @obligation(rule1, rule2) h1(X)[hasJurisdiction(US)].

    :- solve(p2(Y2)).
```

Chapter 6 describes the proof-of-concept implementation, Knowledge Representation for Intellectual Property Law (KR4IPLaw) system.

5.10 Summary

The chapter provided a detailed description on a new framework for representing elementary legal (esp patent) rules with elementary pragmatics. To start with, a conceptual patent information system KR framework involving three interdependent modules were presented. Such a conceptual framework provided an overview on the sub-modules/actors in a legal information system and the flow of information between such them. Based on the conceptual framework, this thesis proposed a generalized process model for modeling legal rules. The process proposed being an open-ended, allowed legal sections from any NPSs as input and the output from such a process could be used as an input to any existing legal argumentation system.

Continuing with the process, the chapter provided several novel methodologies for representing legal information. The framework being loosely coupled with OMGs, MDA architecture, proposed representation of the legal information on three different layers. On the computation independent layer, the thesis proposed the use of decision models to represent the procedural aspects of legal rules. Further, such procedural information were captured in N:M relationship using a semi-formal representation structure. As a semi-formal representation structure, this thesis proposed an enhancement to the existing SBVR technology to fit to the domain of law.

Further, moving from the semi-formal representation to a formal representation format i.e. on to a platform independent layer, this chapter proposed a set of mapping rules, first to transform the existing SLE vocabulary into OWL2 ontology and later the legal rules represented in SLE into formal legal rule representation format. As a formal rule representation format, this chapter proposed KR4IPLaw-s, an ad-hoc representation format, which encompasses the important aspects of RuleML, LegalDocML and LegalRuleML. The chapter also provided the semantic transformation from SLE to KR4IPLaw via LegalRuleML.

Lastly, the chapter provided a detailed discussion on the representation of legal rules with elementary pragmatics on a platform specific layer. For such a representation, the thesis proposed the use of Prova, a semantic rule representation language and an inference engine.

Chapter 6

KR4IPLaw: Proof-of-Concept Implementation

Contents

This chapter describes the proof-of-concept implementation built to support the process described in Chapter 5. First this thesis introduces the general architecture of the system and later describes each module of such an implementation in detail.

6.1 System Architecture

The proof-of-concept system is referred to as the KR4IPLaw-System [113]. The KR4IPLaw is a system built to act as an interface, which can be easily handled by legal practitioners and is capable enough to provide all the necessary inputs as required for an knowledge engineer to model legal rules for (semi-/) automated reasoning thereafter. The system accommodates all the possible KRs (from formal to natural language) and act as a bridge between the legal practitioner and knowledge modeler.

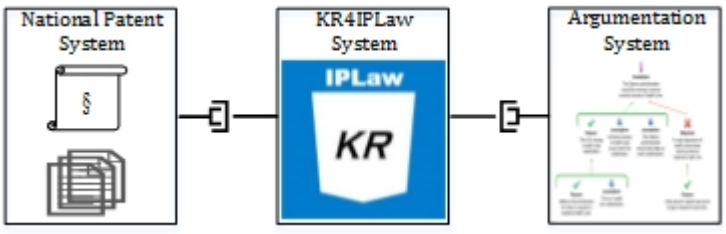


Figure 6.1: KR4IPLaw Use-Case Scenario

The KR4IPLaw system is implemented as an Eclipse RCP application, built on the Eclipse 4.3 (Kepler) platform. Figure 6.1, depicts one of the several use-case scenarios

of the KR4IPLaw system. The system is an open ended system as seen in the figure. It can accommodate any legal text from a NPS with its base language as English. The input to such a system are the legal sections/paragraphs in its natural language format and the output of such a text is a formally represented legal norm derived out of such legal sections. The output of the system can be used in any existing legal argumentation system.

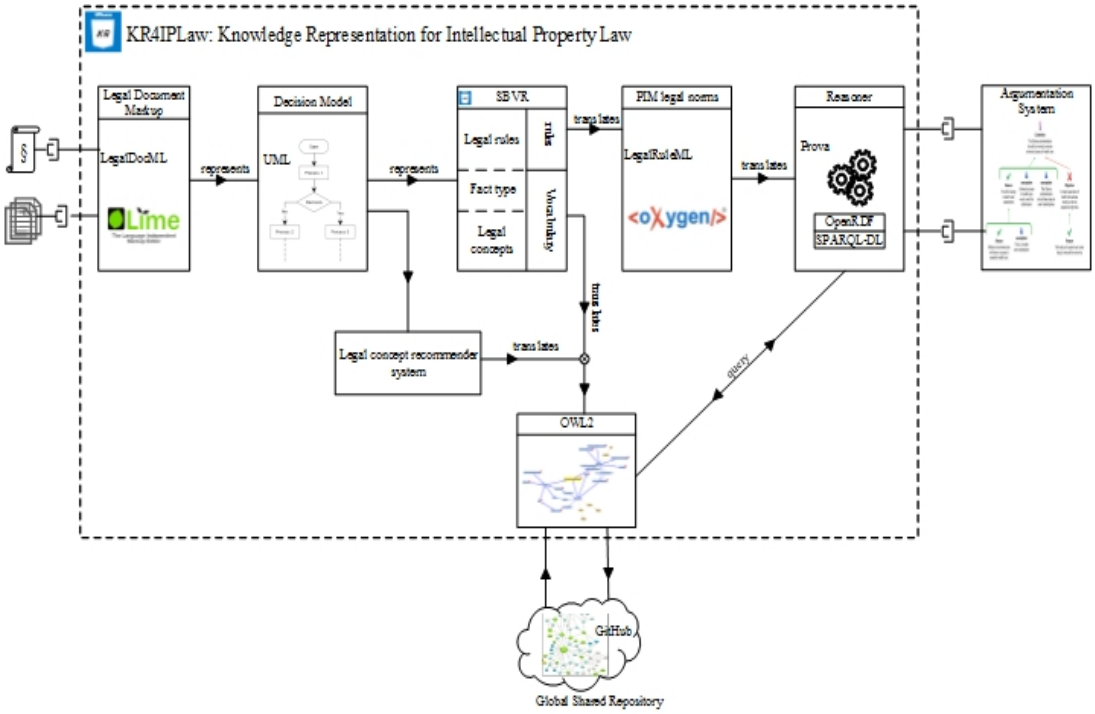


Figure 6.2: The KR4IPLaw System

The system architecture of the KR4IPLaw is as shown in Figure 6.3. The system comprises of several modules, each module is discussed in detail in the upcoming sections.

6.1.1 Annotation Module

The system integrates Language Independent Markup Editor (LIME) [128] for the purpose of legal document annotation. It is a customizable web-based editor integrated into the KR4IPLaw system, It guides the user through the markup of unstructured legal sections/paragraphs into well formed structured LegalDocML document. The web-based editor uses HTML5 standards to markup the documents, marked up elements are wrapped by a generic HTML element with various classes assigned to it for adding a semantic meaning. Further these HTML elements are mapped into LegalDocML representation format using XSLT style-sheets. This provides an additional advantage of having an HTML version of the document immediately without further conversions.

6.1.2 Judgment Miner Module

The proof of concept system, as a simple I/O system, considers formally represented legal sections as input and produces the most relevant judgment or a set of judgment's pertaining to the legal section. While the current system is tailored to work with LegalDocML files, it can be very easily extended to other formal representations.

The inner stack of the module comprises of few translators, a search engine, a topic modeler, and query constructors. Translators help in extracting the required information from a LegalDocML file and thereby translating it to a format necessary for further processing and vice-versa. As for the search engine, we integrate Apache Solr - a high performance, scalable search server built based on Apache Lucene. The advanced caching and replication, the distributed search and easy integration of other modules makes it a better choice than existing search servers such as Minion [129] or Sphinx[1].

Indexing content/all judgment is one of the important steps. CourtListner[2] provides, as a bulk download, the information pertaining to 361 jurisdictions of the United States courts. The data on CourtListener is a combination of many partial sources such as court websites, Public.Resource.Org, and a donation from Lawbox LLC. Thus, making it one of the most covered dataset available. The bulk data from the CourtListner is indexed by Solr.

For Solr scoring model, we use the Best Matching (BM25) algorithm [130], - a probabilistic Information Retrieval (IR) model against the well known TF-IDF - a vector space model for more precision.

Solr provides the ability to create different user defined queries through its API. Solr query support different search patterns such as term, field, wildcard, fuzzy, proximity or range searches. Table 6.1 provides the list of search patterns available within this module.

Table 6.1: KR4IPLaw Search Query Patterns

Type	Pattern	Example
Term	"text"	"Supreme Court"
Field	text:"text"	court:"Supreme Court"
Boolean	AND, OR, NOT	"Supreme Court" OR "CAFC"
Wildcard	? or *	Supre?me or Supreme*
Fuzzy	~	Supreme~
Range	TO	filed_date: [20150601 TO 200150801]
Boosting	^	"Supreme Court"^10

To increase the accuracy of the search results, an obvious step would be to construct complex queries, which combine multiple patterns. One such pattern is the query boosting parameter which utilizes certain keywords to boost the search query and thereby altering its default ranking.

While such a process of identifying keywords could be a manual operation, our module uses the topic model approach to automate the process of extracting and transforming

[1] http://sphinxsearch.com/
[2] http://www.courtlistener.com/

the keywords, to be used as boosting terms. We integrate Mallets'[3] statistical natural language processing (NLP) and topic modeling modules. Within the NLP module, we integrate the Snowball stemmer [131] and provide legal dictionaries (e.g. USPTO- glossary[4]) as an exclusion list. For the Topic Modeler module, we use the Parallel Topic Modeler [132], which realizes the Latent Dirichlet Allocation (LDA) model to compute the topics. The input to the topic modeler being the content part (i.e. legal paragraphs) of the LegalDocML file, we assume that each document is composed of only a few dominant topics and each topic is composed of only a few dominant words. With such an assumption, both values of the hyper-parameters, α and β are set to 0.01. The output from the topic modeller is used to build the complex query described before. Additionally, the most frequently occuring terms from particular domains of law (i.e. *patent* in the case of patent law) were explicitly excluded from boosting parameters list, as they are routinely used in judgements and therefore are of little use for ranking purposes.

The purpose built-in translator thereafter translates the obtained judgments from its native format to a required formal judgment representation format (e.g. LegalDocML). With the annotation of legal sections and its relevant judgments completed, the next step is to integrate them into a decision model so that further disaggregation and formal representation and reasoning can be performed.

6.1.3 Decision Model Module

The Decision-Model module is concerned with the conversion of the *context* part of the annotated Substantive Patent Law (SPL) into Procedural Patent Law (PPL), further into decision models. We make use of Unified Markup Languages (UMLs) activity diagram functionality to graphically represent the procedural norms into decision models. A Java based UML tool, UMlet [133] is used for creating decision models. It provides a simple interface to use/modify the shapes. The decision model is constructed with the basic available UML shapes. *'Diamonds'* represents decision points, *'rounded rectangles'* represent actions, arrows run from start towards the end and represents the order in which activities happen.

6.1.4 Structured Legal English Module

The KR4IPLaw- Structured Legal English (KR4IPLaw-SLE) module is a platform independent tool for assisting the semi-formal representation of procedural norms using Structured Legal English (SLE). The system is based on SBeaVeR [134]. The overview of the KR4IPLaw-SLE module is as depicted in Figure 6.3. Legal practitioners/domain experts either define case based vocabularies or use a pre-agreed vocabulary stored in a central public/privately shared repositories (such as GitHub) and build legal rules based on it.

As discussed in Section 5.7, the legal vocabularies are (semi/-)automatically transformed onto a formal representation format such as OWL2 using the mapping schemes

[3] http://mallet.cs.umass.edu/

[4] http://www.uspto.gov/main/glossary/

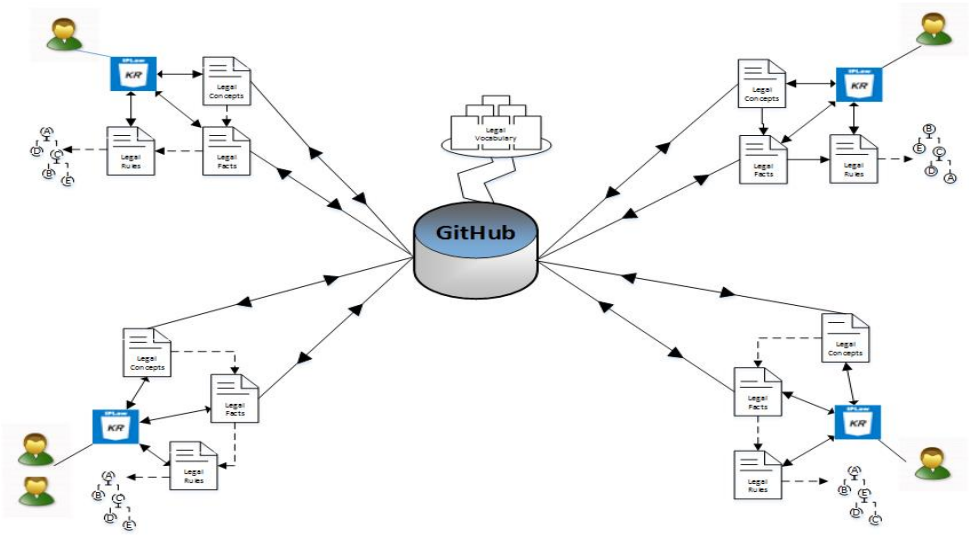


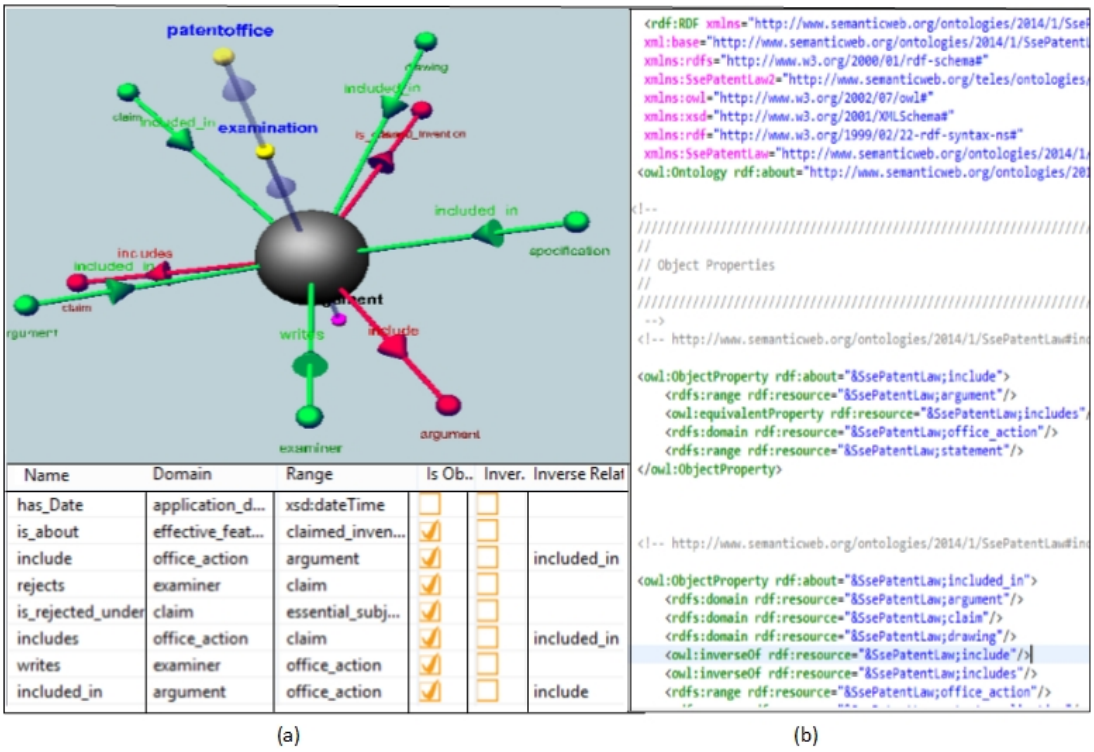
Figure 6.3: KR4IPLaw- Structured Legal English (KR4IPLaw-SLE) tool

proposed in Tables 5.4, 5.5, etc.. Such transformed pre-agreed formal vocabularies are stored and accessed through a central public/privately shared repositories such as OntoMaven [135] [136] or OpenRDF. The KR4IPLaw-SLE module, provides an interface which allows legal practitioners/domain experts, who have little technical skills in the field of Semantic Web, to graphically inspect, modify and review ontology components and a second interface which provides knowledge modelers with the tools for editing, which includes helpful features like syntax highlighting, auto-completion and error-detection, validation, dependency-check, etc.. for thus generated ontologies. KR4IPLaw tool makes use of SWeDE [5] and OntoSpere3D [137]. SWeDE integrates existing tools like OWL Validator, Kazuki and DumpOnt. OntoSpere3D uses a dynamic collapsing mechanism to render a rich intuitive interface featuring *rotation, panning, zoom, selection, etc...* Figures 6.4 (a) and 6.4 (b), depict the two interfaces (legal vocabulary management), with snippets from formal legal vocabulary built based on the example 5.3.

6.1.5 Legal Concept Recommender Module

As seen in the previous chapter, SLE provides an efficient solution by allowing a non-technical domain experts to engineer knowledge (vocabularies and rules) in natural language, with an underlying semi-formal semantics defined in the SBVR standard. But manual modeling of the legal vocabularies from legal text is one of the most time consuming parts of the legal knowledge engineering [113]. In addition, the non-existence of a global public/privately shared vocabulary makes the task of building legal vocabulary more tedious and time consuming. As a solution to this problem, this thesis contributes with a semi-automated vocabulary (and thereby, rule development) development process which is supported by automated suggestions of legal concepts computed by a semantic legal text analysis.

[5] http://owl-eclipse.projects.semwebcentral.org



Name	Domain	Range	Is Ob..	Inver.	Inverse Relat
has_Date	application_d...	xsd:dateTime	☐	☐	
is_about	effective_feat...	claimed_inven...	☑	☐	
include	office_action	argument	☑	☐	included_in
rejects	examiner	claim	☑	☐	
is_rejected_under	claim	essential_subj...	☑	☐	
includes	office_action	claim	☑	☐	included_in
writes	examiner	office_action	☑	☐	
included_in	argument	office_action	☑	☐	include

(a) (b)

Figure 6.4: KR4IPLaw-SLE module interfaces

To achieve the said (sub-) process, the system integrates an additional conceptual functionality in the architecture of KR4IPLaw through a terminology recommender system. A recommender system (semi-)automatically translates the procedural legal texts from the decision models into semantically enriched legal concepts. Such a system, complementary to KR4IPLaw helps to fill the gap between a legal vocabulary/rules built by legal practitioner and all possible concepts/rules which can be identified by the automated system. At the time of this thesis, it is strongly believed, that in a legal domain, as an effect of the pragmatics involved, it is rarely possible for a system to fully automate the entire process of building legal rules/vocabularies accurately. Human intervention in confirming the system's automatically generated results is needed in an iterative process during the whole knowledge engineering and formalization process. The recommender system proposed here should provide the required additional context information that can be derived out of the legal context in which a legal vocabulary is built (e.g., case-law, definitions, synonyms etc. pertaining to the section/legal text under consideration).

For the purpose of legal concept recommendation, we consider a small subset of the available Semantic-Knowledge Extraction (S-KE) tools suitable for its application to our considered legal domain:

6.1.5.1 AlchemyAPI

AlchemyAPI [138], is a tool which employs the methods of deep linguistic parsing, statistical natural language processing, and machine learning for named entity extraction,

keyword extraction, fact and relation extraction, document categorization, concept tagging and language detection. It builds upon semantic web functionality, AlchemyAPI concepts and entities are linked to DBpedia, Freebase, OpenCyc, GeoNames etc. It is available as a Web application or as a REST service.

6.1.5.2 DBpedia Spotlight

DBpedia Spotlight is a tool for automatically annotating entities in text as DBpedia resources, providing a solution for linking unstructured information sources to the Linked Open Data cloud through DBpedia. It is available as a Web application, as a REST service or as downloadable source [139]. Also language specific versions exist, e.g. DBPedia German[6].

6.1.5.3 NERD

NERD [140] proposes a Web framework which unifies numerous named entity extractors using the NERD ontology which provides a rich set of axioms aligning the taxonomies of these tools. Extractors supported by NERD are AlchemyAPI, DBpedia Spotlight, OpenCalais, etc..

6.1.5.4 FRED

FRED is a tool for automatically producing RDF/OWL ontologies and linked data from natural language sentences. It links the extracted knowledge to both lexical linked data and datasets. It is available as a Web application or as a REST service [141].

A feature based comparison of the considered S-KE tools is as shown in Table 6.2. The comparison is based on features such as identification of *topic* under discussion [142], identification of atomic elements in a text belonging to a predefined category also known as Named Entity Recognition (NER) [143], determination of the correct meaning of a polysemous legal word in a given legal context also known as Word Sense Disambiguation (WSD) [144], Semantic Taxonomy (TAX) identification and relation identification [145], identification of Semantic Roles (SemRole) against an extracted concept [145], *event* extraction from texts [146] [147] and semantic frame recognition [148].

Table 6.2: Feature based comparison of the S-KE tools for legal concept recommendation (adapted from [149])

	Topic	NER	WSD	TAX	Rel	Sem Role	Events	Frames
AlchemyAPI	Yes	Yes	Yes	No	Yes	No	No	No
DBpedia- Spotlight	No	Yes	Yes	Yes	No	No	No	No
NERD	No	Yes	Yes	No	No	No	No	No
FRED	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes

[6] http://www.corporate-semantic-web.de/dbpedia-deutsch-spotlight.html

As seen from Table 6.2, FRED offers more features than its considered counterparts. Based on performance evaluation of the S-KE tools as shown in [149], FRED at the time of review provides better results than DBpedia, AlchemyAPI and NERD. FRED, adapted to the legal domain considers a legal sections/text given in its base (English) language as an input to produce semantic data and ontologies with a quality closer to what is expected at-least from average linked data-sets and vocabularies by passing through DRS [150] produced by Boxer [7] [151] [152]. It includes NER (based on Apache Stanbol) and WSD.

FRED offers several functionalities [153] as required by any legal recommender system. Some of the functionalities supported are as stated below.

(a) Captures accurate semantic structures,

(b) Represents complex relations,

(c) Supports integration of sophisticated lexical reasoners (like OpenNLP [8], VerbNet, FrameNet[9])

(d) Supports open information extraction,

(e) Maps natural language to RDF/OWL and

(f) Links the extracted knowledge to both lexical linked data and datasets (WordNet, DB-pedia and other foundational ontologies.)

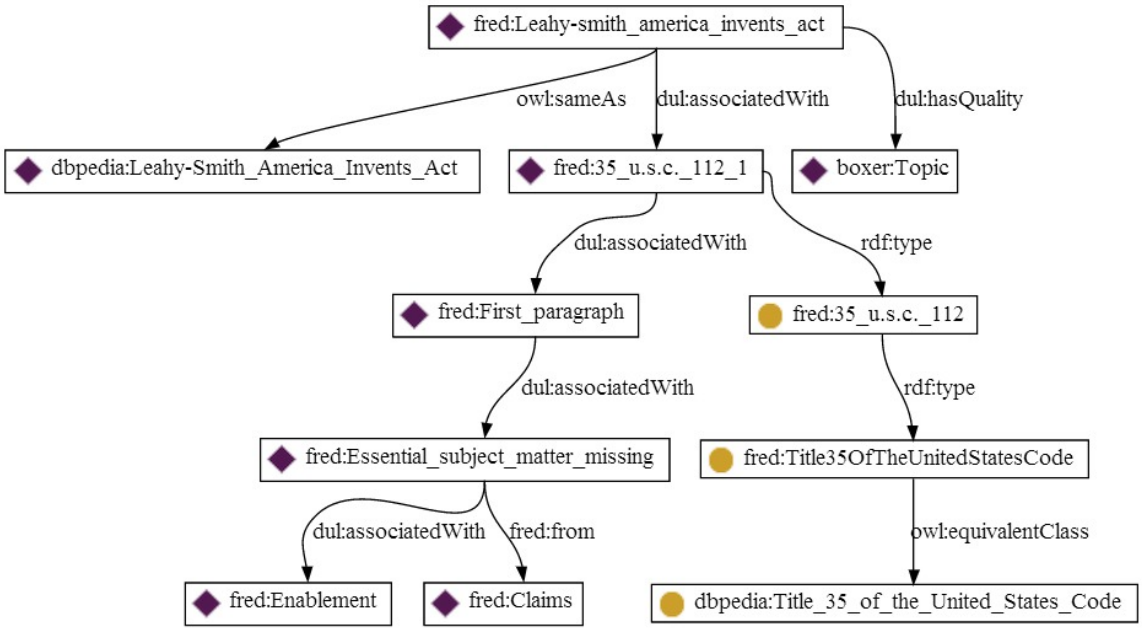


Figure 6.5: FRED's output for a legal text

[7] http://svn.ask.it.usyd.edu.au/trac/candc/wiki/boxer
[8] https://opennlp.apache.org/
[9] http://verbs.colorado.edu/semlink/

Fig 6.5, shows a snippet of the output for a legal text out of the paragraph ¶ 7.30.01. With the legal text provided as an input, the next step requires the extraction of the required semantic information out of the obtained RDF/OWL ontology. The extracted information is thereafter used to enrich the existing legal vocabulary. The required information is extracted using SPARQL queries and then mapped to legal vocabulary with the help of a mapping scheme as proposed in Table 6.3.

Table 6.3: Mapping scheme: Parsed legal text to legal (SLE) vocabulary

RDF class	$\longmapsto$	Legal (Noun) Concept
rdfs:subClassOf	$\longmapsto$	General Concept
owl:sameAs	$\longmapsto$	Synonyms
owl:equivalentClass	$\longmapsto$	Synonyms
wn:lang	$\longmapsto$	Language
wn:gloss	$\longmapsto$	Legal Concept Definition
$\langle$ boxer$\rangle$: hasModality	$\longmapsto$	Necessity/Possibility
boxerpos/pennpos: 'v'/'VB'	$\longmapsto$	Legal (verb) concept
boxerpos/pennpos: 'n'/'NN'/'NNS'	$\longmapsto$	Legal (Noun) Concept
boxerpos/pennpos: 'np'/'NNP'/'NNPS'	$\longmapsto$	Legal (Individual Noun) Concept

Thus obtained legal vocabulary, is further (semi/-)automatically transformed onto a OWL2 formal representation format using the mapping schemes proposed in Tables 5.4, 5.5.

A part of the recommender system involving (semi/-)automatized building of legal rules is still an open research question. There have been several works in automatic extraction of SBVR business rules [154] [155] [156] [157]. Adapting it to legal domain has shown high inconsistencies between the actual legal texts to its constructed legal rules. The architecture of FRED is designed to allow the use of domain specific legal lexical resources, which includes the knowledge base (legal vocabulary) built during the semi-formal representation of the procedural legal rules.

6.1.6 Knowledge Base

The knowledge created during the process are stored as OWL2 ontologies. To support reasoning, core-ontology depends on several external ontologies (such as, *case-laws*, *NPS-definitions*, etc...) to provide the intended semantics. In order to handle such dependencies existing within the core-ontology, we make use of OntoMaven [135], a collaborative agile ontology development, life cycle management and transitive dependency management technology.

OntoMaven adopts Maven's artifact concept [10] by describing ontologies as ontology artifacts in a Maven Project Object Model (POM). The dependency management is handled by the OntoMvnImport plugin, the dependencies are described in the POM file. A simple example showing the snippet of the POM file handling dependency is as shown in Listing 6.1.

[10] http://maven.apache.org/index.html

Listing 6.1: OntoMaven POM snippet for handling dependencies

```
<project>
   ...
   <dependencies>
      @<dependency>@
         <groupId>us.onto.maven</groupId>
         <artifactId>CaseNumberOntologie</artifactId>
         <version>1.0</version>
      @</dependency>@
      ...
   </dependencies>
</project>
```

OntoMaven, checks the existence of imported ontology referenced by Uniform Resource Identifier (URI) *import* statement and downloads the ontology artifacts from remote repositories to a local/central development repository. It maintains an updated list of reference URIs to the ontology resource loaded to the repository and supports automated replacement of the URI references to imported external ontologies with references to internal ontology artifacts, managed by OntoMaven.

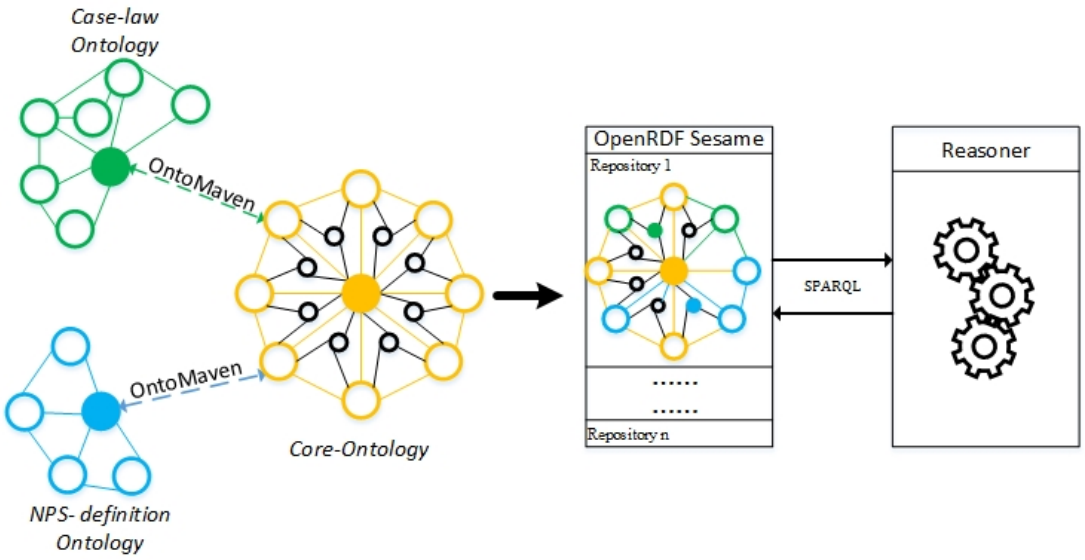


Figure 6.6: KR4IPLaw Knowledge Base

Further, the core-ontology (derived from previous section) along with its dependencies are pushed to a global/central repository. KR4IPLaw supports two possibilities to handle such global/central repositories. Knowledge engineers could either make use of the OntoMavens *'OntoMvnSvn'* module to handle central repositories or as this thesis proposes, make use of OpenRDF Sesame [11] an de-facto standard framework for processing and storing RDF data. OpenRDF Sesame also supports the SPARQL query language for expressive querying and offers transparent access to remote RDF repositories using the same API as for local access. Figure 6.6, depicts the knowledge base module of the system.

[11] http://www.openrdf.org/index.jsp

6.1.7 Reasoner Module

For reasoning, the platform independent legal rules are transformed into platform specific elementary legal rules. We use Prova [124], a rule language and rule engine for legal rule representation (platform specific) and for reasoning such rules on top of legal knowledge bases. Prova is both a Semantic Web rule language and a high expressive distributed rule engine. It supports the execution of declarative (decision) legal rules [158], access to external semantic web data via SPARQL, ontology reasoners and supports scoped reasoning. It is designed specifically for distributed information, knowledge and computation integration i.e. to support separation of logic, data access and computation as required for reasoning legal rules described in this thesis.

Prova provides a tight combination with Semantic Web technologies to integrate external Semantic Web data. Prova SPARQL operators are based on OpenRDF Sesame, which is an open source Java framework for storage and querying of RDF data. Also, Sesame offers an easy-to use API that supports the leverage of RDF data in applications. In particular, it can run in a standalone server mode with multiple applications connecting to it. Based on the tight integration with Java, Prova embeds Sesame Java API in declarative rules and outsources storage and querying of RDF data to a Sesame server. Such solution not only reduces the complexity of Prova, but also improves the flexibility of SPARQL querying.

Five Prova built-in predicates shown below, encapsulates the Sesame API to query RDF data

(a) ***sparql_connect***: Used to connect to the repository.

(b) ***sparql_disconnect***: Used to disconnect from the repository.

(c) ***sparql_select***: Performs the query stores the results as the facts of predicate *sparql_results*.

(d) ***sparql_ask***: Performs the query and returns a boolean whether a query pattern matches or not.

(e) ***sparql_results***: Query results identified by an identifier *queryId*.

Prova to incorporate SPARQL- DL query engine [159]. The SPARQL-DL query language is a subset of SPARQL and is explicitly tailored to ontology-specific requests related to OWL. It uses SPARQL syntax and is more expressive than existing DL query languages by allowing to mix TBox, RBox, and ABox queries. Figure 6.7 shows the architecture adapted from derivo GmbH into KR4IPLaw system. The SPARQL-DL query engine is settled on top of the OWL API.

The latest implementation of SPARQL-DL API (1.0.0) by derivo GmbH in 2011 is fully compatible with OWL 2 and can be regarded as an interface to every ontology reasoner supporting OWL API. In this thesis, the KR4IPLaw employs Hermit [160] [161] , which is a Java-based OWL reasoner, to act as a real reasoner to reason domain ontologies.

The EP's represented as meta-data acts for a considered legal rule as explicit scope for constructive queries on the knowledge base. In addition to scopes, we make use

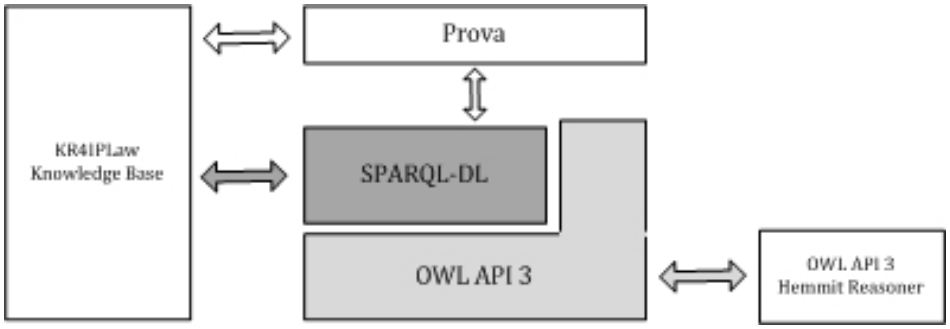


Figure 6.7: SPARQL-DL module architecture

of the *guards* functionality provided by Prova. *Guards*, act as additional pre-condition constraints (e.g. reasoning only those rules from trusted authors).

6.2 Summary

This chapter presented a proof-of-concept system, Knowledge Representation for Intellectual Property Law (KR4IPLaw), to support the process, in-turn the framework defined for representing and reasoning elementary legal rules with elementary pragmatics.

The chapter initially presented the general system architecture envisioned for the process followed by a detailed description of each module. Annotation module discussed the integration of an open source legal document markup editor, LIME. Judgment Miner module provided an implementation for automatizing the process of identifying the most relevant judgment pertaining to a legal section and further transforming them into a formal representation format. To fine tune the retrieved judgments sorting order, we proposed an approach wherein topic modeling using LDA was used. Further these judgments were transformed into a formal representation format using a purpose built translator. After which, the annotated legal section and its related judgments can be mapped into a decision model for down the line processing. The decision-model module extended UMLet, a java based UML tool for creating decision models. As a semi-formal representation module, this thesis introduces a tool, KR4IPLaw-SLE for representing legal rules in a SBVR-SE based controlled natural language format. Also introduced were two visualization modules for OWL2 ontology GUI-access (formal transformation of legal vocabulary). Later we introduced, a legal concept recommender module for assisting in filling the missing context information pertaining to a legal vocabulary. We introduced a reasoner module, in which we integrated SPARQL-DL, a OWL query engine in to Prova, a rule engine as a reasoner module to our existing KR4IPLaw system. As a proof-of concept implementation, in later section 7.3, we introduce a module for evaluating legal vocabulary transformations using CQs.

Chapter 7

Evaluation

Contents

A Knowledge Based System (KBS) may be evaluated on two levels:

(a) Knowledge level evaluation.

(b) System level evaluation.

A system level evaluation not only considers the knowledge in the system, but also the inferencing mechanism, knowledge acquisition mechanism and on how user intuitive it is. Such overall evaluation is necessary for any system, however it requires each component to be individually correct. This thesis partially covers both type of evaluation. A complete evaluation of all the components in defined is out of the scope of this thesis.

This chapter provides a detailed evaluation of the process and of the Proof-of-concept system KR4IPLaw described in previous chapters. Some modules of the system/process are dealt independently, while others are dealt as a group for the purpose of evaluation. Finally, this chapter concludes with the empirical evaluation of the entire process (with the help of KR4IPLaw)

Each section, first introduces the evaluation methodology by providing some background information and later describes how such methodology is applied in this thesis, thereby evaluating the process itself.

7.1 Case-Law retrieval evaluation

7.1.1 Gold Standard

The proposed approach is evaluated on the well known precision-recall metric of information retrieval. An ordered set of judgments based on its relevance (i.e. on case-law citations [1]) to a legal section under consideration was created (a gold-standard or *ground truth judgment of relevance*)[2]. To evaluate the system for its cross domain performance, the gold standard included a minimum of 5 legal sections each from different Intellectual Property (IP) Laws of the Code of Laws of the United States of America (U.S.C). As shown in Fig 7.1, the legal sections from US Patent Law, Copyright law and Trademark law were considered. 20 most relevant, based on its number of citations pertaining to each legal section were identified. Relevance based on a citation count approach providing an easy-to-compute, context-free, and relatively accessible metric, may, in fact, be better for finding high-quality documents than sophisticated human or pattern recognition queries and models [162]. In total, the gold standard comprised of 300 judgments from courts of 361 jurisdictions within the US.

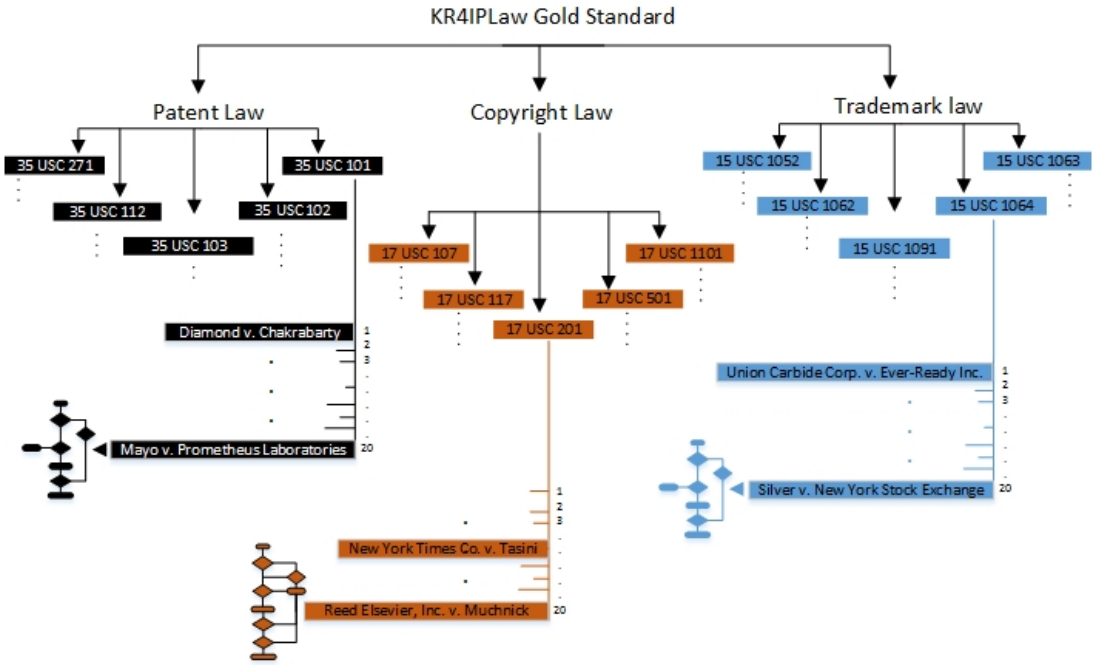


Figure 7.1: Gold Standard Structure

[1] While an ideal approach is to use a expert driven gold standard construction, a common consensus on thus generated standards for relevance is debatable in the context of case-laws.

[2] http://github.com/shashi792/KR4IPLaw-Act2Judgement/KR4IPLaw_Gold_Standard.xlsx

7.1.2 Precision & Recall

A metric, F-measure i.e. the harmonic mean of the precision and recall is used for depicting the results.

$$F_1 score = 2 \times \frac{precision \times recall}{precision + recall} \tag{7.1}$$

Wherein;

$$precision = \frac{|\{relevant\ documents\} \bigcap \{retrived\ documents\}|}{|\{retrived\ documents\}|}$$

and

$$recall = \frac{|\{relevant\ documents\} \bigcap \{retrived\ documents\}|}{|\{relevant\ documents\}|}$$

The F-measure for the judgments retrieved from the judgment-miner module is as shown below in Fig 7.2. The F-measure is plotted for legal sections from different domains of the intellectual property law.

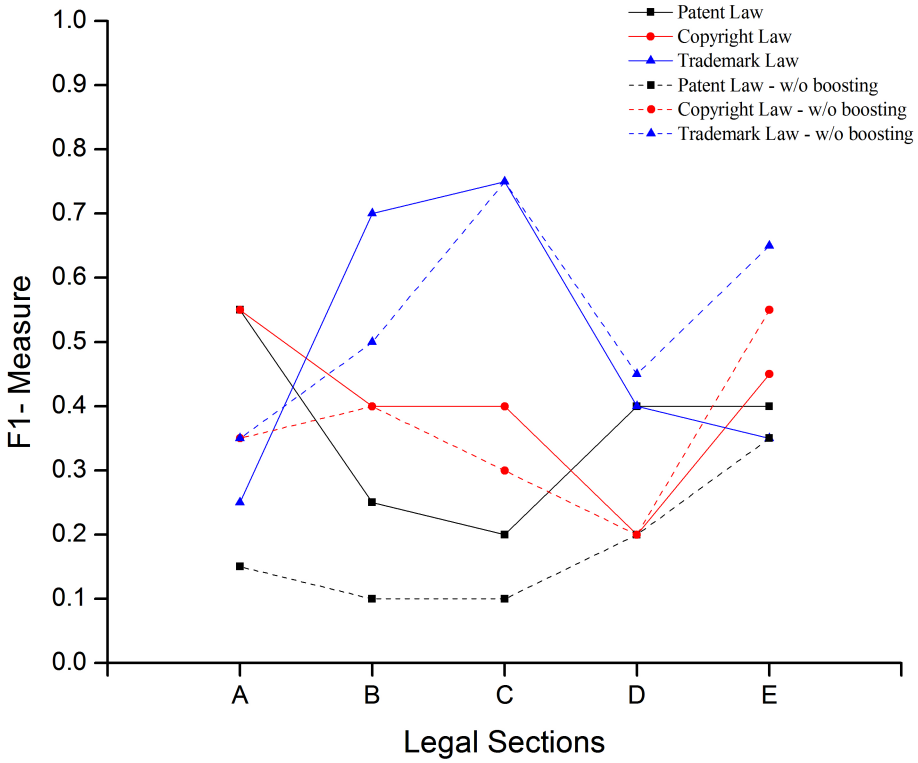


Figure 7.2: F-measure

While vocabularies from US patent, trademark and copyright laws were introduced as an exception list for stemming, an observation made during this study was the necessity to perform a proximity search while indexing. This has been due to the fact that judges responsible for writing decisions use a slightly non-aligned nomenclature such

as `35 USC 112 (a), 35 USC 112 First Paragraph or 35 USC sec 112`, while referencing legal sections within a judgment.

While the use of boosting keywords resulted in significant result improvement in the domain of patent law, the outcome for copyright and trademark law remains satisfactory. Further improvement of results could be achieved by fine-tuning the proximity search method in case of those two domains.

7.2 Semi-automated legal vocabulary building evaluation

As described before, the process of vocabulary building for structured legal english is accomplished by a terminological recommender system. This section focuses on evaluation of such a system. As a part of the evaluation, we adapt the performance evaluation of NLP tools, proposed by Hirschman and Thompson [163] and its derived methodology proposed in [155]. We assume that a legal practitioner builds a legal vocabulary from scratch to suit, e.g., case-law requirements as an alternative to use/build the existing (pre-agreed) shared vocabulary.

Figure 7.3, shows a Venn diagram depicting different terms (and its relations) used in this methodology. Building legal arguments (based on legal rules) being the main concern

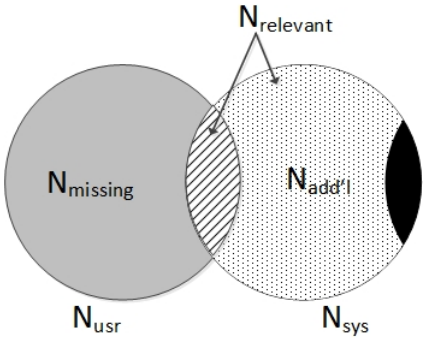


Figure 7.3: Venn diagram

in this evaluation study, a legal practitioner is only interested in the concepts required to build legal rules and rule-based arguments (i.e., $N_{legal(Noun)}$ concepts, $N_{Legal(verb)}$ concept, and $N_{legal(Indv)}$concepts). In this evaluation we study to which extent the system is capable of filling/enriching the semantic information attached to each legal concept.

N_{usr} here denotes the inputs from the user in building the legal vocabulary, (where N denotes the number of respective items added). N_{sys} denotes the systems effort in identifying the information/items related to this section of legal text under consideration. $N_{relevant}$ refers to the items that are relevant/meaningful out of the identified items by the system (i.e N_{sys}). The relevance of an item is determined by a domain expert. $N_{missing}$ refers to the difference in number between the items that are relevant and the items that were used/identified by the user/legal practitioner. $N_{add'l}$ refers to the additional

Table 7.1: Recommender system outcome analysis

	N_{usr}	N_{sys}	$N_{relevant}$	$N_{missing}$	$N_{add'l}$
Language (Legal concepts)	0	1	1	0	1
Definitions identified	0	4	4	8	0
General concepts identified	0	14	2	NA	2
Synonyms identified	0	4	2	NA	2
$N_{Legal(Noun)}$concepts identified	12	14	9	8	5
$N_{Legal(Verb)}$concepts identified	6	5	3	5	2
$N_{Legal(Indv)}$concepts identified	1	4	4	0	3

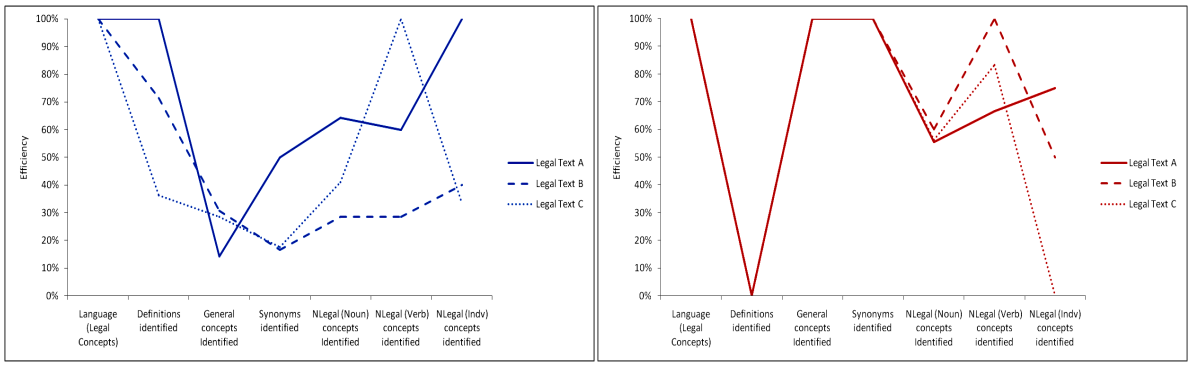
relevant items identified by the system which are currently not used by the user. To evaluate the efficiency of such systems, we consider two parameters, $Eff_{sys\ vs.\ relevant}$ and $Eff_{relevant\ vs.\ add'l}$ as shown below:

$$Eff_{sys\ vs.\ relevant} = \frac{N_{relevant}}{N_{sys}} \times 100\% \Rightarrow \text{System Reliability} \tag{7.2}$$

$$Eff_{relevant\ vs.\ add'l} = \frac{N_{add'l}}{N_{relevant}} \times 100\% \Rightarrow \text{System Intelligence} \tag{7.3}$$

wherein;
$Eff_{sys\ vs.\ relevant}$ denotes the efficiency of the system in identifying relevant/meaningful items in a given legal passage/text and $Eff_{relevant\ vs.\ add'l}$ refers to the efficiency of the system in providing additional information out of its identified relevant items. We consider the example shown (i.e. decision point 'B2' from Figure 5.7) in section 5.6 as an input to the recommender system. Table 7.1 shows a chart comprising of both inputs from the user as well as from the system. The Efficiency of the system is as shown in Figure 7.4 (i.e. Legal Text A).

Figure 7.4 gives the results of the evaluation on two additional legal paragraphs (denoted here as legal texts 'B' and 'C'). Specifically, Fig 7.4(a), shows the efficiency $Eff_{sys\ vs.\ relevant}$ and Fig 7.4(b), shows the efficiency $Eff_{relevant\ vs.\ add'l}$.



(a) $Eff_{sys\ v.\ relevant}$ (b) $Eff_{relevant\ v.\ add'l}$

Figure 7.4: Efficiency evaluation

7.3 Legal vocabulary and its transformation evaluation

The process of knowledge creation and its transformation for further reasoning requires evaluation. In 5.7, with the help of different transformation rules, the thesis described approaches of moving from computational independent layer (i.e. SBVR-SE) to a platform independent layer. Evaluation of such transformation is indispensable, as - for example - it increases intended users confidence in using the knowledge base for further operations such as deriving the inferences for an in-court argumentation, etc.. The evaluation of legal vocabulary and its transformation involves *'Verification'* and *'Validation'*.

As per intelligent systems design,

(a) *'Verification'* is understood as the process of ensuring;

  i) that the intelligent system conforms to specifications, and

  ii) its knowledge base is consistent and complete within itself [164] and

(b) *'Validation'* is the process of ensuring that the output of the intelligent system is equivalent to those of human experts when given the same inputs [164].

As the collaboration between a knowledge engineer and a domain expert is an inevitable part of ontology development [165], the methods used for evaluation have to account for their diverse background and be accessible for both of them. In this respect a set of competency questions, as defined informally by the domain expert, can be used as a basis for ontology evaluation. The usage of informal CQs for ontology requirements description and its further evaluation has already been accounted for by a number of ontology design methodologies, including: Uschold and King's [166], Grüninger and Fox's [167] and Methontology [168]. Similarly, in the rules domain, amongst other methods, the verification and validation of rule bases using CQs have been proposed by Paschke in [169]. The evaluation of ontologies with the use of CQs have been addressed in a few interesting domains such as medicine [170][171] or software development [172]. Khan in [165] has also proposed a template-based approach of competency questions generation in the enterprise modelling domain. A notable omission, until now, was therefore in the field of legal ontologies, wherein, either the CQs have been used for their design, but not evaluation [173] or their evaluation was performed without the use of CQs [174]. Additionally, we supply a domain expert with a set of easy-to-understand metrics that represent a quantitative view of ontology quality. According to [175], knowledge engineering consists both of methods (used for knowledge acquisition, system design and its implementation) and measures used to evaluate them. As such, there already exist a number of methodologies that employ the quantitative approach (e.g. OntoMetric [176]). This thesis adapts the metrics defined as a part of OntoQA ontology evaluation tool, as it works not only on schema-level, but also on populated ontology level.

Further sub-sections of this chapter introduces an approach wherein CQs are used for both verification and validation. The evaluation is based on a set of questions regarding the subject-matter knowledge and pre-defined answers the ontology is supposed to give.

The answers to such CQs could be viewed as functional requirements of thus constructed ontology. The design of questions and answers evaluation can prove to be a great tool that enables the cooperation between the legal expert and knowledge modeler.

Four actors can be distinguished in the process of questionnaire design and application: content specialist, design specialist, interviewer and respondent [177]. The content specialist and design specialist may represent different fields of expertise, similarly to the legal expert and knowledge modeler in our case. Therefore those four actors can be mapped to the actors in our process as shown in Table 7.2.

Table 7.2: Actors in the process of ontology evaluation using CQs

Content specialist	↔	Legal practitioner (creates a set of CQs defined in semi-formal language)
Design specialist	↔	Knowledge modeler (performs the translation into formal query language)
Interviewer	↔	Ontology query engine
Respondent	↔	The Ontology itself

7.3.1 Verification

Verification can be viewed as a part of validation. There are a number of ontology quality criteria that can be taken into account in the verification process [178]. Such criteria are;

(a) Accuracy: Are the axioms of the ontology in line with stakeholders' knowledge?.

(b) Adaptability: Can the ontology be extended easily?.

(c) Clarity: Is the ontology easy to understand?.

(d) Completeness: Is the problem domain sufficiently covered?.

(e) Computational Efficiency: How efficient are the tools used with the ontology?.

(f) Conciseness: Is there any amount of redundancy in the ontology?.

(g) Consistency: Does the ontology include or allow any contradictions?.

(h) Organizational Fitness: Does the ontology conform to particular organizations need?.

It should be noted that not all of those quality criteria are of immediate interest in case of our system. We are not concerned with ontology quality *per se*: the aim is to verify the transformation itself. Therefore, the aspects that are to be put under scrutiny are as follows: whether ontology complies to the knowledge of the stakeholders, already expressed in semi-formal representation (*'Accuracy'*), can be easily expanded (*'Adaptability'*) and is it the complete translation of semi-formal SSE (*'Completeness'*). All these aims can be fulfilled with the use of questionnaire method using competency questions (CQ) [178].

7.3.2 Validation

Validation of ontologies in general refers to real world content evaluation. When a given ontology is an application ontology (as in our case), the validation can only include ontology checking from domain experts and users against the real world. Even-though, there exist few techniques to address this issue of validation, very few address it from end-user/domain expert perspective. We adapt the methods proposed by Lovrencic and Cubrilo [179] and Tatir et.al [180] for the legal vocabulary and its transformation validation.

Validation of legal vocabulary (or formal ontology) can be based on certain important characteristics/aspects that a vocabulary needs to fulfill inorder to be valid. Few such characteristics/aspects are as shown below:

(a) (In-)consistent: A legal concept(or knowledge defined within) is consistent if no-contradictory knowledge exist or can be inferred from the vocabulary.

(b) (In-)complete: A legal vocabulary is complete if all knowledge required is explicitly stated or can be inferred from the vocabulary.

(c) (In-)concise: A legal vocabulary is concise if it does not contain definitions unrelated to the vocabulary itself.

(d) (In-)sensitive: A vocabulary is sensitive, if small changes to a legal concept alters a set of well-defined concepts.

Validation of an vocabulary on based on these aspects is generally qualitative, and done by a domain expert/legal practitioner. Proving for consistency, completeness, conciseness or sensitivity of knowledge is still a *subjective* issue on defining what is a complete knowledge, but it is easier to detect if something is contrary to it. Through this approach, we try to bind such validation aspects, to aspects that provide an quantitative evaluation through CQs. Quantitative evaluation uses a metric based ontology evaluation approach. Whereby, the answers to the CQs offer quantitative perspective rather than an *effective/ineffective* or *good/bad* answer.

Legal vocabulary may be regarded as incomplete, if the legal concepts do not contain of all the necessary definitions, some defined formal and some via its annotation properties. Identification of legal concepts with missing properties *may* point towards a possibility of some missing knowledge, thereby showing its (In-)completeness.

A legal vocabulary may be termed as (In-)concise if there exist legal concepts which have not further been utilized. These are concepts which have been defined in the vocabulary, but have not been used to build further facts or rules needed. Identification of such concepts may be done using the *Class Richness* or *Relationship Richness* identification approaches which are defined as follows:

$$Class\ Richness = \frac{\text{Total No. of non-empty classes}}{\text{Total No. of classes}}$$

$$Relationship\ Richness = \frac{\text{No. of relationships used at instance level}}{\text{No. of relationships defined at schema level}}$$

To validate the sensitiveness of a legal concept within a legal vocabulary, we use the *Class Importance* aspect. It points to the domain expert/legal practitioner what legal concepts are important in this domain, thereby, specifying any changes to this concept requires changes to other legal concept definitions. *Class Importance* may be defined as follows:

$$Class\ Importance = \frac{\text{No. of instances of inheritance sub-tree rooted at a class}}{\text{Total No. of class instances}}$$

While (In-)consistency is handled in a way similar to the logic based approach, the explanations of errors for (In-)consistencies detected by the reasoner are converted into a more semi-formal level for a domain expert to work on. Sub-section 7.3.4, proposes an CQs based process for the evaluation of a legal vocabulary and its transformation.

7.3.3 Formal definition of CQs-based evaluation

Formal competency questions can be defined formally as an entailment or consistency problem with respect to the axioms in the ontology. Addressing such problems also addresses a part of the functional requirement verification of the legal ontology. If $T_{ontology}$ denotes a set of axioms and T_{ground} is a set ground literals with Q, a first-order legal sentence in the language of $T_{ontology}$, then every competency question should address two generalized conditions inorder for the transformation to be correct and verifiable.

(a) Determine if $T_{ontology} \bigcup T_{ground} \models Q$

(b) Determine if $T_{ontology} \bigcup T_{ground} \not\models \neg\, Q$

7.3.4 Process for CQs based evaluation

The process of such evaluation is depicted in Fig 7.5, and is used as a reference point in the description that follows. Firstly, a number of templates is prepared. They are used by the legal expert for the creation of informal CQs that are used for an ontology evaluation. Our systems is also extensible, as we provide procedures for new template addition, in case a legal expert's intended CQs do not conform to any of the already provided templates. In that case CQs, informally expressed in Step 1, are provided to the knowledge modeler, who identifies patterns in those questions (Step 2). In that step a consultation with legal expert may be needed for the clarification of his informal CQs. Those - now formally defined - CQs will serve as a basis for a body of new templates that are used for system extension (Step 3). Conversely, if the legal expert finds the suitable template (Step 4), it can be used for the automatic generation of a formal competency question (Step 5). We have created a tool that can automatically transform those into DL-queries (Step 6),

which are then executed against the knowledge base (Step 7). Finally, the process results can be used for the verification as well as validation of the created legal knowledge and its transformed form.

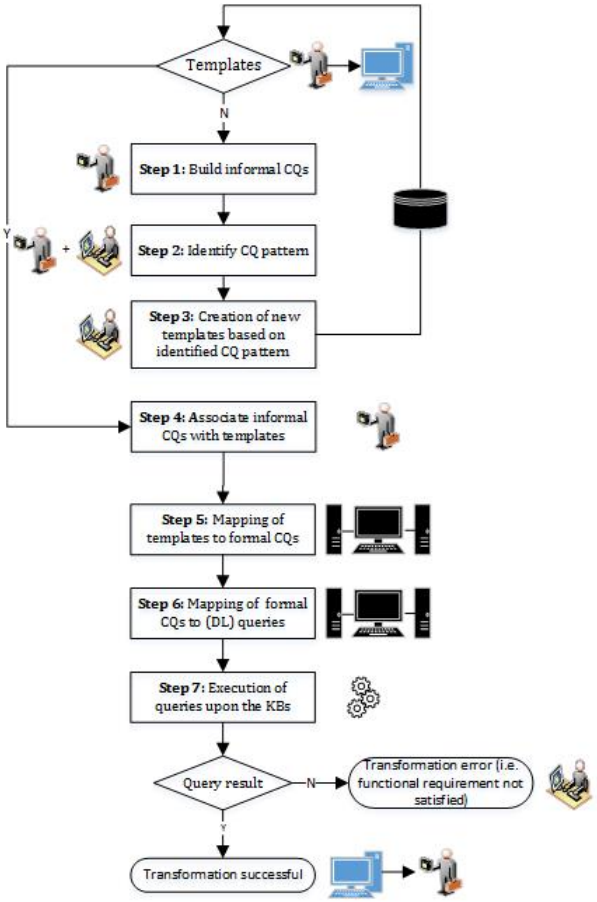


Figure 7.5: Evaluation process for (semi-formal to formal) transformation of legal information using CQs

In section 5.7 of Chapter 5, we proposed a set of mapping schemes for informal to formal legal vocabulary transformation. Wherein, the formal description of objects, properties of objects, and relations among objects provided the language that will be used to express the definitions and constraints in the axioms.

7.3.5 Experiments

To show the process of evaluation, we identify the CQs pertaining to the legal vocabulary defined in section 5.3. Such CQs are identified by a legal practitioner informally in natural language. Secondly, the patterns that match the informal CQs are identified. The patterns are used to define templates, that are presented to an expert in the particular field; his task is then to fill in the variables in templates and define the desired output for each CQ. Sample templates may be grouped into three modules [181]:

(a) ***Selection question*** - a type of question answered with a set of entities or values

(b) ***Binary question*** - a question answered with a boolean value

(c) ***Counting question*** - question answered with the number of items or values returned from relevant selection question

Consult the Table 7.3 for a non-exhaustive list of CQs and relevant patterns extracted from them. From a legal practitioner's perspective, the informal questions pertaining to validation and verification have the same weight and purpose. Therefore, they are presented without any distinction in Table 7.3.

Table 7.3: CQs and patterns

Sample CQs	Pattern
Selection question module	
What is a claim?	Define a X
Define a claim	
What is the language of a claim? **What** are the examples of a claim?	**W** is/are the X of Y? (where **W** ∈ {Where, What, Who})
What is a relation between claim and a patent?	What is the relation between X and Y?
What is the relation between office action and **paragraph 7.33.01**?	
What does patent *include*[3]?	What does X *R*?
Binary question module	
Is the patent application *filed* in **US**? Does the patent application *include* a claim?	Is/are X *relation* Y? Do/Does X *relation* Y?
Counting question module	
What is the minimum number of claims in a patent application?	The **Q** of X in Y? (where ***Q*** ∈ {*number*, *minimum number*, *maximum number*})
How many criteria compose the essential subject matter requirement?	

7.3.6 Formal CQs and Formal Queries

Continuing with our example, we transform the pattern identified in Table 7.3, into formal CQs, which are later represented using existing query languages. Table 7.4, provides some basic formal CQs and its query pattern.

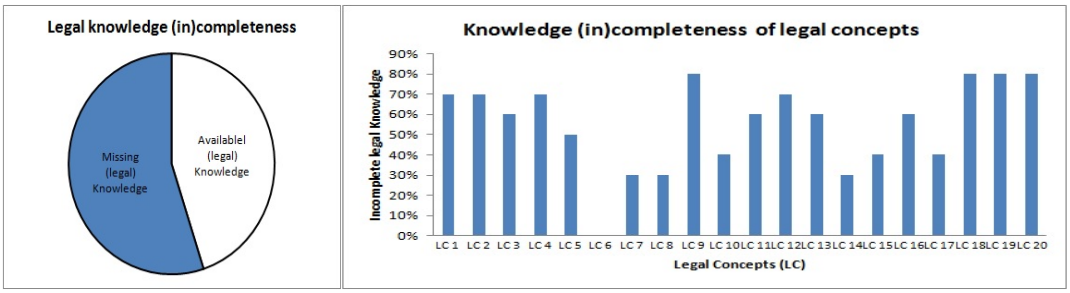
The queries depicted in Table 7.4, may use a simple *de-facto* standard such as SPARQL or a query language which can accomodate higher expressiveness such as SPARQL-DL [159]. The answers to the queries (presented in Table 7.4) are in line with the definitions given in Table 5.3 of section 5.6.

Fig 7.6a, provides an overview on the overall knowledge (in)completeness within the legal vocabulary for our considered example. Fig 7.6b, provides the knowledge (in)completeness within each considered legal concept. E.g. Legal Concept (LC) 3 has around 60 percent of variable/placeholders that needs to be instantiated with (legal) knowledge pertaining to LC 3.

[3]Question/pattern belongs to a non-exhaustive list of CQs pertaining to validation.

Table 7.4: Formal CQs mapped to DL based queries

Define a X	[skos:definition][CE]?	SELECT ?LegalConcept ?definition WHERE { ?LegalConcept a owl:Class ; skos:definition ?definition .}
What is the relation between X and Y?	[OPE][CE1][CE2]?	processQuery("PREFIX : ⟨abc#⟩ \n" + "SELECT ?x ?y WHERE {\n" + PropertyValue(?x,:OPE, ?y)" +"}");
What does X *R*?	[CE1][OPE]=[CE2]?	SELECT {?y WHERE {\n" + PropertyValue(CE1,:OPE, ?y)"+"};
[Is/are][Do/Does] X *relation* Y?	[OPE][CE1][CE2]=Y/N?	ASK {:OPE rdfs:domain :CE1 ; rdfs:range :CE2.}
The Q of X in Y?	[OPE@min/maxCardinality] [CE1][CE2]?	SELECT ?cardinality WHERE { ⟨http://abc⟩ owl:equivalentClass ?c. ?c owl:qualifiedCardinality ?cardinality }



(a) Overall (in)completeness (b) Knowledge (in)completeness of legal concepts

Figure 7.6: (In-)completeness validation

Similarly, Fig 7.7, provides an overarching view on the sensitiveness of legal concepts pertaining to the example considered. From Fig 7.7, it can be seen that any modification to the legal concept *office_action* would require modifications to several other definitional changes to be done to other legal concepts connected to it. The permissible sensitivity pertaining to a legal vocabulary can be varied using the *'Threshold'* parameter, where *'Threshold'* is inversely proportional to sensitivity.

7.4 DL-based legal knowl. expressivity and complexity evaluation

Reasoning on a particular section/paragraph requires additional knowledge i.e. it requires several external domain ontologies (such as, case-laws, *NPS-definitions*, etc...). As seen

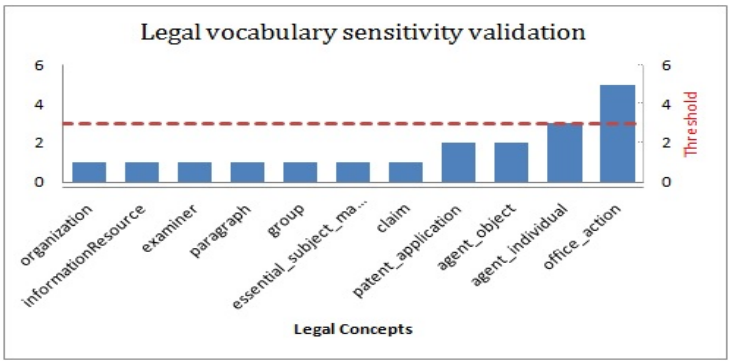


Figure 7.7: Sensitivity Validation

in 6.1.6, we use OntoMaven [135], a collaborative agile ontology development, life cycle management and transitive dependency management technology. Domain ontologies can be in RDFS or OWL, this section presents the expressiveness and complexity of the overall ontology. Based on the flexible SPARQL-DL API integrated in Prova, ontology reasoners that implement OWL API can be easily configured to meet specific requirements

RDFS is built on top of RDF and provides well defined meanings for RDF vocabularies. RDFS defines classes, their properties and relations by using subClassOf, Class, Property, subPropertyOf, Resource, range, domain, etc. However, RDFS also has limitations to express: range restrictions, disjointness of classes, Boolean combinations of classes, cardinality restrictions, and special characteristics of properties.

Reasoning of RDFS data is an entailment process from one (source) RDF graph to another target RDF graph by means of the set of rules that define the semantics of RDFS. Based on the result from [182], the entailment for RDFS is decidable, NP-complete, and in P if the target graph does not contain blank nodes. Moreover, the authors argue that the standard set of entailment rules for RDFS is incomplete, which can be corrected by allowing blank nodes in predicate position.

RDFS enriches the data model presented by RDF and provides support for describing simple ontologies. OWL is another ontology format, but is more expressive than RDFS and provides far larger vocabulary to express the relationships between things. The main facilities that OWL can express over RDFS are: object property relations between classes, constraints on properties, equivalences between classes, properties of properties, and Boolean combinations of classes and constraints. However, it is impossible to compute all interesting logical conclusions from an OWL ontology, and an OWL reasoning could be exponential or even undecidable. To address this issue, OWL 1 and OWL 2 have different sub-languages/profiles with increasing expressiveness.

Decidability is an important factor to be considered when dealing with a logic applied in a (sub/-)languages under consideration. A problem may be defined as *decidable*, if there exists an always terminating algorithm which determines, whether or not a solution exists. Contarary to this, a problem is *undecidable*, if its not *decidable*. A problem is *semi-decidable*, if there exists an algorithm which, in case a solution exists, finds this out in a finite time and

OWL 1 provides three expressive sublanguages, OWL Lite, OWL DL and OWL Full.

The semantics of OWL lite and OWL DL are based on DL and are decidable [4]. Concerning the computational complexity, OWL Lite entailment is known to be complete for EXPTIME, while the entailment for OWL DL is known to be complete for NEXPTIME. OWL Full entailment is known to be undecidable.

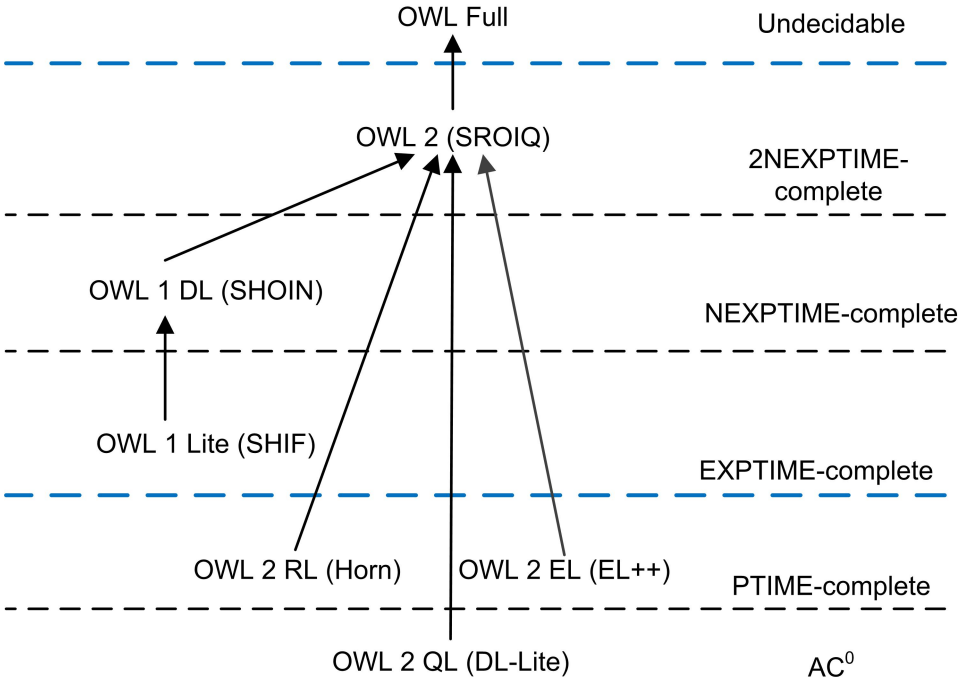


Figure 7.8: Expressiveness and Complexity of OWL Family

OWL 2 is a subsequent compatible revision to its previous versions and W3C Recommendation December 2012. OWL 2 provides an increased expressive power, such as qualified cardinality restrictions, property chain inclusion axioms and reflexive, irreflexive and asymmetric object properties. OWL 2 direct semantics is strongly related to the semantics of DL (i.e. *'SROIQ'*). In OWL 2, the semantics of DL is extended to accommodate datatypes and punning. Syntactic restrictions are applied inorder to obtain the decidability factor in direct semantics. Such restrictions include type separation/punning, No cycles in property chains and No transitive properties in cardinality restrictions. Reasoning complexity of OWL 2 is 2NEXPTIME-complete. A complete list of expressivity and complexity of OWL family can be seen in Figure 7.8

From the mapping scheme proposed in Section 5.7.1, we re-iterate the criteria's of a formal representation format for SLE vocabulary:

- Punning - To allows an entity to be interpreted as two different types of thinga class and an individualdepending on its syntactic context.
  Eg: Individual Concepts in SLE

- Atomic negation - To allow representation of objects which are not members of the class itself.
  Eg: General Concept and Concept relationship in SLE

[4] A DL is decidable if *entailment of axioms* is decidable

- Universal Restriction - To accommodate the quantification aspects of SLE.
  Eg: patent_application *includes* at least 1 claim

- Role Hierarchy - Allows representation of property chains i.e subPropertyOf.
  Eg: *includes* and *is_included_under*

- Inverse Property - To accommodate representation of one property by taking another property and changing its direction.
  Eg: *discloses* and *conceals*

- datatype Property - Allows individuals to be connected to a data value via a datatype property.
  Eg: hasDate

From the above discussed criteria's we see that, OWL 2 representation format is a best fit for our legal vocabulary transformation. Based on the requirements, the expressivity required for such representation falls to $\mathcal{ALCHOIQ}$. Thereby leading to a computation complexity [5] of *'NExpTime-complete'* [183].

The SPARQL-DL API [159] used for reasoning within KR4IPLaw is built on OWL API and is fully aligned with OWL 2. The flexibility of SPARQL-DL API to accommodate wide range of reasoners supporting OWL API 3 makes it a favorable choice. A comparison of reasoners implementing OWL API, capable of aligning with the KR4IPLaw are as shown in Table 7.5

Table 7.5: Comparison of a subset of reasoners that can be aligned with KR4IPLaw

Reasoner	OWL-DL Support	OWL-2 Support	Supported Expressivity	License
Pellet	Yes	Yes	SROIQ(D)	Open Source & Commercial
FaCT++	Yes	Yes	SROIQ(D)	Open Source
HermiT	Yes	Yes	SROIQ+	Open Source
RacerPro	Yes	Yes	SROIQ(D)	Commercial
Chainsaw	Yes	Yes	SROIQ(D)	Open Source

The subset of reasoners considered here for comparison are Pellet [184], FacT++ [185], HermiT [186], RacerPro [187] and Chainsaw [188]. Pellet and HermiT are Java based OWL-DL reasoners that support reasoning on OWL 2 ontologies. RacerPro is an tableau reasoner availabel commercially. Chainsaw is a metareasoner, which computes ontology modules first and then delegates the processing of the modules to an existing OWL 2 DL reasoner.

The current instantiation of the proof of concept system, KR4IPLaw integrates HermiT as an DL reasoner. HermiT is based on hypertableau calculus method and performs better than other reasoners in implementing OWL API on the three standard reasoning tasks: *classification, consistency* and *concept satisfiability* [189].

[5] Complexity of reasoning in Description Logics- http://www.cs.man.ac.uk/~ezolin/dl/

7.5 Legal rule (rule-base) evaluation

In previous Section 7.3, this thesis with the help of experiments proposed and evaluated an approach for legal vocabulary evaluation. The next goal in this direction is to evaluate the thus built and transformed legal rules.

In general, the process of accumulation of knowledge and expertise by domain expert in building a knowledge base (e.g. rule-base) is incremental and intuitive. This often creates conflicts in expertise (when knowledge building is crowd-sourced) thus resulting in a rule-base with structural errors. This brings in the need for evaluating thus built rule-bases for structural errors.

As like in the previous section, we address the two aspects of evaluation - Verification & Validation. Verification of rule-base is concerned with comparing the rule-base against its specifications. This ensures that the transfer of legal rules from its natural language format to machine-understandable form (via semi-formal step) does not violate any constraints. Even-though, rules are used to define constraints, generally, a representation formats may not have any intrinsic constraints, but the inferencing mechanism that reasons on top of this will have constraints. Such constraints are mapped onto the representation formats to ensure that a verified rule-base can be inferenced without causing an error.

Verification of rule-base may be broadly divided into two categories, consistency and completeness. A standard list of errors against which rule-bases are to be checked are as shown below:

(1) Consistency.

- (a) redundant rules
- (b) conflicting rules
- (c) subsumed rules
- (d) circular rules

(2) Completeness.

- (e) dead-end rules
- (f) missing rules
- (g) unreachable rules

A legal rule in its simplest form, as addressed before may be viewed as a production rule with additional modal operator in the consequent of the rule.

$$\text{Elementary Legal Norm} = [\text{Modal Operator}]\ [\text{Production rule}].$$

There have been well known works on verification mechanism for production rules [190] [190] [191] [192]. In this thesis, we build upon these well known verifications methodologies to handle the modal operator during verification process.

Consider:

- h_n= Consequent/ head of a rule,
- $b_{n,1}$, $b_{n,2}$, $\cdots$ $b_{n,m}$= Antecedents/ body of the rule h_n
- $\mathcal{M}$ = Modal Operator for the production rule h_n and
- α_n = legal rule

wherein;

- $\mathcal{M}$ = Obligation or Forbidden or Permission and
- $\alpha_n = \mathcal{M}\ h_n\ :-\ b_{n,1},\ b_{n,2},\ \cdots\ b_{n,m}$

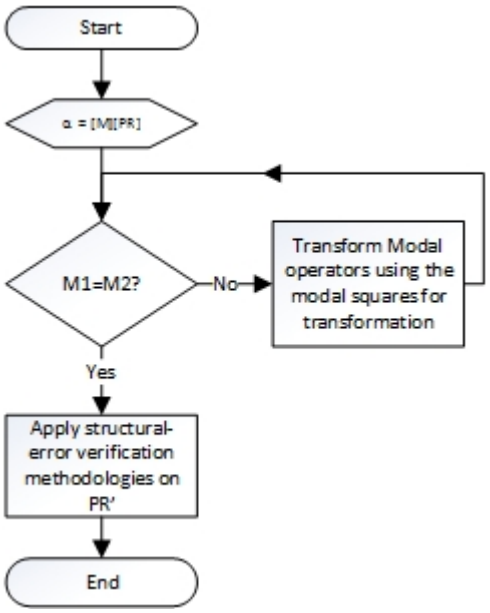


Figure 7.9: Two step process for legal rule verification

We propose a two step process for verification of legal rules for their structural errors. The first step involves *'unification'* of the modal operators in a set of legal rules. If the modal operators cannot be unified, we transform the modal operator using modal transformation functions defined by their respective transformation squares i.e. Deontic transformation square, as defined in section 4.3 or Alethic transformation square as defined in 4.2. With the unification of modal operators, the next step is to use the existing structural error detection methodologies on the thus obtained new production rule part of the legal rule. Figure 7.9, depicts the described two step verification process.

Instantiations of the generalized verification process on different structural errors is addressed in the following subsections.

7.5.1 Redundancy

Redundant rules are those that succeed in same situation and have same results. This also applies for legal rules.

Consider the following example

(a) It is forbidden that **USPTO** *discloses* patent_application if the patent_application *is_filed_with* **USPTO** and *application_status_is* processing.

(b) It is obligatory that it is not the case that the**USPTO** *discloses* patent_application if the patent_application *is_filed_with* **USPTO** and *application_status_is* rejected.

As seen before, the semi-formal legal rules can be transformed into platform-specific prova rules α_1 and α_2

```
1 %
2    α1 => @Forbidden discloses(USPTO, Pat.App)
3                                   :- filed_at(Pat.App, USPTO), app_Status(X).
4    α2 => @Obligatory ¬ discloses(USPTO, Pat.App)
5                                   :- filed_at(Pat.App, USPTO), app_Status(X)
```

Further-on, rules α_1 and α_2 can be viewed as a legal rule having two components. Firstly, the modal operator and second part corresponding to it production rule. Wherein, $\mathcal{M}1$ and $\mathcal{M}2$ are the considered modal operators. The relationship between the modal operators are defined using the modal square (Figurer 4.6 of Chapter 4)

```
7 %
8       α1 => M1   h1(A, B) :- b1,1(B, A), b1,2(X).
9       α2 => M2 ¬ h2(A, B) :- b2,1(B, A), b2,2(X).
10
11      M1 = Forbidden
12      M2 = Obligation
13
14      M2p = M1¬p
```

Applying our two-step verification process with the transformation of the modal operators, we can conclude that legal rules α_1 and α_2 are redundant.

7.5.2 Conflict

Conflicting rules are those which succeed in the same situation with conflicting results. Continuing with our example from the previous subsection 7.5.1. Consider an additional legal rule in its semi-formal representation as shown below:

(c) It is obligatory that **USPTO** *discloses* patent_application if the patent_application *is_filed_with* **USPTO** and *application_status_is* granted.

Which when transformed into a platform-specific language results in the legal rule α_3

```
16 %
17    α3 => @Obligatory discloses(USPTO, Pat.App)
18                      :- filed_at(Pat.App, USPTO), app_Status(X)
19
20    α3 => M3 h3(A, B) :- b3,1(B, A), b3,2(X).
```

Again, applying our two-step verification process, with the transformation of the modal operators, we see that the legal rules α_1 and α_3 are conflicting.

7.5.3 Subsumed

If two rules' consequent are equal and one rules' antecedent consists of the other and some literals, we say that the more restrictive rule is subsumed by the other. I.o.w. a subsumed rule, α is one such that there exists a rule α' replacing it in every situation.

Continuing with our example from before, consider a legal rules α_4

(d) It is obligatory that **USPTO** *discloses* patent_application if *application_status_is* granted.

which results in:

```
%
      α4 => @Obligatory discloses(USPTO, Pat.App) :- app_Status(X)

      α4 => M4 h4(A, B) :- b4,1(X).
```

Thus, we can conclude that legal rule α_3 is subsumed by rule α_4 beacuse α_4 needs only a portion of information required by α_3 to conclude the results.

7.5.4 Circular

Circular errors are closely tied to what is know as *'begging the question'* in legal reasoning. If antecedent of a rule is the consequent of another rule, whose antecedent is the consequent of the first rule, then such rules are said to be in circular. Consider and example closely tied to our previous legal rule α_4.

(e) It is obligatory that *application_status_is* granted if the **USPTO** *discloses* patent_application.

This can be seen as:

```
%
      α5 => @Obligatory app_Status(X) :- discloses(USPTO, Pat.App)

      α5 => M4 h5(X) :- b5,1 (A, B).
```

From the examples above, we see that antecedent of the legal rule α_4 is the consequent of the legal rule α_5 and vice-versa. Thus, the two rules, α_4 and α_5 are deemed to be circular.

Further subsections deal with the next set of errors associated with *'Completeness'* aspect of legal rule verification.

7.5.5 Completeness

Ligeza [193], defines the problem of completeness as *"Does the system react in any possible state providing some/satisfactory/optimal action/conclusions for any input state?*. I.o.w a complete rule-based system is one able to fire at least one rule for any input. There exist few approaches for checking completeness of a rule-base systems such as *'Exhaustive enumeration'*, *'Selected test case validation'* or *FOL-based approach.*

Under a logic based approach, a system is logically complete if and only if the disjunction of the preconditions is always true. i.e. But in a more closed world assumption scenario (as in our approach), we can define completeness as

Definition 1: *A set of legal rules, α_{legal} is specifically complete with respect to situation defined by α's $\mathcal{M}$ i.e. by $\mathcal{P}(\alpha)$.*

$$\text{iff } \alpha \models_{\mathcal{P}(\alpha)} h_1 \vee h_2 \vee h_3 \cdots \vee h_n.$$

Subgoals are created from the rules whose consequent matches the current goal. In order for a rule to be reachable, each subgoal must match a fact whose truth value is provided by the user or by the consequent of a another rule. A rule with unreachable antecedent is known as dead-end rule. Such gaps can be fixed by adding missing rule (or by missing facts).

7.6 LP-based legal knowl. expressivity and complexity evaluation

In Section 5.9, we showed how the transformation of legal rules from their formal representation format to platform specific representation format using the rule language Prova. This section deals with the expressivity and complexity evaluation of the representation format. This thesis first introduces the background theory into standard complexity classes in computational complexity theory, followed by a evaluation in the context of legal rule representation.

7.6.1 Complexity Class Overview

Informally, a Turing machine is a device that reads from and writes on a semi-infinite tape, whose contents may be locally accessed and changed in a computation by a cursor. It is often used as a computational model to define the amount of time and space used by a problem, i.e., the complexity of the problem. In addition to the normal states, there are three additional states: *halt, yes,* and *no*. For an input I, a Turing machine T accepts the input I if T halts in *yes* and rejects the input I if T halts in *no*. The state *halt* is reached when the output of T on I is computed.

A Deterministic Turing Machine (DTM) is the most basic Turing machine, which uses a set of fixed steps to determine its future actions. Compared with a DTM,a Nondeterministic Turing Machine (NDTM) has no fixed steps to determine its future actions, and there would be multiple possible computational paths from a given state. A NDTM solves a problem if there is at least one path that can solve the problem.

The time expended by a DTM T on an input I is defined as the number of steps taken by T on I from the start to halt. If T does not halt on I, the time is considered to be infinite. For a NDTM T, the time expended by T on I is defined as one if T does not accept I; otherwise, it is defined as the minimum over the number of steps in any accepting computation of T. The space required by a DTM T on I is the number of cells visited by the cursor during the computation on I. In the case of a NDTM, the space is defined as one if T does not accept I; otherwise, it is defined as the minimum number of cells visited on the tape over all accepting computations.

Furthermore, a DTM or NDTM T decides a language L such that for all strings x

(a) if $x \in L$ then T accepts x, and
(b) if $x \notin L$ then T rejects x.

The time and space required by T to decide L are defined as follows:

- $TIME(f(n)) = \text{L} \mid \text{L is decided by some DTM in time } O(f(n))$,
- $NTIME(f(n)) = \text{L} \mid \text{L is decided by some NDTM in time } O(f(n))$,
- $SPACE(f(n)) = \text{L} \mid \text{L is decided by some DTM within space } O(f(n))$,
- $NSPACE(f(n)) = \text{L} \mid \text{L is decided by some NDTM within space } O(f(n))$

Here, $f(n)$ is a function from the positive integers to themselves, and $O(f(n))$ denotes an upper bound on the growth rate of the function $f(n)$. Note that the complexity of the function $f(n)$ determines the complexity of solving a computational problem. Some general functions of $f(n)$ are polynomials, exponents or logarithms, and their corresponding complexity classes are denoted as follows [109]:

- $P = \bigcup_{d>0} TIME(n^d)$,
- $NP = \bigcup_{d>0} NTIME(n^d)$,
- $EXPTIME = \bigcup_{d>0} TIME(2n^d)$,
- $NEXPTIME = \bigcup_{d>0} NTIME(2n^d)$,
- $PSPACE = \bigcup_{d>0} SPACE(n^d)$,
- $EXPSPACE = \bigcup_{d>0} SPACE(2n^d)$,
- $L = SPACE(logn)$,
- $NL = NSPACE(logn)$,

In addition, Recursively Enumerable (RE) is the class of decision problems for which a Turing machine can list all the *yes* instances, one by one in a finite amount of time. But the machine might never halt if the answer is *no*. co-RE is the class of decision problems for which a Turing machine can list all the *no* instances in a finite amount of time. The set of recursive languages R is a subset of both RE and $co-RE$. In other words, R is the class of decision problems that are decidable (proved or disproved) in a finite amount of time.

Fig 7.10 depicts the computational difficulty of different complexity classes. The computational difficulty increases from the left to the right along the axis: the simplest complexity class is P (problems solvable in polynomial time). Complexity classes NP (decision problems solvable in polynomial time via a lucky algorithm[6]), EXP (problems solvable in exponential time) and R (problems solvable in a finite amount of time) are more complex than P. All decision problems in such classes are decidable. Moreover, each complexity class can be further subdivided into complete and hard classes. Informally, they are defined as [109]: let C be a complexity class. A problem L is called $C-hard$ if all problems in C can be reduced to L. If L is also a problem in C, then L is called $C-complete$. For examples, a problem L is called $NP-complete$ if all NP problems can be reduced to L, and L is also an NP problem. $C-complete$ problems are also $C-hard$, but not all $C-hard$ problems are $C-complete$. In a sense, $C-complete$ problems are the hardest in C.

[6]A magical algorithm that always makes a right guess among the given set of choices [194]

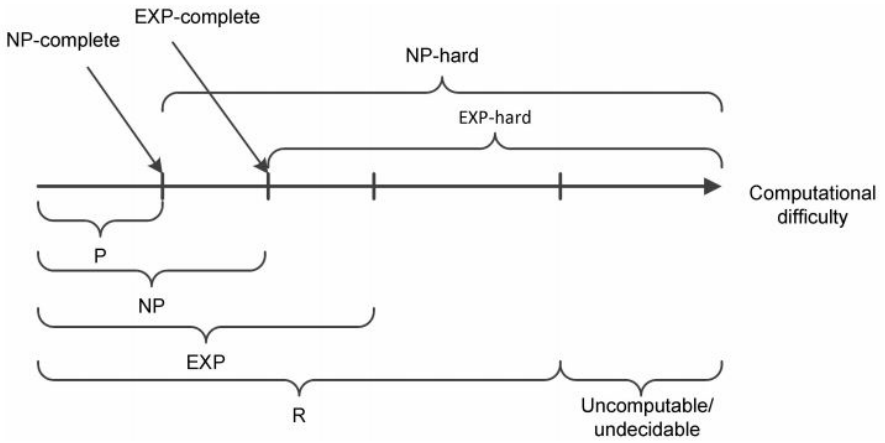


Figure 7.10: Computational Difficulty

7.6.2 LP-based Knowledge Representation Evaluation

Section 4.6 of Chapter 4, provided a brief introduction to different type of logic programs. Further to that, this thesis discusses the complexity of each logic program. Unless specified, complexity refers to combined complexity. Following theorems characterize the complexities of each logic programs, which thereafter is used to define the complexity involved in handling legal rules.

The complexity of plain Datalog programs is characterized by Theorem 1:

Theorem 1: *Datalog is data complete for $\mathcal{P}$ and program complete for EXPTIME [195].*

A finite logic program goes beyond Datalog programs by allowing functions, which provide the ability of handling finite sets of constants, such as encoding lists, trees and many other common data structures. The complexity of definite logic programs is characterized by Theorem 2:

Theorem 2: *Definite logic programming is RE-complete [110] [196].*

A natural decidable fragment of logic programming with functions are *nonrecursive* programs, in which no predicate depends syntactically on itself [197]. Their complexity is characterized by Theorem 3.

Theorem 3: *Nonrecursive logic programming is NEXPTIME-complete [198].*

Definite logic programs in general are not expressive for general knowledge representation, which involves decision (and situational) logic. This is because definite logic programs exclude negation, an important feature for real knowledge representation applications. As described before (see Section 4.6), this leads to another form of logic program, know as normal logic program.

Before introducing the complexities of other logic programs, this thesis, introduces the aspect of *Negation*, from a logic program perspective.

7.6.2.1 Negation

In general, there are two types of negations based on the knowledge assumptions as shown in Figure 7.11.

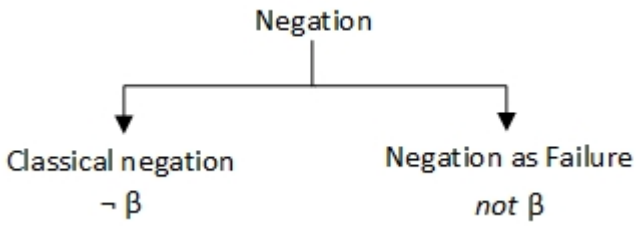


Figure 7.11: Negation

(a) Negation as Failure
(b) Classical Negation

Negation as Failure: Negation as Failure (NaF) is based on the Closed World Assumption (CWA), which assumes something is false if it cannot be proved to be true. In other words, it transforms proving something is false to proving its truth. This is very useful for representing normals and exceptions. The following example shown in Listing 7.1 presents a Prova rule, which describes that an patent application is rejected if it has jurisdiction other than specified. *'patent_application'* is proved to be rejected since the rule engine cannot prove *hasJurisdiction(patent_application)*.

Listing 7.1: NaF implementation in Prova

```
1   filed_in(us)
2   application(patent_application)
3
4   hasJurisdiction(X):- filed_in(X).
5   reject(X):- not(hasJurisdiction(X)), println(["Yes"]) !.
6
7   reject(X):-println(["No"]).
8
9   :- eval(reject(patent_application)).
10
11  % <!-- NaF as implemented in Prova internal function. -->
12
13  not(A) :-   derive(A), !, fail ().
14  not([X| Args]).
```

Classical Negation: The classical negation is based on the Open World Assumption (OWA), which assumes something to be false, if it is explicitly proved to be false. In other words, the negation is in the head of a rule. However, the classical negation might lead to logical conflicts between rules.

The negations in normal logic programs is NaF, and a negative literal succeeds when all attempts to prove the literal fail in a finite amount of time. However, NaF is safe only when the test goal is ground. Revisiting our example from Listing 7.1, *'patent_application'* is rejected since has it has no jurisdiction. But for a goal that queries all rejected application, i.e. `:- eval(reject(X)).`, Prova returns nothing. The reason is that the call

to `not(hasJurisdiction(X))` does not return the applications that are rejected. It fails because there is at least an application (i.e. *'patent_application'*) that has jurisdiction. Therefore, it is important for domain experts to ensure this condition holds when expressing domain-specific knowledge.

Moreover, although negation is needed in many practical knowledge representation application, it is a complex problem in deductive databases and logic programming, since it may result in a computation that does not fail finitely. This problem can be explicitly seen in the following example concerning a patent application from the Listing 7.2.

Listing 7.2: Recursion through negation

```
1   :- eval(patent_application(invention))
2
3   prior_art(X):-
4               not (patent_application(X)),
5               not (mental_thought(X)).
6
7
8   mental_thought(X):- abstract_idea(X).
9
10  abstract_idea(X):- not(prior_art(X)).
11
12  patent_application(X): - not(mental_thought(X)), !
13                      println(["Accepted"])
14
15  patent_application(X): -
16                      println(["Rejected"])
17
18
19
20  not([X| Args ]) :-  derive ([X| Args ]), !, fail ().
21
22  not([X| Args ]).
```

When trying to derive `patent_application(invention)`, it needs to check `not (mental_thought(invention))` and for this it is necessary to check `abstract_idea (invention)` which implies to check `prior_art(invention)`, and then the derivation of `patent_application(invention)` enters an infinite loop. A key feature of this kind of program is that there is a negation wrapped in a recursion. For the purpose of solving this problem, many approaches have been proposed: *Stratified semantics, stable model semantics, well-founded semantics,* etc.

The complexity of Stratified logic program is characterized bt Theorem 4:

Theorem 4: *Stratified Datalog with negation is data complete for $\mathcal{P}$ and program complete for EXPTIME (implicit in [199]).*

However, not all logic programs can be stratified. Stable Model Semantics (SMS) addresses this issue by checking whether a candidate set of atoms is stable or not. Informally, an interpretation $\mathcal{I}$ of a normal logic program $\mathcal{P}$ is a stable model of $\mathcal{P}$ if I is the least Herbrand model of $\mathcal{P}$. The complexity of computing stable models of a logic program is characterized by Theorem 5.

Theorem 5: *Given a propositional normal logic program $\mathcal{P}$, deciding whether $\mathcal{P}$ has stable models is NP-complete [200][201] .*

If a logic program can be stratified, then it has a unique stable model, and its stratified semantics and stable semantics coincide [197]. Legal rules generated in this thesis assume NaF, which is based on CWA. For a logic program that can be stratified, Prova can execute it directly. However, if a logic program cannot be stratified, they have to be handled by rule engines based on SMS and Well Formed Semantics (WFS). Un-Stratisfiable logic programs are out of the scope of this thesis.

Legal knowledge (rule) representation usually involves non-monotonic reasoning, i.e. propositions derived from a knowledge base may be changed by adding or removing clauses. Moreover, representing a legal rule usually involves describing exceptions, which do not conform to general rules. Among the aforementioned logic programs, the most suited for representing legal rules is normal logic program. A normal logic program inherits the expressiveness of propositional and finite logic programs and supports NaF. It also provides a simple and practical formalism for expressing defaults and exceptions, and other forms of non-monotonic reasoning.

Two problems that needs to be addressed when considering normal logic programs are:

(a) Recursion through negation and
(b) Undecidability when using function symbols with no restrictions

As already discussed, the first problem can be resolved by checking if a logic program can ve stratified or not. Prova can execute stratified programs directly. The issue of undecidability when using function symbols with no restrictions is more complex. As aforementioned, using function symbols in logic programs makes reasoning tasks undecidable in general cases. To overcome this issue, many solutions are proposed to impose restrictions on the program syntax to guarantee the decidability of reasoning tasks. A decidable fragment mentioned before is non-recursive logic programs. However, the restriction (i.e.,nonrecursive) is fairly strong and causes a loss of expressive power to express recursion relations. In addition, there are many decidable and expressive fragments of logic programs with function symbols that have been identified. Prova employed in KR4IPLaw itself is undecidable because it has unrestricted functions and external procedural attachments. However, with the use of built-in restriction mechanism such as *guards*, as seen in previous sections, makes it decidable by adding restriction (i.e.,nonrecursions).

7.7 Feature-set Comparison

The section wraps up the chapter on evaluation by providing a comparison of features provided by the proposed KR4IPLaw framework to its existing set of knowledge representation frameworks (or structures) discussed in Chapter 4.

Table 7.6, depicts such feature set comparison, wherein, the rows provide a list of existing legal knowledge representation frameworks and the columns represent the set of features against which the frameworks are evaluated. The evaluation is a simple three valued logic, with

- supported (+),

- not-supported (-), or
- not-applicable

Form the scope of this thesis, we consider only the features which co-relate to legal knowledge representation frameworks' criteria's. This thesis re-emphasizes on few such essential features [7]:

- Isomorphism (a well-defined correspondence between legal knowledge base to their legal source).
- Legal procedures (relation between rules/guidelines).
- Modal behavior (Necessity, Obligatory, Forbidden etc..).
- Life-Cycle (ability to track a norm from its creation to annulment).
- Reasoning (deriving inferences on the represented knowledge).
- etc..

The set of representation frameworks considered for feature-set evaluation may be grouped in-line to the OMG's MDA style of knowledge representation architecture. Such structures may be associated to Computational Independent Model (CIM) layer, which uses controlled natural language based approaches. Platform Independent Model (PIM) layer, which uses XML based mark-up language approaches, Platform Specific Model (PSM) layer, which uses hard-coding approach or a new stack based representation, which utilizes a minimum two of the above proposed approaches (or layers on OMG's MDA) and includes a bi-directional mapping scheme for transitions between the stacks (layers) of knowledge representation.

[7] Much deeper discussion on a long list of feature-set is provided in Section 4.1

Table 7.6: Feature-set Comparison

Legal Rep. Lang.	CIM Layer	PIM Layer	PSM Layer	Multilingual Support	Modal behavior	Isomorphism	Legal - Procedures	Life-Cycle Management	Jurisdiction	Author & Authority	Temporal Behavior	Reasoner Support
EURLex	NA	+	NA	+	−	+	−	+	+	+	+	−
Drafters Language	+	−	−	+	−	−	−	−	+	−	−	+
MetaLex	NA	+	−	+	−	+	−	+	−	+	−	−
EnACT	NA	+	NA	−	−	−	−	−	+	−	+	−
ACE	+	NA	NA	−	+	−	−	−	−	−	−	+
ACE												
Massachusetts LDL	+	NA	NA	−	−	−	+	−	−	+	+	−
SLE	+	NA	NA	−	+	−	−	−	−	−	+	+
Controlled Legal German	+	NA	NA	−	+	−	−	−	−	−	+	+
LegalDocML	NA	+	NA	+	NA	+	−	+	+	+	+	NA
KRIPL	NA	NA	+	−	−	−	+	−	−	−	−	+
LKIF	NA	+	NA	+	+	−	−	+	−	+	+	NA
Processable English	+	NA	NA	−	−	−	−	−	−	−	−	+
RuleML	NA	+	NA	−	−	+	−	NA	NA	+	+	+
LegalRuleML	NA	+	NA	+	+	+	−	+	+	+	+	+
KR4IPLaw	+	+	+	−	+	+	+	+	+	+	+	+

Not Applicable (NA) | Supported (+) | Not Supported (−) [8]
Computational Independent Model (CIM)
Platform Independent Model (PIM)
Platform Specific Model (PSM)
Attempto Controlled English (ACE)
Legal Knowledge Interchange Format (LKIF)
Knowledge Representation for Intellectual Property Law (KR4IPLaw)

[8] Not supported at the time of this evaluation

7.8 Summary

This chapter presented a detailed evaluation of the KR4IPLaw process, framework and system from different perspectives. A system level evaluation which not only considered the knowledge in the system, but also the inferencing mechanism, knowledge acquisition mechanism and on how user intuitive it is was discussed. To start with, we evaluated our system module, KR4IPLaw-LCR used for legal concept recommendations. Later, we proposed a new evaluative (Verification & Validation) process with the help of CQs to enable a person with/without the knowledge of a domain (i.e. Legal / Semantic Web) to evaluate formal transformations.

This chapter also presented the evaluation of knowledge base and its inference mechanism from both LP and DL perspectives. From the DL perspective, the SPARQL-DL query engine integrated provides an expressive DL query language and acts as an interface to every ontology reasoner that supports OWL API. From a LP perspective, the results showed that legal rules can be represented by normal logic programs, which support Negation as Failure (NaF) and are more expressive than propositional logic programs.

Chapter 8

Conclusion and Outlook

Contents

8.1 Conclusion

To this end, this thesis provided a framework for representing legal norms with elementary pragmatics. While the framework proposed can be applied on any existing norms defined within a National Patent System, for a proof-of-concept evaluation, this thesis provided an instance of such application on United States Patent System.

The main contribution of this thesis was in engineering different approaches for legal knowledge representation. The framework defined was loosely coupled with OMG's Model Driven Architecture. The framework proposed a representation of legal norms on different layers of knowledge representation formats. While the thesis proposed a process for such representation, the use of the different knowledge representations can be changed in a subtle way as to accommodate the end-users grasp on knowledge representation technologies. The process proposed, being an open-ended, allowed legal sections from any NPSs as input and the output from such a process could be used as an input to any existing legal argumentation system.

At first, this thesis, based on a general classification structure introduced the foundations to the domain of law (esp. patent), the Semantic Web and their interplay. Later, the notion of pragmatics and its role in legal norm representation were provided. A detailed discussion on the basic concepts of speech theory act and its interconnections to the domain of law were provided. Furthermore, the thesis proposed a set of theoretical guidelines grounded in legal theories from Grice and Marmor on how one may interpret pragmatics when dealing with the task of legal norm representation. Thereby, providing a base for grounding all the definitions defined in this thesis

General requirements for any knowledge representation structure grouped based on representational adequacy, inferential adequacy, representational efficiency criteria's were

discussed. In addition, a set of requirements specific for legal knowledge representation were also discussed. Based on the requirement set, the thesis provided a detailed discussion on existing prior arts. Wherein, first, document annotation technologies and tools were discussed. Later, a set of semi-formal legal representation structures, which were based on controlled natural language approach were addressed. A series of existing legal rule logical formalisms such as Deductive, Inductive, Default, Defeasible Modal and its sub-groups such as Deontic, Alethic and FODAL logic for representing legal rules were presented. Formal legal rule representation languages such as KRIP/L, SWRL, LKIF and LegalRuleML were discussed later. Lastly, a discussion on a subset of logic programming languages, which were used in the domain of legal rule representation was presented.

A conceptual patent information system KR framework involving three interdependent modules was presented. Such a conceptual framework provided an overview on the sub-modules/actors in a legal information system and the flow of information between such them. Based on the conceptual framework, this thesis proposed a generalized process model for modeling legal rules. The thesis proposed the use of decision models to represent the procedural aspects of legal rules. Further, such procedural information were captured in N:M relationship using a semi-formal representation structure. As a semi-formal representation structure, this thesis proposed an enhancement to the existing SBVR technology to fit to the domain of law. Further, moving from the semi-formal representation to a formal representation format i.e. on to a platform independent layer, this chapter proposed a set of mapping rules, first to transform the existing SLE vocabulary into OWL2 ontology and later the legal rules represented in SLE into formal legal rule representation format. As a formal rule representation format, chapter 5 proposed KR4IPLaw-s, an ad-hoc representation format, which encompasses the important aspects of RuleML, LegalDocML and LegalRuleML. Semantic transformation from SLE to KR4IPLaw via LegalRuleML were also proposed. The thesis provided a detailed discussion on the representation of legal rules with elementary pragmatics on a platform specific layer. For such a representation, the thesis proposed the use of Prova, a semantic rule representation language and an inference engine.

The thesis presented a proof-of-concept system, Knowledge Representation for Intellectual Property Law (KR4IPLaw), to support the process, in-turn the framework defined for representing and reasoning elementary legal rules with elementary pragmatics. The system includes modules for representing legal knowledge at different level of granularity. Modules integrated to the system were; Annotation module for legal document annotation, Judgment Miner module for automatizing the process of identifying the most relevant judgment pertaining to a legal section and further transforming them into a formal representation format, Topic Modeling Module, for boosting the miner module. The decision model module extended UMlet, a java based UML tool for creating decision models. As a semi-formal representation module, this thesis introduces a tool, KR4IPLaw-CNL for representing legal rules in a SLE based controlled natural language format. Also introduced were two visualization modules for OWL2 ontology GUI-access (formal transformation of legal vocabulary). Later the thesis introduced, a legal concept recommender module for assisting in filling the missing context information pertaining to a legal vocabulary. We introduced a reasoner module, in which we integrated SPARQL-DL, a OWL query engine in to Prova, a rule engine as a reasoner module to our existing KR4IPLaw system. The

decidability aspect of the Prova was achieved by the use of built-in restrictions as an integral part of an elementary legal rule. Lastly, the Knowledge Based System- KR4IPLaw- was evaluated both on knowledge and system level.

To wrap things up, the proposed framework, KR4IPLaw-f/w, in addition to supporting the functional requirements, also covers the non-functional requirements/necessities required for any legal knowledge representation framework. The *Granularity* aspect was addressed with the notion of disaggregating legal knowledge on different layer (CIM, PIM, PSM), thereby separating their concerns. The aspect of representing *Implicit* knowledge, the thesis proposed a conceptual framework for the interpretation of legal pragmatics. *Interoperability* and *Evolvability* aspects were addressed with the introduction of notions such as generic decision model, isomorphism principles etc.. to representation formats. Aspects of *Modularity* and *Reusability*, were integrated in the core-framework design structure such that the elementary rules represented on the final KR layer could be re-used with their intended pragmatics in a different domain.

During the course of this study, several problems pertaining to legal knowledge representation were identified. Of which, solutions to a subset of there problems were provided. Section 8.2, provides a list of candidate research problems which fell out-of scope of this thesis. Some of such problems are in itself dealt as an separate sub-branch of legal informatics.

8.2 Outlook

While, a concrete solution with a strong legal knowledge representation and reasoning system was the vision of the project, due to time and resource constraints, some out-of scope research problems were not directly dealt with. This section, in line to our knowledge representation requirements presented in chapter 4 re-iterates such problems and thereby provides a conceptual solutions to some of them.

- Semi-formal legal norm representation discussed in this thesis requires building vocabularies as a base for its representation. A more deeper study to generate a requirement set which can be used to define, build and evaluate a well-formed vocabulary needs an independent research in itself. This should include aspects such as semantic clarity- word sense disambiguation-, naturalness etc.. within vocabulary definitions.

- Simplicity for achieving efficiency, usability and maintainability of a knowledge representation format as defined before has been one of the key requirement. While achieving this has been an important aspect, a generic metric for deciding the threshold level in the domain of legal knowledge based systems seems to be question of relevance.

- On the logical layer this thesis presented FODAL - a quantified flavor of the combination of alethic and deontic logics -. Wherein, a *kripke* style possible world semantics was assumed. There exists several paradoxes on the core-assumptions of modal logic. While the thesis addressed certain top level paradoxes such as

Contrary To Duty (CTD), paradoxes dealing with the fundamental assumption of the logics/semantics such as Urmson's Puzzle [202], Ross Paradox [203] or knowers paradox [204] etc.. are beyond the scope of this thesis. Frameworks such as Simple Kangerian Agency Framework [205] or Meinong-Chisholm Reduction [206] provides some directions towards a possible solution.

- End users being a legal practitioner's, a great amount of care was taken into designing technologies suited for the purpose. While the system, KR4IPLaw provides tool support to provide all the necessary interfaces for interaction, a rich set of GUI's with features such as deeper syntax highlighting, error correction, template navigation would add to the user intuitiveness of the existing system.

Bibliography

[1] A. Galton, "Spatial and temporal knowledge representation," *Earth Science Informatics*, vol. 2, no. 3, pp. 169–187, 2009. [Online]. Available: http://dx.doi.org/10.1007/s12145-009-0027-6

[2] L. Guijarro, "Semantic interoperability in egovernment initiatives," *Computer Standards & Interfaces*, vol. 31, no. 1, pp. 174–180, 2009.

[3] A. Toval, A. Olmos, and M. Piattini, "Legal requirements reuse: a critical success factor for requirements quality and personal data protection," in *Requirements Engineering, 2002. Proceedings. IEEE Joint International Conference on*, 2002, pp. 95–103.

[4] H. J. Levesque and R. J. Brachman, "Expressiveness and tractability in knowledge representation and reasoning1," *Computational Intelligence*, vol. 3, no. 1, pp. 78–93, 1987.

[5] H. Takeda, P. Veerkamp, T. Tomiyama, and H. Yoshikawa, "Modeling design processes," *AI Magazine*, vol. 11, no. 4, pp. 37–48, Oct. 1990. [Online]. Available: http://dl.acm.org/citation.cfm?id=95788.95795

[6] V. K. Vaishnavi and W. Kuechler, Jr., *Design Science Research Methods and Patterns: Innovating Information and Communication Technology*. Auerbach Publications, Oct. 2007. [Online]. Available: http://dl.acm.org/citation.cfm?id=1554779

[7] H. P. Grice, *Studies in the Way of Words*. Harvard University Press, 1989.

[8] A. Marmor, "The pragmatics of legal language," *Ratio Juris*, vol. 21, no. 4, pp. 423–452, 2008.

[9] J. Bézivin and O. Gerbé, "Towards a precise definition of the OMG/MDA framework," in *Automated Software Engineering, 2001.(ASE 2001). Proceedings. 16th Annual International Conference on*. IEEE, 2001, pp. 273–280.

[10] WIPO, "Fields of Intellectual Property Protection," in *WIPO Intellectual Property Handbook: Policy, Law and Use*, 2nd ed. Geneva: WIPO, 2008, ch. 2. [Online]. Available: http://www.wipo.int/about-ip/en/iprm/

[11] R. J. Brachman and H. J. Levesque, *Knowledge Representation and Reasoning*, 1st ed., ser. Morgan Kaufmann. Morgan Kaufmann, 2004.

[12] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, vol. 284, no. 5, pp. 34–43, May 2001. [Online]. Available: http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21

[13] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisition*, vol. 5, no. 2, pp. 199–220, Jun. 1993. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1042814383710083

[14] R. Soley, "Model Driven Architecture," 2000.

[15] THE MIDDLEWARE COMPANY, "Model driven development for j2ee utilizing a model driven architecture (mda) approach."

[16] K. Nitta, J. Nagao, and T. Mizutori, "A knowledge representation and inference system for procedural law," *New Generation Computing*, vol. 5, no. 4, pp. 319–359, Mar. 1988. [Online]. Available: http://link.springer.com/10.1007/BF03037414

[17] H. Boley, A. Paschke, and O. Shafiq, "RuleML 1 . 0 : The Overarching Specification of Web Rules." in *RuleML*, 2010, pp. 162–178.

[18] M. Dastani, "2APL: A Practical Agent Programming Language," *Autonomous Agents and Multi-Agent Systems*, vol. 16, no. 3, pp. 214–248, Jun. 2008. [Online]. Available: http://dx.doi.org/10.1007/s10458-008-9036-y

[19] G. P. Zarri, "NKRL, a Knowledge Representation Tool for Encoding the &Lsquo;Meaning&Rsquo; of Complex Narrative Texts," *Nat. Lang. Eng.*, vol. 3, no. 2, pp. 231–253, Sep. 1997. [Online]. Available: http://dx.doi.org/10.1017/S1351324997001794

[20] Y. Tang and R. Meersman, "SDRule Markup Language: Towards Modeling and Interchanging Ontological Commitments for Semantic Decision Making," in *Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches*, A. Giurca, D. Gasevic, and K. Taveter, Eds. IGI Global, May 2009, ch. 5, pp. 99–123. [Online]. Available: http://www.igi-global.com/chapter/sdrule-markup-language/35856/

[21] M. Palmirani, G. Governatori, A. Rotolo, S. Tabet, H. Boley, and A. Paschke, "LegalRuleML: XML-Based Rules and Norms," in *Rule - Based Modeling and Computing on the Semantic Web*, ser. Lecture Notes in Computer Science, F. Olken, M. Palmirani, and D. Sottara, Eds. Springer Berlin Heidelberg, 2011, vol. 7018, pp. 298–312. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-24908-2_30

[22] T. Athan, H. Boley, G. Governatori, P. M, A. Paschke, and A. Wyner, "OASIS LegalRuleML," in *Proceedings of 14th International Conference on Artificial Intelligence and Law (ICAIL 2013)*. ACM, 2013.

[23] J. W. Lloyd, "Practical advtanages of declarative programming." in *GULP-PRODE (1)*, 1994, pp. 18–30.

[24] P. Leith, "Fundamental errors in legal logic programming," *The Computer Journal*, vol. 29, no. 6, pp. 545–552, 1986.

[25] B. Subirana and M. Bain, "Legal programming," *Communications of the ACM*, vol. 49, no. 9, pp. 57–62, 2006.

[26] A. v. d. L. Gardner, "Artificial intelligence approach to legal reasoning," Stanford Univ., CA (USA), Tech. Rep., 1984.

[27] A. P. Hans Weigand, "The Pragmatic Web: Putting Rules in Context," in *RuleML 2012*, 2012.

[28] M. Schoop, A. de Moor, and J. L. Dietz, "The pragmatic web: A manifesto," *Communications of the ACM*, vol. 49, no. 5, pp. 75–76, 2006.

[29] H. Weigand and J. J. Arachchige, "Value network analysis for the pragmatic web: A case of logistic innovation," in *Proceedings of the 6th International Conference on Semantic Systems*, ser. I-SEMANTICS '10. New York, NY, USA: ACM, 2010, pp. 32:1–32:8. [Online]. Available: http://doi.acm.org/10.1145/1839707.1839747

[30] A. Paschke, "Pragmatic web 4.0: Towards an active and interactive semantic media web," 2013, fachtagung Semantische Technologien, Humboldt University of Berlin.

[31] B. Schafer, "Reinach, pragmatic universals and the methodology of comparative law," *Beitrge der Ludwig Wittgenstein Gesellschaft*, vol. 20, pp. 580–86, 1997.

[32] B. Smith, "Towards a history of speech act theory1," *Speech Acts, Meaning, and Intentions: Critical Approaches to the Philosophy of John R. Searle*, p. 29, 1990.

[33] J. F. Crosby, "Reinach's discovery of the social acts," 1983.

[34] D. Kurzon, *It is hereby performed...: Explorations in legal speech acts*. John Benjamins Publishing, 1986.

[35] J. Habermas, *The Theory of Communicative Action, Volume 1: Reason and the Rationalization of Society*. Beacon Press, 3 1985. [Online]. Available: http://amazon.com/o/ASIN/0807015075/

[36] ——, *The Theory of Communicative Action, Volume 2: Lifeworld and System: A Critique of Functionalist Reason*. Beacon Press, 3 1985. [Online]. Available: http://amazon.com/o/ASIN/080701401X/

[37] J. L. Austin, *How to do things with words*, ser. William James Lectures. Oxford University Press, 1962.

[38] J. R. Searle, *Speech acts: An essay in the philosophy of language*. Cambridge university press, 1969, vol. 626.

[39] M. Sinclair, "Law and language: the role of pragmatics in statutory interpretation," *U. Pitt. L. Rev.*, vol. 46, p. 373, 1984.

[40] R. Siltala, *Law, Truth, and Reason: A Treatise on Legal Argumentation*, ser. Law and Philosophy Library. Springer, 2011. [Online]. Available: https://books.google.de/books?id=Qhg3lDPMMOYC

[41] A. Marmor, "What does the law say? semantics and pragmatics in statutory language," *USC Law Legal Studies Paper*, no. 07-9. [Online]. Available: http://ssrn.com/abstract=1009622

[42] K. J. Osenga, "Cooperative patent prosecution: Viewing patents through a pragmatics lens," *Available at SSRN 1575010*, 2010.

[43] G. Tank, "Graver mfg. co. v. linde co." p. 605, 1950.

[44] USSC, "Winans v. denmead," p. 330, 1854.

[45] S. Gostojić, B. Milosavljević, and Z. Konjović, "Ontological model of legal norms for creating and using legislation," *Computer Science and Information Systems*, vol. 10, no. 1, pp. 151–171, 2013.

[46] R. C. Barcan, "A functional calculus of first order based on strict implication," *The journal of symbolic logic*, vol. 11, no. 01, pp. 1–16, 1946.

[47] KSR Intl Co. v. Teleflex Inc, "U.S. 550 U.S. 398," 2007.

[48] J. M. Balkin, "Understanding legal understanding: The legal subject and the problem of legal coherence," *Yale Law Journal*, pp. 105–176, 1993.

[49] National Association of Legal Assistants, "Introduction to the American Legal System," in *NALA Manual for Paralegals and Legal Assistants: A General Skills and Litigation Guide for Today's Professionals*, 6th ed. Delmar Cengage Learning, 2 2014, ch. 1.

[50] European Patent Academy, *Inventors Handbook*.

[51] H. J. Levesque and R. J. Brachman, "Expressiveness and tractability in knowledge representation and reasoning," *Computational intelligence*, vol. 3, no. 1, pp. 78–93, 1987.

[52] F. Baader, "A formal definition for the expressive power of terminological knowledge representation languages," *J. of Logic and Computation*, vol. 6, no. 1, pp. 33–54, 1996.

[53] N. Chomsky, "Three models for the description of language," *Information Theory, IRE Transactions on*, vol. 2, no. 3, pp. 113–124, September 1956.

[54] W. U. Dressler, *Naturalness and Morphological Change*. Blackwell Publishing Ltd, 2008, pp. 461–471. [Online]. Available: http://dx.doi.org/10.1002/9780470756393.ch12

[55] J. Orešnik, *Naturalness in (morpho) syntax: English examples*. Slovenska akademija znanosti in umetnosti, 2004, no. 61.

[56] T. F. Gordon, G. Governatori, and A. Rotolo, "Rules and norms: Requirements for rule interchange languages in the legal domain," in *Proceedings of the 2009 International Symposium on Rule Interchange and Applications*, ser. RuleML '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 282–296. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-04985-9_26

[57] M. L. Markus, "Toward a theory of knowledge reuse: Types of knowledge reuse situations and factors in reuse success," *J. Manage. Inf. Syst.*, vol. 18, no. 1, pp. 57–93, May 2001. [Online]. Available: http://dl.acm.org/citation.cfm?id=1289679.1289683

[58] T. J. Bench-Capon and F. P. Coenen, "Isomorphism and legal knowledge based systems," *Artificial Intelligence and Law*, vol. 1, no. 1, pp. 65–86, 1992.

[59] J. Karpf, "Quality assurance of legal expert systems," *Logica Informatica Diritto*, vol. 1, pp. 411–440, 1989.

[60] T. Bench-Capon, M. Araszkiewicz, K. Ashley, K. Atkinson, F. Bex, F. Borges, D. Bourcier, P. Bourgine, J. Conrad, E. Francesconi, T. Gordon, G. Governatori, J. Leidner, D. Lewis, R. Loui, L. McCarty, H. Prakken, F. Schilder, E. Schweighofer, P. Thompson, A. Tyrrell, B. Verheij, D. Walton, and A. Wyner, "A history of ai and law in 50 papers: 25 years of the international conference on ai and law," *Artificial Intelligence and Law*, vol. 20, no. 3, pp. 215–319, 2012. [Online]. Available: http://dx.doi.org/10.1007/s10506-012-9131-x

[61] J. Karpf, "Inductive modelling in law: Example based expert systems in administrative law," in *Proceedings of the 3rd International Conference on Artificial Intelligence and Law*, ser. ICAIL '91. New York, NY, USA: ACM, 1991, pp. 297–306. [Online]. Available: http://doi.acm.org/10.1145/112646.112684

[62] P. Missier and L. Moreau, "PROV-dm: The PROV data model," W3C, W3C Recommendation, Apr. 2013, http://www.w3.org/TR/2013/REC-prov-dm-20130430/.

[63] G. Governatori, A. Rotolo, and G. Sartor, "Temporalised normative positions in defeasible logic," in *Proceedings of the 10th International Conference on Artificial Intelligence and Law*, ser. ICAIL '05. New York, NY, USA: ACM, 2005, pp. 25–34. [Online]. Available: http://doi.acm.org/10.1145/1165485.1165490

[64] M. Palmirani, G. Governatori, and G. Contissa, "Modelling temporal legal rules," in *Proceedings of the 13th International Conference on Artificial Intelligence and Law*, ser. ICAIL '11. New York, NY, USA: ACM, 2011, pp. 131–135. [Online]. Available: http://doi.acm.org/10.1145/2018358.2018378

[65] M. Andrei, *Philosophy of Law*. Princeton University Press, 2010.

[66] H. Kelsen, "The validity of a norm and its observance or violation," in *General Theory of Norms*. Oxford University Press, mar 1991, pp. 46–49. [Online]. Available: http://dx.doi.org/10.1093/acprof:oso/9780198252177.003.0012

[67] T. Arnold-Moore and J. Clemes, "Connected to the Law: Tasmanian Legislation Using EnAct." *Journal of Information, Law and Technology*, no. 1, 2000. [Online]. Available: http://dblp.uni-trier.de/db/journals/jilt/jilt2000.html#Arnold-MooreC00

[68] EUR-OP. EUR-Lex. Luxemburg.

[69] A. Boer, R. Hoekstra, and R. Winkels, "MetaLex: Legislation in XML," 2002.

[70] M. Palmirani and F. Vitali, "Akoma-Ntoso for Legal Documents," in *Legislative XML for the Semantic Web*, ser. Law, Governance and Technology Series, G. Sartor, M. Palmirani, E. Francesconi, and M. A. Biasiotti, Eds. Springer Netherlands, 2011, vol. 4, pp. 75–100.

[71] "No Title."

[72] N. E. Fuchs and R. Schwitter, "Attempto controlled english (ace)," *arXiv preprint cmp-lg/9603003*, 1996.

[73] R. Power and D. Scott, "Multilingual Authoring Using Feedback Texts," in *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 2*, ser. ACL '98. Stroudsburg, PA, USA: Association for Computational Linguistics, 1998, pp. 1053–1059. [Online]. Available: http://dx.doi.org/10.3115/980691.980742

[74] C. Paris and K. V. Linden, "{DRAFTER}: {A}n {I}nteractive {S}upport {T}ool for {W}riting {M}ultilingual {I}nstructions," {*IEEE Computer*}, *Special Issue on Interactive NLP*, Jul. 1996.

[75] D. E. Sullivan, "Legislative drafting and legal manual," 2003.

[76] OMG, "Semantics of Business Vocabulary and Business Rules (SBVR)," O M G Document, Tech. Rep. November, 2013.

[77] T. Halpin, "A Logical Analysis of Information Systems: Static Aspects of the Data-oriented Perspective." PhD Thesis, University of Queensland, 1989.

[78] T. Kuhn, "Controlled English for Knowledge Representation by," Ph.D. dissertation, 2009.

[79] ——, "A Survey and Classification of Controlled Natural Languages," *Computational Linguistics*, vol. 40, no. 1, pp. 121–170, Mar. 2014. [Online]. Available: http://www.mitpressjournals.org/doi/abs/10.1162/COLI_a_00168

[80] T. Mitamura and E. H. Nyberg, "Controlled English for knowledge-based MT: experience with the kant system," in *Proceedings of TMI-95*, 1995.

[81] S. Boyd, D. Zowghi, and A. Farroukh, "Measuring the expressiveness of a constrained natural language: an empirical study," in *Requirements Engineering, 2005. Proceedings. 13th IEEE International Conference on*, Aug. 2005, pp. 339–349.

[82] J. Pool, "Can controlled languages scale to the Web?" in *International Workshop on Controlled Language Applications 5*, 2006.

[83] A. Wyner, K. Angelov, G. Barzdins, D. Damljanovic, B. Davis, N. Fuchs, S. Hoefler, K. Jones, K. Kaljurand, T. Kuhn, M. Luts, J. Pool, M. Rosner, R. Schwitter, and J. Sowa, "On Controlled Natural Languages: Properties and Prospects," in *Controlled Natural Language*, ser. Lecture Notes in Computer Science, N. Fuchs, Ed. Springer Berlin Heidelberg, 2010, vol. 5972, pp. 281–289. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-14418-9_17

[84] R. Carnap, *Logical Foundations of Probability*. Chicago]University of Chicago Press, 1962.

[85] B. Fitelson, "Inductive logic," *The Philosophy of Science: An Encyclopedia, Oxford: Routledge*, pp. 384–394, 2005.

[86] G. Antoniou, "A Tutorial on Default Logics," *ACM Comput. Surv.*, vol. 31, no. 4, pp. 337–359, Dec. 1999. [Online]. Available: http://doi.acm.org/10.1145/344588.344602

[87] R. Reiter, "Readings in Nonmonotonic Reasoning," M. L. Ginsberg, Ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1987, ch. On Closed, pp. 300–310. [Online]. Available: http://dl.acm.org/citation.cfm?id=42641.42663

[88] G. Brewka, "Reasoning about Priorities in Default Logic," in *AAAI*, B. Hayes-Roth and R. E. Korf, Eds. AAAI Press / The MIT Press, 1994, pp. 940–945.

[89] R. Alur, T. A. Henzinger, and O. Kupferman, "Alternating-time Temporal Logic," *J. ACM*, vol. 49, no. 5, pp. 672–713, Sep. 2002. [Online]. Available: http://doi.acm.org/10.1145/585265.585270

[90] J. Broersen, A. Herzig, and N. Troquard, "From Coalition Logic to {STIT}," *Electronic Notes in Theoretical Computer Science*, vol. 157, no. 4, pp. 23–35, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1571066106003197

[91] N. D. Belnap, *Facing the Future: Agents and Choices in Our Indeterminist World*. Oxford University Press on Demand, 2001.

[92] E. Lorini, "Temporal STIT logic and its application to normative reasoning." *Journal of Applied Non-Classical Logics*, vol. 23, no. 4, pp. 372–399. [Online]. Available: http://dblp.uni-trier.de/db/journals/jancl/jancl23.html#Lorini13

[93] B. Johnston, "Induction of Defeasible Logic Theories in the Legal Domain," 2002.

[94] G. Sartor, *Defeasibility in legal reasoning*. Springer, 1995.

[95] D. Nute, "Defeasible logic, Handbook of logic in artificial intelligence and logic programming (vol. 3): nonmonotonic reasoning and uncertain reasoning," 1994.

[96] P. McNamara, "Deontic Logic," in *The Stanford Encyclopedia of Philosophy*, spring 201 ed., E. N. Zalta, Ed., 2014.

[97] ——, "Kripke-Style Semantics for SDL," in *The Stanford Encyclopedia of Philosophy*, winter 2014 ed., E. N. Zalta, Ed., 2014.

[98] S. Dmitry, "Logical Formalization of Semantic Business Vocabulary and Rules," Master Thesis, Vienna University of Technology, 2010.

[99] W. Carnielli and C. Pizzi, "Multimodal logics," in *Modalities and Multimodalities*, ser. Logic, Epistemology, and the Unity of Science. Springer Netherlands, 2008, vol. 12, pp. 205–239. [Online]. Available: http://dx.doi.org/10.1007/978-1-4020-8590-1_8

[100] P. Blackburn, M. de Rijke, and Y. Venema, *Modal Logic*, ser. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2001, no. 53.

[101] P. McNamara, "Deontic logic," in *The Stanford Encyclopedia of Philosophy*, fall 2010 ed., E. N. Zalta, Ed. Stanford University, 2010.

[102] D. Solomakhin, E. Franconi, and A. Mosca, "Logic-based reasoning support for SBVR," in *Proceedings of the 26th Italian Conference on Computational Logic (CILC2011), Pescara, Italy, 31 August 31-2 September, 2011*, 2011.

[103] M. Fitting and R. L. Mendelsohn, *First-order Modal Logic*. Norwell, MA, USA: Kluwer Academic Publishers, 1999.

[104] Thomas F. Gordon, "The Legal Knowledge Interchange Format (LKIF)," European project for Standardized Transparent Representations in order to Extend LegaL Accessibility Specific Targeted Research or Innovation Project, Tech. Rep., 2008.

[105] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, M. Dean, and Others, "SWRL: A semantic web rule language combining OWL and RuleML," *W3C Member submission*, vol. 21, p. 79, 2004.

[106] S. Bechhofer and I. Horrocks, "Hoolet: An OWL Reasoner with Support for Rules," University of Manchester, Manchester, Tech. Rep.

[107] T. Bench-Capon and T. F. Gordon, "Isomorphism and Argumentation," in *Proceedings of the 12th International Conference on Artificial Intelligence and Law*, ser. ICAIL '09. New York, NY, USA: ACM, 2009, pp. 11–20. [Online]. Available: http://doi.acm.org/10.1145/1568234.1568237

[108] C. Chesñevar, J. McGinnis, S. Modgil, I. Rahwan, C. Reed, G. Simari, M. South, G. Vreeswijk, and S. Willmott, "Towards an Argument Interchange Format," *Knowl. Eng. Rev.*, vol. 21, no. 4, pp. 293–316, Dec. 2006. [Online]. Available: http://dx.doi.org/10.1017/S0269888906001044

[109] E. Dantsin, T. Eiter, G. Gottlob, and A. Voronkov, "Complexity and expressive power of logic programming," *ACM Computing Surveys (CSUR)*, vol. 33, no. 3, pp. 374–425, 2001.

[110] H. Andreka and I. Nemeti, *The generalized completeness of Horn predicate-logic as a programming language.* Department of Artificial Intelligence, University of Edinburgh, 1976.

[111] S. Marek, "Stratified logic programs - knowledge representation," January 2005.

[112] S. Ramakrishna, "First Approaches on Knowledge Representation of Elementary (Patent) Pragmatics," *Joint Proceedings of the 7th International Rule Challenge, the Special Track on Human Language Technology and the 3rd RuleML Doctoral Consortium*, 2013.

[113] S. Ramakrishna and A. Paschke, "Bridging the gap between Legal Practitioners and Knowledge Engineers using semi-formal KR," in *The 8th International Workshop on Value Modeling and Business Ontology, VMBO*, Berlin, 2014.

[114] C. K. Ogden, I. A. Richards, B. Malinowski, and F. G. Crookshank, *The meaning of meaning.* Kegan Paul London, 1923.

[115] K. Elisa and L. Mark H, "Mapping SBVR to OWL2," IBM Research Division, New York, NY, Tech. Rep., 2013.

[116] J. Karpovic and L. Nemuraite, "Transforming SBVR Business Semantics into Web Ontology Language OWL2 : Main Concepts," in *In Proc. 17th International Conference on Information and Software Technologies IT 2011*, 2011, pp. 231–254.

[117] E. Reynares, C. M. A., and M. R. Galli, "An Automatable Approach for SBVR to OWL2 Mappings," in *XVI Ibero-American Conference on Software Engineering (CIbSE 2013)*, Montevideo, Uruguay, 2013.

[118] A. Paschke, "Reaction RuleML 1.0 for Rules, Events and Actions in Semantic Complex Event Processing," in *Rules on the Web. From Theory to Applications*, ser. LNCS. Springer International Publishing, 2014, vol. 8620, pp. 1–21. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-09870-8_1

[119] M. P. et al., "Legalruleml: Xml-based rules and norms," in *Rule-Based Modeling and Computing on the Semantic Web.* Springer, 2011, pp. 298–312.

[120] M. Palmirani, T. Ognibene, and L. Cervone, "Legal rules, text and ontologies over time," in *RULEML*, 2012.

[121] A. Paschke, H. Boley, Z. Zhao, K. Teymourian, and T. Athan, "Reaction RuleML 1.0 : Standardized Semantic Reaction Rules," in *RuleML 2012*, 2012.

[122] H. Boley, A. Paschke, and O. Shafiq, "RuleML 1.0: The Overarching Specification of Web Rules," in *Semantic Web Rules - International Symposium, RuleML 2010, Washington, DC, USA, October 21-23, 2010. RuleML 2010 Proceedings*, ser. LNCS, vol. 6403. Springer, 2010, pp. 162–178.

[123] A. Paschke, "Reaction RuleML 1.0 for Rules, Events and Actions in Semantic Complex Event Processing," in *Rules on the Web. From Theory to Applications*, ser. Lecture Notes in Computer Science, A. Bikakis, P. Fodor, and D. Roman, Eds. Springer International Publishing, 2014, vol. 8620, pp. 1–21. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-09870-8_1

[124] A. Kozlenkov, "Prova Rule Language Version 3.0 User's Guide," *Internet: http://prova. ws/index. html*, 2010.

[125] A. Paschke, "Rules and Logic Programming for the Web," in *Reasoning Web*, ser. Lecture Notes in Computer Science, vol. 6848. Springer, 2011, pp. 326–381.

[126] A. Paschke and H. Boley, "Rule Responder: Rule-Based Agents for the Semantic-Pragmatic Web," *International Journal on Artificial Intelligence Tools*, vol. 20, no. 06, pp. 1043–1081, Dec. 2011. [Online]. Available: http://www.worldscientific.com/doi/abs/10.1142/S0218213011000528

[127] A. Paschke, *Rule based service level agreements: RBSLA; knowledge representation for automated e-contract, SLA and policy management*. Idea Verlag GmbH, 2007.

[128] M. Palmirani, F. Vitali, and L. Cervone, "LIME: The Language Independent Markup Editor." University of Bologna. [Online]. Available: http://lime.cirsfid.unibo.it/

[129] A. Jeff and G. Stephen, "The Minion Search Engine: Indexing, Search, Text Similarity and Tag Gardening," Sun Microsystems, New York, NY, Tech. Rep., 2008.

[130] S. Robertson and H. Zaragoza, *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc, 2009.

[131] M. F. Porter, "Snowball: A language for stemming algorithms," 2001.

[132] D. Newman, A. Asuncion, P. Smyth, and M. Welling, "Distributed algorithms for topic models," *The Journal of Machine Learning Research*, vol. 10, pp. 1801–1828, 2009.

[133] M. Auer, T. Tschurtschenthaler, and S. Biffl, "A Flyweight UML Modelling Tool for Software Development in Heterogeneous Environments," *EUROMICRO Conference*, vol. 0, p. 267, 2003.

[134] M. Tommasi and A. Corallo, "SBEAVER: A Tool for Modeling Business Vocabularies and Business Rules," in *Knowledge-Based Intelligent Information and Engineering Systems SE - 137*, ser. Lecture Notes in Computer Science, B. Gabrys, R. Howlett, and L. Jain, Eds. Springer Berlin Heidelberg, 2006, vol. 4253, pp. 1083–1091. [Online]. Available: http://dx.doi.org/10.1007/11893011_137

[135] A. Paschke, "OntoMaven: Maven-based Ontology Development and Management of Distributed Ontology Repositories," *CoRR*, vol. abs/1309.7, 2013.

[136] ——, "OntoMaven API4KB - A Maven-based API for Knowledge Bases," in *SWAT4LS*, 2013.

[137] P. P. Alessio Bosca, Dario Bonino, "Ontosphere: more than a 3d ontology visualization tool." [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.62.8462

[138] J. Turian, "Using AlchemyAPI for Enterprise-Grade Text Analysis," AlchemyAPI, Tech. Rep., Aug. 2013.

[139] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer, "DBpedia Spotlight: Shedding Light on the Web of Documents," in *Proceedings of the 7th International Conference on Semantic Systems*, ser. I-Semantics '11. New York, NY, USA: ACM, 2011, pp. 1–8.

[140] M. Van Erp, G. Rizzo, and R. Troncy, "Learning with the Web: Spotting Named Entities on the intersection of NERD and Machine Learning," in *Proceedings of the 3rd Workshop on Making Sense of Microposts (# MSM2013)*, 2013.

[141] F. Draicchio, A. Gangemi, V. Presutti, and A. G. Nuzzolese, "FRED: From Natural Language Text to RDF and OWL in One Click," in *The Semantic Web: ESWC 2013 Satellite Events*. Springer, 2013, pp. 263–267.

[142] M. W. Berry and M. Castellanos, "Survey of text mining," *Computing Reviews*, vol. 45, no. 9, p. 548, 2004.

[143] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.

[144] R. Navigli, "Word Sense Disambiguation: A Survey," *ACM Comput. Surv.*, vol. 41, no. 2, pp. 10:1—-10:69, Feb. 2009. [Online]. Available: http://doi.acm.org/10.1145/1459352.1459355

[145] R. Snow, D. Jurafsky, and A. Y. Ng, "Semantic taxonomy induction from heterogenous evidence," in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2006, pp. 801–808.

[146] N. Lagos, F. Segond, S. Castellani, and J. ONeill, "Event extraction for legal case building and reasoning," in *Intelligent Information Processing V*. Springer, 2010, pp. 92–101.

[147] F. Hogenboom, F. Frasincar, U. Kaymak, and F. De Jong, "An overview of event extraction from text," in *Workshop on Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE 2011) at Tenth International Semantic Web Conference (ISWC 2011)*, vol. 779, 2011, pp. 48–57.

[148] B. Coppola, A. Gangemi, A. Gliozzo, D. Picca, and V. Presutti, "Frame detection over the semantic web," in *The Semantic Web: Research and Applications*. Springer, 2009, pp. 126–142.

[149] A. Gangemi, "A Comparison of Knowledge Extraction Tools for the Semantic Web," in *The Semantic Web: Semantics and Big Data*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, vol. 7882, pp. 351–366. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-38288-8_24

[150] H. Kamp and U. Reyle, "A calculus for first order Discourse Representation Structures," *Journal of Logic, Language and Information*, vol. 5, no. 3-4, pp. 297–348, 1996. [Online]. Available: http://dx.doi.org/10.1007/BF00159343

[151] J. Curran, S. Clark, and J. Bos, "Linguistically Motivated Large-Scale NLP with C&C and Boxer," in *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*. Prague, Czech Republic: Association for Computational Linguistics, Jun. 2007, pp. 33–36. [Online]. Available: http://www.aclweb.org/anthology/P07-2009

[152] J. Bos, "Wide-Coverage Semantic Analysis with Boxer," in *Semantics in Text Processing. STEP 2008 Conference Proceedings*, ser. Research in Computational Semantics, J. Bos and R. Delmonte, Eds. College Publications, 2008, pp. 277–286.

[153] A. Gangemi, F. Draicchio, V. Presutti, A. G. Nuzzolese, and D. Reforgiato, "A Machine Reader for the Semantic Web."

[154] I. Bajwa, L. M. B., and Behzad, "SBVR Business Rules Generation from Natural Language Specification," in *AAAI 2011 Spring Symposium AI for Business Agility*, San Francisco, USA, 2011, pp. 2–8.

[155] H. Afreen and I. S. Bajwa, "Generating UML Class Models from SBVR Software Requirements Specifications," in *23rd Benelux Conference on Artificial Intelligence (BNAIC 2011)*, Gent, Belgium, 2011, pp. 23–32.

[156] J. L. Martinez-Fernandez, J. C. Gonzalez, J. Villena, P. Martinez, J. L. Martnez-Fernndez, J. C. Gonzlez, and P. Martnez, "A Preliminary Approach to the Automatic Extraction of Business Rules from Unrestricted Text in the Banking Industry." in *NLDB*, ser. Lecture Notes in Computer Science, E. Kapetanios, V. Sugumaran, and M. Spiliopoulou, Eds., vol. 5039. Springer, 2008, pp. 299–310.

[157] O. Chaparro, J. Aponte, F. Ortega, and A. Marcus, "Towards the Automatic Extraction of Structural Business Rules from Legacy Databases," in *Reverse Engineering (WCRE), 2012 19th Working Conference on*, Oct. 2012, pp. 479–488.

[158] A. Paschke and S. Ramakrishna, "Legal RuleML Tutorial Use Case - LegalRuleML for Legal Reasoning in Patent Law," Seattle, USA, 2013. [Online]. Available: http://www.csw.inf.fu-berlin.de/ruleml2013/presentations/RuleML2013Tutorial_PaschkePatentLaw.pdf

[159] E. Sirin and B. Parsia, "SPARQL-DL: SPARQL Query for OWL-DL," in *In 3rd OWL Experiences and Directions Workshop (OWLED-2007*, 2007.

[160] B. Motik, R. Shearer, and I. Horrocks, "Optimized Reasoning in Description Logics using Hypertableaux," in *Proc. of the 21st Conference on Automated Deduction (CADE-21)*, ser. LNAI, F. Pfenning, Ed., vol. 4603. Bremen, Germany: Springer, Jul. 2007, pp. 67–83.

[161] B. Glimm, I. Horrocks, and B. Motik, "Optimized Description Logic Reasoning via Core Blocking," in *Proc. of the 5th Int. Joint Conf. on Automated Reasoning (IJCAR 2010)*, ser. LNCS, J. Giesl and R. Hähnle, Eds., vol. 6173. Edinburgh, UK: Springer, Jul. 2010, pp. 457–471.

[162] E. Bernstam, J. Herskovic, Y. Aphinyanaphongs, C. Aliferis, M. Sriram, and W. Hersh, "Using citation data to improve retrieval from MEDLINE," *Journal of the American Medical Informatics Association*, vol. 13, no. 1, pp. 96–105, 1 2006.

[163] R. A. Cole, I. Chief, J. Mariani, H. Uszkoreit, A. Zaenen, G. Varile, A. Z. (eds.), A. Zampolli, R. Cole, and V. Zue, "Survey of the State of the Art in Human Language Technology," 1995.

[164] V. Barr, "A quagmire of terminology: Verification and validation, testing, and evaluation." in *FLAIRS Conference*. AAAI Press, 2001. [Online]. Available: http://dblp.uni-trier.de/db/conf/flairs/flairs2001.html#Barr01

[165] M. T. Khan, "Involving domain experts in ontology construction: A template based approach." in *ESWC*, ser. Lecture Notes in Computer Science, vol. 7295. Springer, 2012. [Online]. Available: http://dblp.uni-trier.de/db/conf/esws/eswc2012.html#Khan12

[166] M. Uschold and M. King, "Towards a methodology for building ontologies," in *In Workshop on Basic Ontological Issues in Knowledge Sharing*, 1995.

[167] M. Gruninger and M. S. Fox, "The role of competency questions in enterprise engineering," in *IFIP WG5.7 Workshop on Benchmarking - Theory and Practice*, 1994.

[168] M. Fernndez-Lpez, A. Gmez-Prez, and N. Juristo, "Methontology: from ontological art towards ontological engineering," 1997.

[169] A. Paschke, "Verification, validation and integrity of distributed and interchanged rule based policies and contracts in the semantic web," *arXiv preprint cs/0609119*, 2006.

[170] D. Corrigan et al., "An ontolological treatment of clinical prediction rules implementing the alvarado score," *Studies in Health Technology and Informatics*, vol. 186, 2013. [Online]. Available: http://www.hrbcentreprimarycare.ie/ppt/Fr_11-30_Corrigan.pdf

[171] P. C. B. Fernandes, R. S. S. Guizzardi, and G. Guizzardi, "Using goal modeling to capture competency questions in ontology-based systems." *JIDM*, vol. 2, no. 3, 2011. [Online]. Available: http://dblp.uni-trier.de/db/journals/jidm/jidm2.html#FernandesGG11

[172] M. Copeland et al., "The swo project: A case study for applying agile ontology engineering methods for community driven ontologies." in *ICBO*. CEUR-WS.org, 2012. [Online]. Available: http://dblp.uni-trier.de/db/conf/icbo/icbo2012.html#CopelandBPSM12

[173] P. C. et al., "Opjk and diligent: ontology modeling in a distributed environment." *Artif. Intell. Law*, vol. 15, no. 2, 2007. [Online]. Available: http://dblp.uni-trier.de/db/journals/ail/ail15.html#CasanovasCTVB07

[174] N. Casellas, "Ontology evaluation through usability measures." in *OTM Workshops*, ser. LNCS, vol. 5872. Springer, 2009. [Online]. Available: http://dblp.uni-trier.de/db/conf/otm/otm2009-w.html#Casellas09

[175] A. Preece, "Evaluating verification and validation methods in knowledge engineering," *Micro-Level Knowledge Management*, 2001.

[176] A. Lozano-Tello and A. Gomez-Perez, "ONTOMETRIC: A method to choose the appropriate ontology," *Journal of DataBase Management*, vol. 15, no. 2, Apr-Jun 2004.

[177] J. L. Esposito, "A Framework Relating Questionnaire Design and Evaluation Processes to Sources of Measurement Error," 2003.

[178] D. Vrandecic, "Ontology Evaluation," PhD Thesis, Karlsruher Instituts fur Technologie (KIT), 2010.

[179] S. Lovrencic and M. Cubrilo, "Ontology evaluation -comprising verification and validation," in *CECIIS*, 2008.

[180] S. Tartir, I. Arpinar, and A. Sheth, "Ontological evaluation and validation," in *Theory and Applications of Ontology: Computer Applications*. Springer Netherlands, 2010. [Online]. Available: http://dx.doi.org/10.1007/978-90-481-8847-5_5

[181] Y. Ren et al., "Towards competency question-driven ontology authoring," in *The Semantic Web: Trends and Challenges*, ser. LNCS. Springer, 2014, vol. 8465.

[182] H. J. ter Horst, "Completeness, decidability and complexity of entailment for rdf schema and a semantic extension involving the owl vocabulary," *Web Semant.*, vol. 3, no. 2-3, pp. 79–115, Oct. 2005. [Online]. Available: http://dx.doi.org/10.1016/j.websem.2005.06.001

[183] S. Tobies, "The complexity of reasoning with cardinality restrictions and nominals in expressive description logics," *Journal of Artificial Intelligence Research*, vol. 12, pp. 199–217, 2000.

[184] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical owl-dl reasoner," *Web Semant.*, vol. 5, no. 2, pp. 51–53, Jun. 2007. [Online]. Available: http://dx.doi.org/10.1016/j.websem.2007.03.004

[185] D. Tsarkov and I. Horrocks, "Fact++ description logic reasoner: System description," in *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2006)*, ser. Lecture Notes in Artificial Intelligence, vol. 4130. Springer, 2006, pp. 292–297.

[186] B. Motik, R. Shearer, and I. Horrocks, "Optimized Reasoning in Description Logics using Hypertableaux," in *Proc. of the 21st Conference on Automated Deduction (CADE-21)*, ser. LNAI, F. Pfenning, Ed., vol. 4603. Bremen, Germany: Springer, July 17–20 2007, pp. 67–83.

[187] V. Haarslev, K. Hidde, R. Möller, and M. Wessel, "The racerpro knowledge representation and reasoning system," *Semant. web*, vol. 3, no. 3, pp. 267–277, Jul. 2012. [Online]. Available: http://dl.acm.org/citation.cfm?id=2590200.2590204

[188] D. Tsarkov and I. Palmisano, "Chainsaw: a metareasoner for large ontologies." ser. CEUR Workshop Proceedings, I. Horrocks, M. Yatskevich, and E. Jimnez-Ruiz, Eds., vol. 858. CEUR-WS.org, 2012. [Online]. Available: http://dblp.uni-trier.de/db/conf/ore/ore2012.html#TsarkovP12

[189] R. S. Gonalves, S. Bail, E. Jimnez-Ruiz, N. Matentzoglu, B. Parsia, B. Glimm, and Y. Kazakov, "Owl reasoner evaluation (ore) workshop 2013 results: Short report," in *ORE'13*, 2013, pp. 1–18.

[190] A. Ligeza, *Principles of Verification of Rule-Based Systems*. Springer, 2006.

[191] F. Polat and H. A. Guvenir, "Uvt: A unification-based tool for knowledge base verification," *IEEE Expert*, vol. 8, no. 3, pp. 69–75, 1993.

[192] S. J. Yang, A. S. Lee, W. C. Chu, and H. Yang, "Rule base verification using petri nets," in *Computer Software and Applications Conference, 1998. COMPSAC'98. Proceedings. The Twenty-Second Annual International*. IEEE, 1998, pp. 476–481.

[193] A. Ligeza, *Principles of Verification of Rule-Based Systems*. Springer, 2006.

[194] D. Erik, "Lecture notes on computational complexity," February 2011.

[195] N. Immerman, "Relational queries computable in polynomial time," in *Proceedings of the fourteenth annual ACM symposium on Theory of computing*. ACM, 1982, pp. 147–152.

[196] S.-Å. Tärnlund, "Horn clause computability," *BIT Numerical Mathematics*, vol. 17, no. 2, pp. 215–226, 1977.

[197] E. Dantsin, T. Eiter, G. Gottlob, and A. Voronkov, "Complexity and expressive power of logic programming," *ACM Computing Surveys (CSUR)*, vol. 33, no. 3, pp. 374–425, 2001.

[198] E. Dantsin and A. Voronkov, "Complexity of query answering in logic databases with complex values," in *Logical Foundations of Computer Science*. Springer, 1997, pp. 56–66.

[199] J. Minker, *Foundations of deductive databases and logic programming*. Morgan Kaufmann, 2014.

[200] W. Marek and M. Truszczyński, "Autoepistemic logic," *Journal of the ACM (JACM)*, vol. 38, no. 3, pp. 587–618, 1991.

[201] N. Bidoit and C. Froidevaux, "Negation by default and unstratifiable logic programs," *Theoretical Computer Science*, vol. 78, no. 1, pp. 85–112, 1991.

[202] J. O. Urmson, "Saints and heroes," 1958.

[203] A. Ross, "Imperatives and logic," *Philosophy of Science*, vol. 11, no. 1, pp. 30–46, 1944.

[204] D. Kaplan, R. Montague *et al.*, "A paradox regained." *Notre Dame journal of formal logic*, vol. 1, no. 3, pp. 79–90, 1960.

[205] P. McNamara, "Deontic logic," *Handbook of the History of Logic*, vol. 7, pp. 197–289, 2006.

[206] F. Feldman, "Adjusting utility for justice: A consequentialist reply to the objection from justice," *Philosophy and Phenomenological Research*, pp. 567–585, 1995.

List of Acronyms

DTD Document Type Definition 11

EP Elementary Pragmatics

EPO European Patent Office 10

EU European Union 9

FODAL First Order Deontic Alethic Logic 5

FORMEX Formalized Exchange of Electronic Publications 34

FOL First Order Logic 37

IPL Imperative Programming Language 16

KB Knowledge Base 65

KBS Knowledge Based System 111

KE Knowledge Engineering 4

KR Knowledge Representation 1

KRIP Knowledge Representation and Inference for Procedural Laws 53

KR4IPLaw Knowledge Representation for Intellectual Property Law 7

KR4IPLaw-p KR4IPLaw- Process 7

KR4IPLaw-s KR4IPLaw- Schema 82

KR4IPLaw-f/w KR4IPLaw- Framework

KR4IPLaw-SLE KR4IPLaw- Structured Legal English

LIME Language Independent Markup Editor 100

LKIF Legal Knowledge Interchange Format 55

LegalRuleML Legal Rule Markup Language 5

MDA Model Driven Architecture 13

MPEP Manual for Patent Examination Procedure 10

S-KE Semantic-Knowledge Extraction . . . 104

SLE Structured Legal English . . . 73

SMS Stable Model Semantics . . . 134

SSE SBVR Structured English . . . 5

SPL Substantive Patent Law . . . 10

STIT Sees To It That . . . 44

SWRL Semantic Web Rule Language . . . 54

SemRole Semantic Roles . . . 105

TAX Semantic Taxonomy . . . 105

UML Unified Markup Language . . . 102

USPTO United States Patent and Trademark Office . . . 10

US United States of America . . . 9

URI Uniform Resource Identifier . . . 108

WFS Well Formed Semantics . . . 135

WSD Word Sense Disambiguation . . . 105

WWW World Wide Web . . . 11

W3C World Wide Web Consortium . . . 79

XML eXtensible Markup Language . . . 11

XMLS XML Schema . . . 11

XSD XML Schema Definition . . . 11

List of Figures

Appendix A

XML-sources

Listing A.1: Annotation of first paragraph of § 112

```
<!--  Akoma_Ntoso document
Author: Shashi Ramakrishna & Adrian Paschke
Organisation: Free University of Berlin, Germany
Project: KR4IPLaw
FileDescription: 35 USC 112 First Paragarph
-->

<?xml-model href="schema.xsd" type="application/xml" schematypens="http://purl.oclc.org/dsdl/schematron"?>
<akomaNtoso xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <act contains="originalVersion">...</act>
    <meta>
        <identification source="#LII">
            <FRBRWork>
                <FRBRthis value="/us/codes;us/main"/>
                <FRBRuri value="/us/codes;us/main"/>
                <FRBRdate date="1946" name="creation"/>
                <FRBRauthor href="#congress" as="#author"/>
                <FRBRcountry value="us"/>
                <FRBRlanguage language="eng"/>
            </FRBRWork>
            <FRBRExpression>
                <FRBRthis value="us/codes;us/patentlaw/main#title35/"/>
                <FRBRuri value="us/codes;us/patentlaw/main#title35/"/>
                <FRBRdate value="2014-03-26" name="Generation"/>
                <FRBRauthor href="#LII" as="editor"/>
                <FRBRlanguage language="eng"/>
            </FRBRExpression>
            <FRBRManifestation>
                <FRBRthis value="us/codes;us/patentlaw/main#title35/␣.akn-html"/>
                <FRBRuri value="us/codes;us/patentlaw/main#title35/␣.akn"/>
                <FRBRdate value="2014-03-26" name="Generation"/>
                <FRBRauthor href="#FUB" as="author"/>
            </FRBRManifestation>>
        </identification>
        <lifecycle source="#FUB">
            <eventRef source="#ref1" id="e1" type="generation" date="1946"/>
            <eventRef source="#ref2" id="e2" type="ammendment" date="1952-07-19"/>
            <eventRef source="#ref3" id="e3" type="ammendment" date="1965-07-24"/>
            <eventRef source="#ref4" id="e4" type="enforcement" date="1965-10-24"/>
            <eventRef source="#ref5" id="e5" type="ammendment" date="1975-01-24"/>
            <eventRef source="#ref6" id="e6" type="enforcement" date="1978-01-24"/>
            <eventRef source="#ref7" id="e7" type="ammendment" date="2011-09-16"/>
            <eventRef source="#ref8" id="e8" type="enforcement" date="2012-09-16"/>
        </lifecycle>
        <temporalData source="#FUB">
            <temporalGroup is="#t1">
```

```
47              <temporalInterval refersto="#inforce" start="e2"/>
48              <temporalInterval refersto="#efficacy" start="e2"/>
49          </temporalGroup>
50          <temporalGroup is="#t2">
51              <temporalInterval refersto="#inforce" start="e4"/>
52              <temporalInterval refersto="#efficacy" start="e4"/>
53          </temporalGroup>
54          <temporalGroup is="#t3">
55              <temporalInterval refersto="#inforce" start="e6"/>
56              <temporalInterval refersto="#efficacy" start="e6"/>
57          </temporalGroup>
58          <temporalGroup is="#t4">
59              <temporalInterval refersto="#inforce" start="e8"/>
60              <temporalInterval refersto="#efficacy" start="e8"/>
61          </temporalGroup>
62      </temporalData>
63      <references source="#FUB">
64          <original href="/us/codes;us/eng/patentlaw/#title35" showAs="␣Title␣35,␣US␣
                Code"
65              id="ref1"/>
66          <passiveRef href="/us/codes/1965-10-24/main" showAs="Pub.␣L.␣89  8 3 " id="
                ref4"/>
67          <passiveRef href="/us/codes/1978-01-24/main" showAs="Pub.␣L.␣94  1 3 1 " id="
                ref6"/>
68          <passiveRef href="/us/codes/2012-09-16/main" showAs="Pub.␣L.␣112  2 9 " id="
                ref8"/>
69          <TLCOrganization id="congress" href="/ontology/organizations/congress/"
70              showAs="US␣Congress"/>
71          <TLCOrganization id="LII" href="/ontology/organizations/LII/"
72              showAs="Cornell␣University"/>
73          <TLCRole id="editor" href="/ontology/roles/editor" showAs="Editor"/>
74          <TLCRole id="author" href="/ontology/roles/author" showAs="Author"/>
75          <TLCPerson id="FUB" href="/ontology/person/editors/FUB"
76              showAs="Free␣University␣of␣Berlin"/>
77      </references>
78  <notes source="#FUB">
79      <note id="#n1">
80          <p>
81              Leahy-Smith America Invents Act: First to file policy.
82          </p>
83      </note>
84      </notes>
85  </meta>
86  <preface>
87      <block name="preface">
88          <docTitle id="title">United States Code </docTitle>
89      </block>
90  </preface>
91  <body>
92      <title id="tit35">
93          <num>Title 35</num>
94          <heading>PATENTS</heading>
95          <section id="tit35-112" period="#t4">
96              <num>    112 </num>
97              <heading>Specification</heading>
98          </section>
99          <clause id="112-a">
100             <num>(a)</num>
101             <noteRef href="#n1"/>
102             <heading>In General.</heading>
103             <list id="tit35-sec112-par1">
104                 <content>
105                     <p> - The specification shall contain a written description of
                            the
106                         invention, and of the manner and process of making and using
                                it, in such
107                         full, clear, concise, and exact terms as to enable any person
                                 skilled in
108                         the art to which it pertains, or with which it is most nearly
                                 connected,
```

```
109                             to make and use the same, and shall set forth the best mode
                                    contemplated
110                             by the inventor or joint inventor of carrying out the
                                    invention. - </p>
111                     </content>
112                 </list>
113             </clause>
114         </title>
115     </body>
116 </akomaNtoso>
```

Listing A.2: Annotation of judgement pertaining to first paragraph of § 112

```
1  <!--  Akoma\_Ntoso document
2  Author: Shashi Ramakrishna \& Adrian Paschke
3  Organisation: Free University of Berlin, Germany
4  Project: KR4IPLaw
5  FileDescription: In re Ruschig Fed Cir decision on 35 USC 112 First Paragarph
6  -->
7
8  <?xml-model href="schema.xsd" type="application/xml" schematypens="http://purl.oclc.org/
       dsdl/schematron"?>
9  <akomaNtoso xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
10 <judgement contains="singleVersion">
11 <meta>
12    <identification source="\#SKGF">
13       <FRBRWork>
14          <FRBRthis value="/us/judgement;us/main"/>
15          <FRBRuri value="/us/judgement;us/main"/>
16          <FRBRdate date="1965" name="Hearing"/>
17          <FRBRauthor href="\#rich" as="\#author"/>
18          <FRBRcountry value="us"/>
19          <FRBRlanguage language="eng"/>
20       </FRBRWork>
21       <FRBRExpression>
22          <FRBRthis value="us/judgement;us/patentlaw/main\#title35/"/>
23          <FRBRuri value="us/judgement;us/patentlaw/main\#title35/"/>
24          <FRBRdate value="2003-03-10" name="Generation"/>
25          <FRBRauthor href="\#author1" as="editor"/>
26          <FRBRlanguage language="eng"/>
27       </FRBRExpression>
28       <FRBRManifestation>
29          <FRBRthis value="us/codes;us/patentlaw/main\#title35/␣.akn-html"/>
30          <FRBRuri value="us/codes;us/patentlaw/main\#title35/␣.akn"/>
31          <FRBRdate value="2014-03-26" name="Generation"/>
32          <FRBRauthor href="\#FUB" as="author"/>
33       </FRBRManifestation>>
34    </identification>
35    <publication date="1965" name="Judgement" showAs="Fer␣Cir␣Judgment" number="910"/>
36    <lifecycle source="\#SKGF">
37       <eventRef date="1965" is="e1" source="ro1" type="Hearing"/>
38    </lifecycle>
39    <analysis source="\#SKGF">
40       <judical>
41          <result type="approve"/>
42       </judical>
43    </analysis>
44    <references source="\#SKGF">
45       <original id="ro1" href="ak/judgement/1965/eng@/main" showAS="Original"/>
46       <TLCOrganisation id="\#court1" href="ak/.../courts" showAS="Feredal␣Circuit"/>
47       <TLCOrganisation id="\#SKGF" href="ak/..." showAS="Sterne␣kessler␣Goldstein␣Fox"/>
48       <TLCPerson id="jud01" href="/ontology/person/ak.rich" showAs="Judge␣Rich"/>
49       <TLCPerson id="jud02" href="/ontology/person/ak.almond" showAs="Judge␣Almond"/>
50       <TLCRole id="Author" href="/ontology/role/Author" showAs="Author␣of␣Document"/>
51       <TLCRole id="Editor" href="/ontology/role/Editor" showAs="Editor␣of␣Document"/>
52       <TLCRole id="Appellant" href="/ontology/role/Appellant" showAs="Appellant"/>
53       <TLCRole id="Respondent" href="/ontology/role/Respondent" showAs="Respondent"/>
54    </references>
55    <notes source="\#SKGF">
```

```
56       <note id="\#n1">
57          <p>
58             - Judge Rich concurred, believing that the specific claim amendment under
                   review was subject to a new matter rejection, but he disagreed with the
                   majority s application of the written description requirement to non-
                   complex cases -
59          </p>
60       </note>
61       <note id="\#n2">
62          <p>
63             - the CCPA clearly articulated a written description requirement, distinct
                   from the enablement requirement, with respect to the Patent Act -
64          </p>
65       </note>
66    </notes>
67 </meta>
68 <header>
69    <p class="judge">
70       <judge id="jud01" refersTo="\#rich">
71          Judge Rich
72       </judge>
73       <judge id="jud02" refersTo="\#Almond">
74          Judge Almond
75       </judge>
76    </p>
77    <courtType id="\#court1" refersTo="\#FerCir"/>
78    <p class="parties">
79       <party id="p1" refersTo="\#ruschig" as="\#Appellant"/>
80       <party id="p1" refersTo="\#FedCir" as="\#Respondent"/>
81    </p>
82
83    <p class="judgementNumber">
84       <span class="preface">
85          <docketNumber>
86             379 F.2d 990
87          </docketNumber>
88       </span>
89    <neutralCitation>
90       In re Ruschig, 379 F.2d 900, 15 USPQ 118 (CCPA 1967)
91    </neutralCitation>
92    </p>
93    <block name="parties" class="partiesContainer" id="parties1">
94       <party id="p1" refersTo="\#ruschig" as="\#Appellant">Ruschig</party>
95       <party id="p1" refersTo="\#FedCir" as="\#Respondent">Federal Circuit</party>
96    </block>
97    <ref id="ref01" href="ak//us/codes/1965/main">
98       <title id="tit35">
99          <num>Title 35</num>
100         <heading>PATENTS</heading>
101         <section id="tit35-112" period="\#t4">
102         <num>
103               112
104         </num>
105         <heading>Specification</heading>
106      </section>
107         <clause id="112-a">
108            <num>(a)</num>
109            <noteRef href="\#n1"/>
110            <heading>Enablement</heading>
111         </clause>
112      </title>
113   </ref>
114   <summary>
115      <noteRef href="\#n1"/>
116      <noteRef href="\#n2"/>
117         <p>
118            - While we have no doubt a person . . . would be enabled by the
                   specification to make it, this is beside the point for the question is
                   not whether he would be so enabled but whether the specification
                   discloses the compound to him, specifically, as
```

```
119             something appellants actually invented     -
120          </p>
121    </summary>
122    </header>
123    <conclusion>
124       <p class="signature">
125          <judge id="jud01" refersTo="\#rich">Judge Rich</judge><eol/>
126          <judge id="jud02" refersTo="\#Almond">Judge Almond</judge><eol/>
127       <span class="signature">
128    CHIEF JUSTICE
129       </span>
130       </p>
131    </conclusion>
132    </judgement>
133 </akomaNtoso>
```

Appendix B

Decision Model

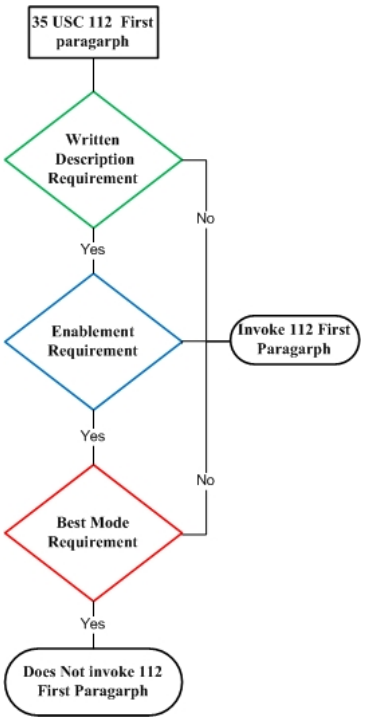


Figure B.1: Decision model for legal section 112, 1^{st} paragraph

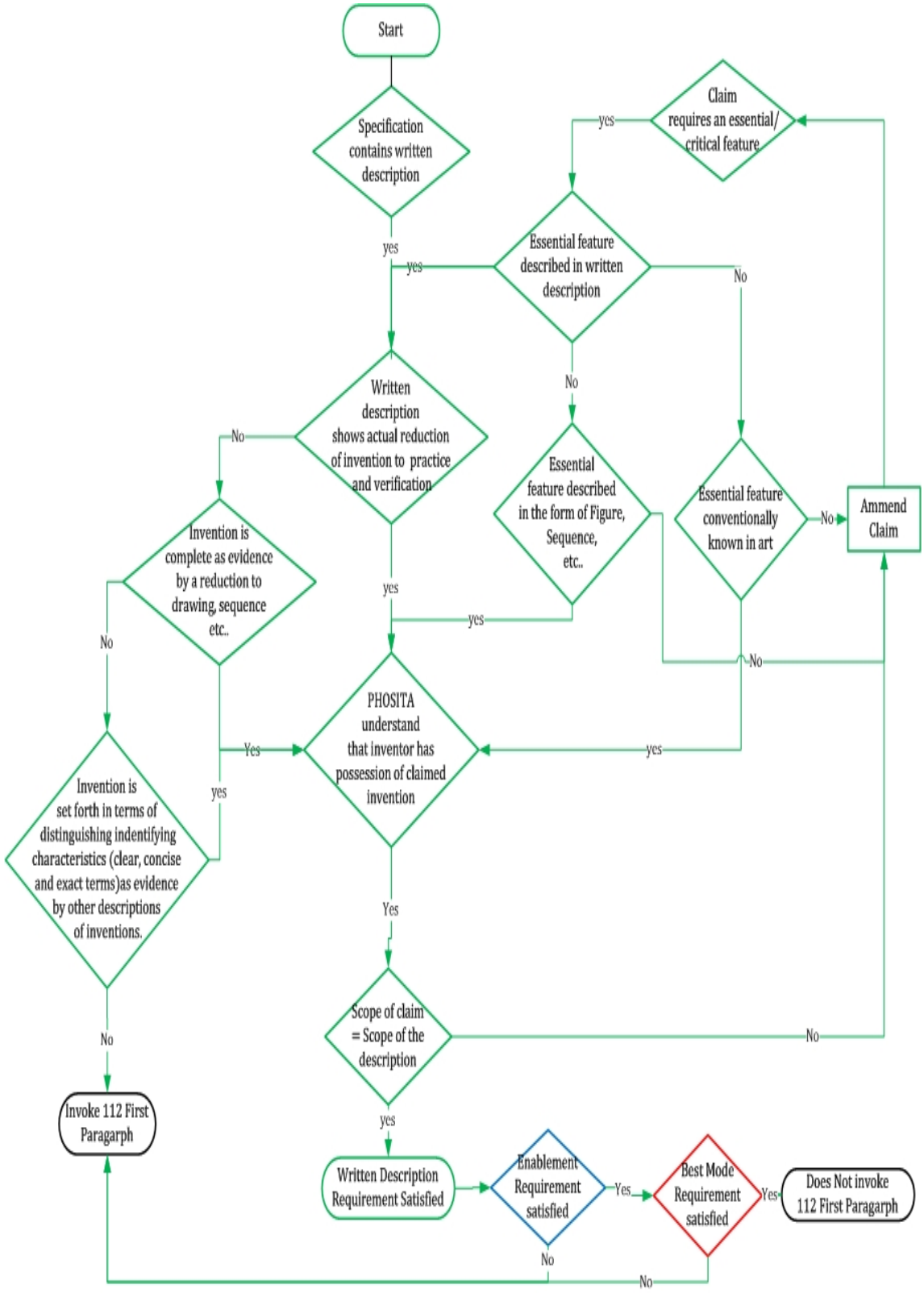


Figure B.2: Decision model for legal section 112, 1^{st} paragraph - Written Description Requirement

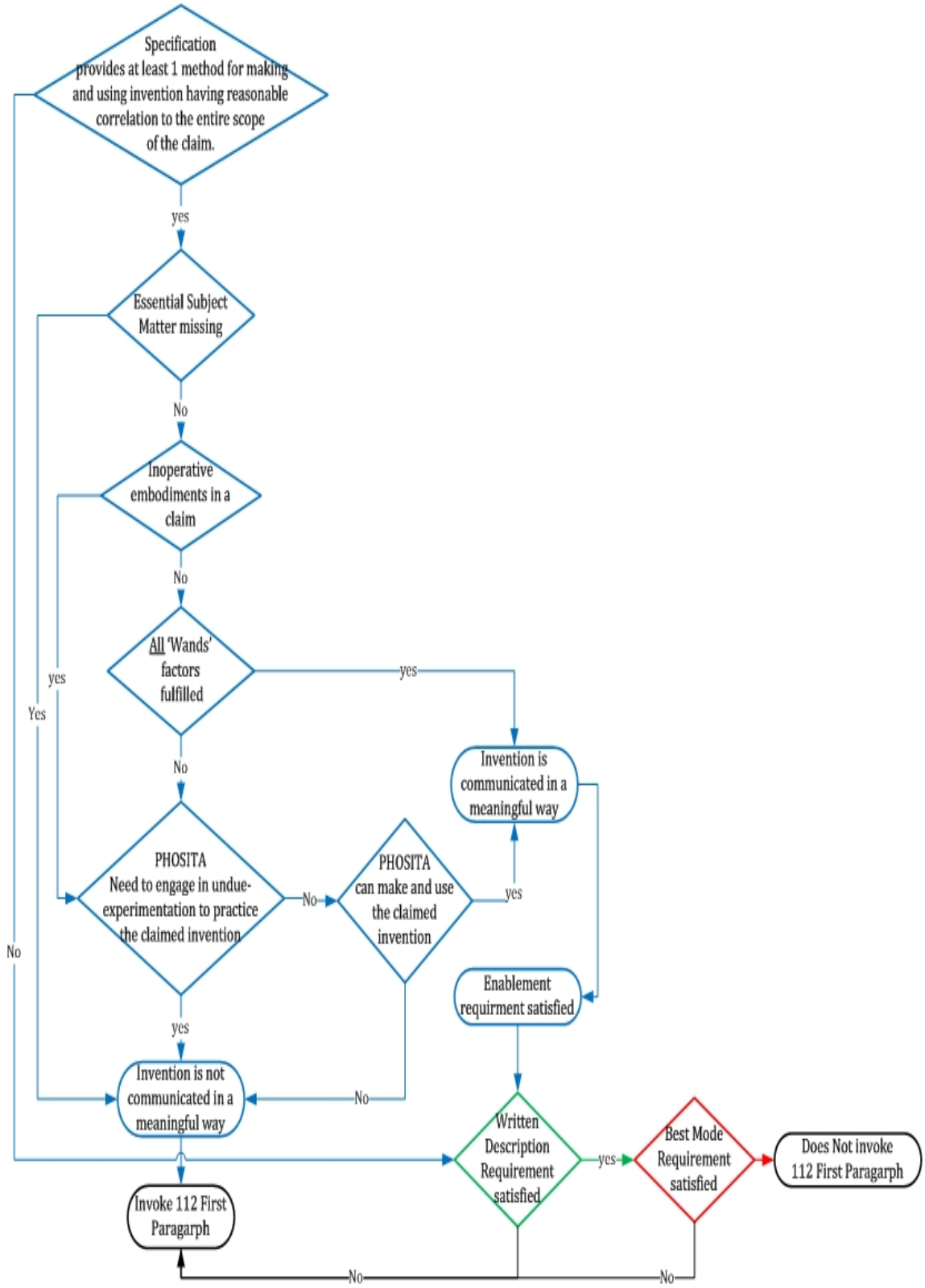


Figure B.3: Decision model for legal section 112, 1^{st} paragraph - Enablement Requirement

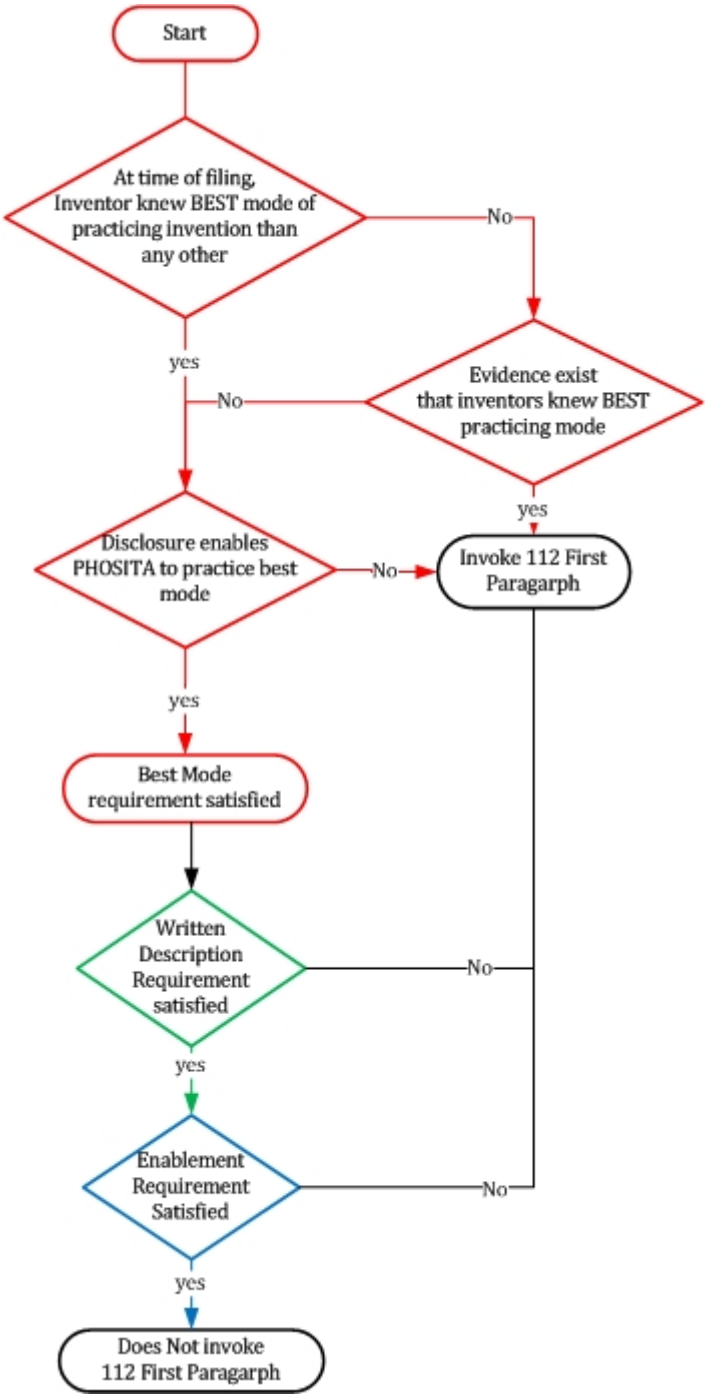


Figure B.4: Decision model for legal section 112, 1^{st} paragraph - Best Mode Requirement

Appendix C

KR4IPLaw- abstract schema: Additional Information

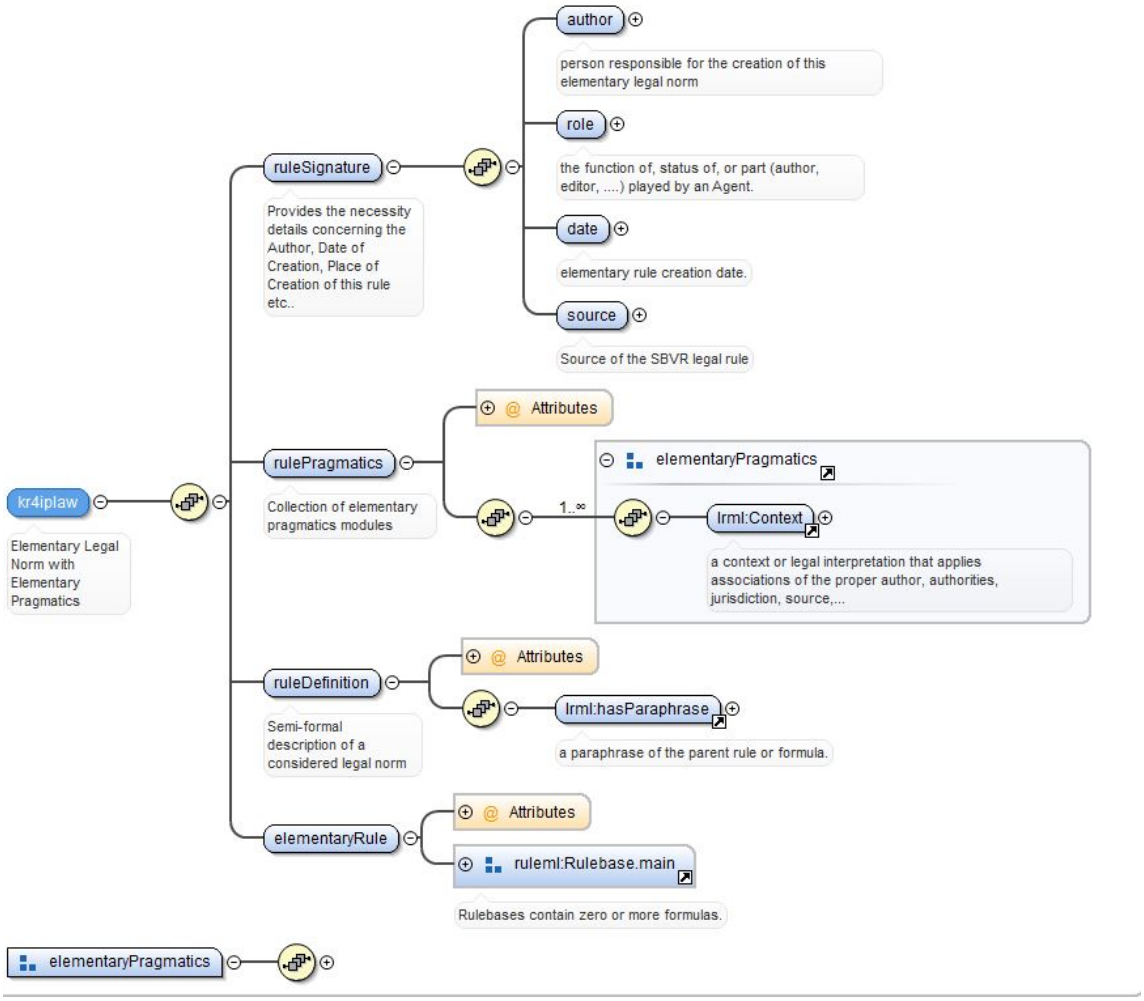


Figure C.1: KR4IPLaw Schema Structure

Appendix D

RuleML and LegalRuleML Glossary

(a) **General Concepts**

- **Actor:** an Agent or a Figure.
- **Deontic Specification:** an Obligation, Permission, Prohibition, Right, ...
- **Internal Identifier:** a local unique identifier of a node in a LegalRuleML document.
- **Isomorphism:** a relationship between a set of Legal Rules with a set of Legal Sources such that the origin of the Legal Rules is tied to the Legal Sources.
- **Legal Norm:** a binding directive from a Legal Authority to addressees (i.e. Bearers or Auxiliary Parties).
- **Legal Rule:** a formal representation of a Legal Norm.
- **LegalRuleML Specification:** an XML schema, Relax NG schema, metamodel, glossary, license, or any other technical normative specification that is an approved outcome of this OASIS TC.
- **Legal Statement:** a LegalRuleML expression of a Legal Rule or a part of a Legal Rule.
- **Legal Status:** a standing that can apply to a Legal Norm at a Time, e.g., "is applicable", "is in force", "has efficacy", "is valid".
- **Status Development:** a kind of event (e.g., start, end) that changes the Legal Status of a Legal Norm, e.g. making a Legal Norm in force.

(b) **Node Elements (normative)**

- **Agent:** an entity that acts or has the capability to act.
- **Alternatives:** a mutually exclusive collection where every member is a LegalRuleML rendering of one or more Legal Norms.
- **Association:** a partial description of the extension of some relations where each non-target entity is paired with every target entity.
- **AuxiliaryParty:** a role in a Deontic Specification to which the Deontic Specification is related.
- **Bearer:** a role in a Deontic Specification to which the Deontic Specification is primarily directed.
- **Compliance:** an indication that an Obligation has been fulfilled or a Prohibition has not been violated.

- **ConstitutiveStatement:** a Legal Statement that defines concepts and does not prescribe behaviours.
- **DefeasibleStrength:** an indication that, in the absence of information to the contrary and where the antecedent of a Legal Rule holds, the conclusion of the Legal Rule holds.
- **Defeater:** an indication that, in the absence of information to the contrary and where the antecedent of a Legal Rule holds, the opposite of the conclusion of the Legal Rule does not hold.
- **FactualStatement:** an expression of fact.
- **Jurisdiction:** a geographic area or subject-matter over which an Authority applies its legal power.
- **LegalRuleML:** a formal representation of one or more LegalSources using the LegalRuleML Specifications.
- **LegalSource:** a source of one or more Legal Norms formulated in any format and endorsed by an Authority.
- **Obligation:** a Deontic Specification for a state, an act, or a course of action to which a Bearer is legally bound, and which, if it is not achieved or performed, results in a Violation.
- **Override:** an indication that a Legal Rule takes precedence over another Legal Rule. The ordered pair of Legal Rules is an instance in a defeasible priority relation.
- **OverrideStatement:** a Legal Statement of an Override.
- **PenaltyStatement:** a Legal Statement of a sanction (e.g. a punishment or a correction).
- **Reparation:** an indication that a PenaltyStatement is linked with a PrescriptiveStatement. It indicates that a sanction may apply where the PrescriptiveStatement entails a Deontic Specification and when there is a Violation of the Deontic Specification.
- **ReparationStatement:** a Legal Statement of a Reparation.
- **Strength:** the quality of a Legal Rule to resist or not to resist a rebuttal.
- **SuborderList:** a Deontic Specification for a sequence of Deontic Specifications, i.e., Obligations, Prohibitions, Permissions, Rights and/or Suborder Lists. When a SuborderList holds, a Deontic Specification in the SuborderList holds if all Deontic Specifications that precede it in the SuborderList have been violated.
- **TemporalCharacteristic:** a pair of Time with a qualification, which consists of a Legal Status and a Status Development, such that the qualification holds at the Time.
- **Violation:** an indication that an Obligation or Prohibition has been violated.

(c) **RuleML Node Elements (normative)**

- **ruleml:Rule** a RuleML Rule encoding a Constitutive Statement or a Prescriptive Statement.
- **ruleml:Time** a neutral temporal entity.

(d) **Edge elements (normative)**

- **appliesAlternatives:** a collection of Alternatives applied by the Context.

- **appliesAssociations:** a collection of Associations applied by the Context.
- **appliesJurisdiction:** a Jurisdiction applied by the Context or Association.
- **appliesStrength:** a Strength applied by the Context or Association.
- **appliesPenalty:** the PenaltyStatement that is linked to a LegalRule in a Reparation.
- **forExpression:** a LegalRuleML expression for which the Role is responsible (e.g., the expression was created or endorsed by the role).
- **forStatus:** the Legal Status of the qualification in a TemporalCharacteristic.
- **fromLegalSources:** the LegalSources from which the Alternatives are derived.
- **hasAlternative:** an Alternative in the collection.
- **hasFigure:** a Figure in the collection.
- **hasReference:** a Reference in the collection.
- **hasStatement:** a Legal Statement in the collection.
- **hasParaphrase:** a Paraphrase of the parent Node Element (e.g. a Legal Rule)..
- **hasQualification:** a qualification (e.g. an Override) of the Statements.
- **hasMemberType:** the type or class of members of the collection.
- **hasType:** the type or class of the parent Node Element.

Appendix E

Source Code

The source code for the system, KR4IPLaw, has been hosted as a private repository[1] on GitHub - a web-based Git repository hosting service-.

Table E.1: Source Code Details

Project:	Knowledge Representation for Intellectual Property Laws
Hosting Service Provider:	GitHub
Repository Name:	KR4IPLaw
Repository URL:	https://github.com/shashi792/KR4IPLaw
Repository access:	private
Authors:	(a) Shashishekar Ramakrishna (b) Prof. Adrian Paschke
Affiliation	AG Corporate Semantic Web, Freie Universität Berlin, Berlin, Germany

[1] Untethered access to the repository has been provided to the reviewers at the time of evaluation.

Appendix F

Zusammenfassung

Trotz der "offensichtlichen Eindeutigkeit" einer rechtlichen regelung, kann ihre anwendung ein urteil nach sich ziehen, das nicht unbedingt konform mit der semantischen ebene des gesetzes ist. Um bei der wissensmodellierung die mehrdeutigkeiten einer juristischen sprache zu minimieren, wird eine elementare repräsentation, die sprache vereinfacht, benötigt. In dieser arbeit wird ein wissensmodellierungsprozess vorgestellt, der auf einem innovativen wissensmodellierungsframework fr patentinformationen, KR4IPLaw (Knowledge Representation for Intellectual Property Law), basiert. Dieses System ermöglicht es, die in einer juristischen Sprache eingebetteten Aspekte zu trennen und damit zu vereinfachen. Zusätzlich wird eine machbarkeitsstudie mitsamt der evaluierung vorgestellt.

Appendix G

Curriculum vitae

"For reasons of data protection, the curriculum vitae is not published in the electronic version of this thesis."