Legal Interpretations in LegalRuleML

Tara Athan[1], Guido Governatori[2], Monica Palmirani[3],
Adrian Paschke[4] and Adam Wyner[5]

[1] Athan Services
[2] NICTA Queensland, Australia*
[3] CIRSFID, University of Bologna, Italy
[4] Corporate Semantic Web, Freie Universität Berlin, Germany
[5] University of Aberdeen, United Kingdom

Abstract. Legislative documents are by their own nature subject to interpretation, and interpretations of one document can diverge. In this paper we discuss the mechanism proposed by LegalRuleML to capture alternative interpretations or renderings of a legal source. LegalRuleML allows for mutually incompatible renderings (or interpretations) of a legal source to coexist in the same LegalRuleML document, and provides facilities to identify the interpretations and to select them. The mechanism is illustrated with an example form Italian Jurisprudence.

1 Introduction

A common trait of legal reasoning and practice is that it is often possible to have multiple interpretations of one and the same textual provision, where there is no true interpretation. Such alternative interpretations might be mutually incompatible. This is often the case in legal disputes where the parties involved put forward their interpretations and where the judge has to select one of them or propose another interpretation.

In this paper we report on the efforts of the OASIS LegalRuleML Technical Committee to capture the phenomenon of multiple interpretations in the LegalRuleML standard [21, 1]. The key intuition is that an interpretation is modelled by a set of LegalRuleML statements (e.g., rules) and a norm or textual provision can be modelled by several alternatives where each alternative has enough metadata to determine its context and provenance. The paper outlines LegalRuleML components and illustrates them with examples, e.g. a real life case from Italian Jurisprudence where the topic of discussion of the case was on different interpretations of a textual provision.

The work on interpretation in LegalRuleML is set in the more general context of interpretation in Linguistics in general and Forensic Linguistics in particular. In Linguistics, issues about interpretation have long been of central concern (from [9] to [18]), where the need for interpretation arises given that the meanings (broadly construed) of "linguistic signs", e.g. words, sentences, and discourses, can vary depending on participants, context, purpose, and other parameters. Interpretation is, then, giving the meaning of the linguistic signs for a given set of parameters. While the relationship between signs

* NICTA is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program.

and meaning is arbitrary in principle [9], most contemporary linguistic research has attempted to identify principles and constraints around interpretation in order to account for a range of consistent, widespread, and observable linguistic patterns. After all, what is truly arbitrary can only be catalogued and not made the object of deeper scientific scrutiny. Research efforst have focussed on syntactic ambiguity, reference, vagueness, semantic scope, and other phenomena. It is worth noting that high performance, statistically based, contemporary parsers (e.g. Stanford's Dependency Parser [8]) or a parser with associated semantic representation, (e.g. C&C/Boxer [7]) do not exclude alternative parses or semantic representations.

In Forensic Linguistics, such considerations are applied to legal texts [22, 24, 4], though with legal specific considerations. Laws prescribe behaviour, so knowing the interpretation of a law in a context matters in terms of guiding conduct. Laws can be challenged, thus opening presentation of alternative interpretations. Laws are composed for social consumption, leading to issues bearing on who composed them, for what purpose, to apply to what other parties, and so on. In addition, laws and circumstances change over time, requiring active maintenance of interpretation. Finally, the practice of law over time has developed its own catalogue of *hermeneutical principles*, a range of techniques to interpret the law, such as catalogued and discussed in [23].

LegalRuleML endeavours not to *account* for how different interpretations arise, but to provide a mechanism to record and represent them. The main novelty of this paper is the presentation of a formal representation of legal interpretation. In Section 2 some of the relevant background literature is reviewed. The running example is presented in Section 3. The case study is formalised in LegalRuleML in Section 4. The paper concludes with Section 5.

2 Related Work and OASIS LegalRuleML

Norms, guidelines and rules are found in a variety of legal texts. As text, the semantic content (including meta-data) is difficult to exchange between parties or to otherwise process or reuse by automated applications. Yet it is essential to eGoverment and eCommerce services that such content has a machine-readable form such that applications can be deployed. The LegalRuleML TC, which was set up inside of OASIS (`www.oasis-open.org`), aims to produce a rule interchange language for the legal domain. Using the representation, developers can provide applications to process legal contents for data interchange, comparison, evaluation, and reasoning.

Over the last twenty years, the Artificial Intelligence (AI) and Law communities have converged on modelling legal norms and guidelines using logic and other formal techniques. With existing methods, a Legal Knowledge Engineer analyses the text, scopes the analysis, extracts the norms and guidelines, applies models and a theory within a logical framework, and finally represents the norms using a particular formalism. In the last decade, several Legal XML standards have been proposed to describe legal texts [19] with XML-based rules (RuleML, SWRL, RIF, LKIF, etc.) [13, 12]. At the same time, the Semantic Web, in particular Legal Ontology research combined with semantic norm extraction based on Natural Language Processing (NLP) [11], has given

a strong impetus to the modelling of legal concepts [5, 2, 6]. We discuss some of these below.

Legal Knowledge Interchange Format (LKIF) is a specification that includes a legal core ontology and a legal rule language that closely represents legal knowledge and reasoning [10, 17, 12]. LKIF does not provide mechanisms to handle concurrent interpretations of a legal source; more specifically, while it might be possible to represent the content of the individual (alternative) interpretations, it is not possible to specify that these representations are mutually exclusive.

RuleML is a family of languages, whose modular system of schemas for XML permits high-precision web rule interchange (`http://wiki.ruleml.org/index.php/RuleML_Home`). LegalRuleML is part of this family of languages. RuleML distinguishes deliberation from reaction rules. Deliberation rules include modal and derivation rules, e.g. facts, queries, and Horn rules. Reaction rules include Complex Event Processing, Knowledge Representation, Event-Condition-Action, and Production. RuleML rules can combine derivation and reaction rules, allowing uniform XML serialization across rules.

RIF (`http://www.w3.org/2005/rules/wiki/RIF_Working_Group`) is a W3C recommendation for a standard Web Rule Interchange Format to exchange rule sets among different rule systems. It makes use of Internationalized Resource Identifiers and supports XML Schema data types. The RIF architecture is conceived as a family of languages. A RIF dialect is a rule-based language with an XML syntax and a well-defined semantics. RIF does not provide direct support for adequate representation of legal rules and legal reasoning since they do not support e.g. logic-based negation, non-monotonic reasoning, events and temporal metadata, among other relevant features.

The Semantics of Business Vocabulary and Business Rules (SBVR) [20] provides a controlled natural language [25] of fixed and finite vocabulary and syntactic form for the expression of the terminology, facts, and rules for business documents across a range of business activities and organisations. SBVR has an associated XML Metadata Interchange (XMI), which supports the interchange of documents across businesses. SBVR and LegalRuleML are closely related in that both provide XML encodings of the semantics of terminology, facts, and rules. SBVR bears on business rules, which may or may not have legal standing; LegalRuleML represents statements of legal standing. LegalRuleML's temporal notions of *enforceability*, *efficacy*, and *applicability* are not provided in SBVR. LegalRuleML enables the expression of *defeasibility*, a rich range of *deontic concepts*, and associated concepts of *penalty* and *reparation*.

Given this context, the LegalRuleML Technical Committee has focused on three specific needs [21, 1]:

1. To close the gap between natural language text description and semantic norm modelling.
2. To provide an expressive XML standard for modelling normative rules that satisfies legal domain requirements [15, 16].
3. To extend the Linked Open Data [3] approach to modelling from raw data (acts, contracts, court files, judgements, etc.) to legal concepts and rules along with their functionality and usage.

The main novelty of LegalRuleML and the contribution of this paper is that it provides a formal representation for alternative interpretations of a legal source (textual provision), which is not found in other formal modelling languages.

3 Case Study

In this section we propose a real life case (taken from the Italian legal system and jurisprudence, originally discussed in [14]) depending on multiple (alternative) interpretation of a norm, and we show possible formalisations of the case and the interpretations. In the next section we are going to use the formal representations developed in this section to illustrate the LegalRuleML mechanisms to cope with the phenomenon of multiple interpretations. The case is based on a dispute of Art. 1, Comma 2, Law 379/1990. The article recites

> The benefit referred to in comma 1 shall be paid in an amount equal 80 per cent of five-twelfths of the income earned and reported for tax purposes by the freelancer in the second year preceding the year of application.[1]

The case 18/96, Bologna Tribunal, Imola Section, concerns the interpretation of the conjunction in *the income earned* ***and*** *reported for tax purposes.*...

A fundamental and unalienable principle of legal language is its close connection with natural language; in particular, the interpretation of a textual provision should be the ordinary meaning conveyed by the text of the provision taking into account its context in the act in which it appears and the purpose or object underlying the act. For example, in the Italian legal systems this connection is prescribed by Article 12 of the Preleggi, Italian Civil Code, stating

> In applying a statute, the interpreter should not attribute to it a meaning different from that made evident by the proper meaning of the words and by their connection, as well as by the intention of the law maker.[2]

Accordingly, the literal interpretation of the norm is given by the rule

$$earned(x, y-2) \wedge reported(x, y-2) \Rightarrow \left[\mathsf{OBL}^{auxiliary=freelancer}_{bearer=employer}\right] paybenefit(f(x), y) \quad (1)$$

The arguments of the predicates *earned* and *reported* are the income x earned/reported in the year in the second argument $(y-2)$. Similarly for *paybenefit* where the function f encodes the computation of the value of the benefit based on the value of the income x. However, according to the Italian taxation legislation in force at the time of the dispute the income received in one year is reported for tax purpose the year after the year it has

[1] L'indennità di cui al comma 1 viene corrisposta in misura pari all'80 per cento di cinque dodicesimi del reddito percepito e denunciato ai fini fiscali dalla libera professionista nel secondo anno precedente a quello della domanda.

[2] Nell'applicare la legge non si può ad essa attribuire altro senso che quello fatto palese dal significato proprio delle parole secondo la connessione di esse, e dalla intenzione del legislatore.

been earned. Thus, for example, the income earned in 1995 is reported in 1996. This principle can be formulated as follows:

$$earned(x, y) \rightarrow reported(x, y + 1) \tag{2}$$
$$reported(x, y) \rightarrow earned(x, y - 1) \tag{3}$$

Consider now the *Income* constant obtained by applying the Russell's definite description operator (ι) on the conjunction in the left-hand side of (1).

$$Income = \iota x(earned(x, y) \wedge reported(x, y)) \tag{4}$$

The conclusion is that the constant *Income* is not denoting, i.e., the interpretation of *Income* is $\emptyset$, thus there is no income "entity" that is earned and reported in one and the same year. Hence, the left hand side of the rule in (1) never holds, and the rule never fires, against the intentions of the legislator.

Based on the textual provision two possible interpretations are possible: in the first interpretation the temporal expression "in the second year preceding the year of application" refers to the income earned in the second year preceding the application, while in the second interpretation it refers to the income reported for tax purposed in the second year preceding the application. For example, for an application in year 1998, the first interpretation bases the computation on the income earned in 1996 (and reported in 1997), while for the second interpretation, the value of the benefit is computed starting from the income reported in 1996 (and earned in 1995). Accordingly, the first interpretation, the interpretation proposed by the freelancer in the case, can be formalised by the rule

$$earned(x, y - 2) \Rightarrow \left[\mathsf{OBL}^{auxiliary=freelancer}_{bearer=employer}\right] paybenefit(f(x), y) \tag{5}$$

Similarly the second interpretation, the interpretation proposed by the employer, can be represented by the rule[3]

$$reported(x, y - 2) \Rightarrow \left[\mathsf{OBL}^{auxiliary=freelancer}_{bearer=employer}\right] paybenefit(f(x), y) \tag{6}$$

The task of the Judge was to decide which of the two interpretations has to be used for the application of the norm. In the case the Judge argue in favour of the interpretation advanced by the freelancer.

4 LegalRuleML Representation of the Case Study

In the previous section we presented three possible interpretations of the norm, the literal interpretation, the interpretation of the freelancer and the interpretation of the

[3] Alternatively, we could use $earned(x, y - 3) \Rightarrow \left[\mathsf{OBL}^{auxiliary=freelancer}_{bearer=employer}\right] paybenefit(f(x))$, while, from a formal point of view, it is semantically equivalent to (6) it is less close in meaning to the textual provision than its counterpart: the temporal reference in the argument would "third year preceding the year of the application".

employer. Here we are going to present the LegalRuleML fragments required to encode the formalisations corresponding to the three interpretations. The formalisations of these three statements can be represented as prescriptive rules which are encoded by <lrml:PrescriptiveStatement> blocks in LegalRuleML, each containing one <ruleml:Rule> Template. The following fragment corresponds to the literal interpretation, i.e., (1)

```
<lrml:PrescriptiveStatement key="literal">
  <ruleml:Rule closure="universal" key=":literal-template">
    <ruleml:if>
      <ruleml:And>
        <ruleml:Atom key=":atom-earned">
          <ruleml:Rel iri="#earned"/>
          <ruleml:Var>income</ruleml:Var>
          <ruleml:Expr>
            <ruleml:Fun iri="#subtract"/>
             <ruleml:Var>year</ruleml:Var>
            <ruleml:Data xsi:type="xs:integer">2</ruleml:Data>
          </ruleml:Expr>
        </ruleml:Atom>
        <ruleml:Atom key=":atom-reported">
          <ruleml:Rel iri="#reported"/>
          <ruleml:Var>income</ruleml:Var>
          <ruleml:Expr>
            <ruleml:Fun iri="#subtract"/>
            <ruleml:Var>year</ruleml:Var>
            <ruleml:Data xsi:type="xs:integer">2</ruleml:Data>
          </ruleml:Expr>
        </ruleml:Atom>
      </ruleml:And>
    </ruleml:if>
    <ruleml:then>
      <lrml:Obligation key="obl-paybenefit">
        <ruleml:slot>
          <lrml:Bearer/>
          <ruleml:Var>Employer</ruleml:Var>
        </ruleml:slot>
        <ruleml:slot>
          <lrml:AuxiliaryParty/>
          <ruleml:Var>Freelancer</ruleml:Var>
        </ruleml:slot>
        <ruleml:Atom>
          <ruleml:Rel iri="#paybenefit"/>
          <ruleml:Expr>
            <ruleml:Fun iri="#80_percent_of_five-twelfths_of"/>
            <ruleml:Var>income</ruleml:Var>
          </ruleml:Expr>
          <ruleml:Var>year</ruleml:Var>
        </ruleml:Atom>
      </lrml:Obligation>
    </ruleml:then>
  </ruleml:Rule>
</lrml:PrescriptiveStatement>
```

Since LegalRuleML is built on top of RuleML we can reuse all RuleML facilities, in particular we can use <ruleml:Expr> and <ruleml:Fun> to encode the computation of the benefit to be paid to the freelancer.

The next snippet captures the interpretation of the freelancer, i.e., (5).

```
<lrml:PrescriptiveStatement key="freelancer">
  <ruleml:Rule closure="universal" key=":freelancer-template">
    <ruleml:if>
      <ruleml:Atom keyref=":atom-earned"/>
    </ruleml:if>
    <ruleml:then>
      <lrml:Obligation keyref="#obl-paybenefit"/>
    </ruleml:then>
  </ruleml:Rule>
</lrml:PrescriptiveStatement>
```

Notice that inside this statement we can use keyrefs to refer to the elements already defined in the block corresponding to the literal interpretation. Similar considerations apply to the block modelling (6), the employer's interpretation, below.

```
<lrml:PrescriptiveStatement key="employer">
  <ruleml:Rule closure="universal" key=":employer-template">
    <ruleml:if>
      <ruleml:Atom keyref=":atom-reported"/>
    </ruleml:if>
      <ruleml:then>
        <lrml:Obligation keyref="#obl-paybenefit"/>
      </ruleml:then>
  </ruleml:Rule>
</lrml:PrescriptiveStatement>
```

The following LegalRuleML Constitutive Statement represents the principle expressed in (2), that earned income will be reported in the following year. Because a Constitutive Statement defines concepts and does not prescribe behaviours, the consequent of its <ruleml:Rule> Template does not contain deontic operators.

```
 <lrml:ConstitutiveStatement key="tax1">
  <ruleml:Rule closure="universal">
    <ruleml:if>
      <ruleml:Atom>
        <ruleml:Rel iri="#earned"/>
        <ruleml:Var>income</ruleml:Var>
        <ruleml:Var>year</ruleml:Var>
      </ruleml:Atom>
    </ruleml:if>
    <ruleml:then>
      <ruleml:Atom>
        <ruleml:Rel iri="#reported"/>
        <ruleml:Var>income</ruleml:Var>
        <ruleml:Expr key=":year+1">
          <ruleml:Fun iri="#add"/>
          <ruleml:Var>year</ruleml:Var>
          <ruleml:Data xsi:type="xs:integer">1</ruleml:Data>
        </ruleml:Expr>
```

```
      </ruleml:Atom>
    </ruleml:then>
  </ruleml:Rule>
</lrml:ConstitutiveStatement>
```

Similarly, the following fragment represents the principle that reported income was earned in the previous year, as expressed in (3).

```
 <lrml:ConstitutiveStatement key="tax2">
  <ruleml:Rule closure="universal">
    <ruleml:if>
      <ruleml:Atom>
        <ruleml:Rel iri="#reported"/>
        <ruleml:Var>income</ruleml:Var>
        <ruleml:Var>year</ruleml:Var>
      </ruleml:Atom>
    </ruleml:if>
    <ruleml:then>
      <ruleml:Atom>
        <ruleml:Rel iri="#earned"/>
        <ruleml:Var>income</ruleml:Var>
        <ruleml:Expr key=":year-1">
          <ruleml:Fun iri="#subtract"/>
          <ruleml:Var>year</ruleml:Var>
          <ruleml:Data xsi:type="xs:integer">1</ruleml:Data>
        </ruleml:Expr>
      </ruleml:Atom>
    </ruleml:then>
  </ruleml:Rule>
</lrml:ConstitutiveStatement>
```

After the renderings of the alternative interpretations and the relationships between the predicates *earned* and *reported* given by the three constitutive rules, we have to specify that they are mutually exclusive formalisation of the same norm. This can be achieved by following Alternatives block that represents a mutually-exclusive collection of renderings of the Legal Norms from the Legal Source `#ls1`. The `<lrml:LegalSource>` with key `#ls1`, not shown in the text, contains the references to the actual text of the norm.

```
<lrml:Alternatives key="maternity-alts">
  <lrml:Comment> These alternatives are mutually
     incompatible formalizations of the same legal source: keyref="#ls1".
  </lrml:Comment>
  <lrml:hasAlternative keyref="#literal" />
  <lrml:hasAlternative keyref="#freelancer" />
  <lrml:hasAlternative keyref="#employer" />
</lrml:Alternatives>
```

A `<lrml:Context>` block is used to render a collection of Associations, e.g. the Association of a Legal Source with a rendering of it as a LegalRuleML Statement, or to constrain other Contexts with respect to Alternatives. The following Context establishes a constraint that at most one of the Alternatives from the collection `#maternity-alts` may be selected by each Context:

```
<lrml:Context key="maternity-alts-ctxt">
  <lrml:appliesAssociations keyref="#asn-alts"/>
  <lrml:appliesAlternatives keyref="#maternity-alts"/>
</lrml:Context>
```

The Context metadata, e.g. authorship, source, authority, temporal and jurisdictional properties, are specified in an external (to the Context) Association block with identifier `asn-alts`, not shown in the paper, which is referenced using `keyref`. Similarly other Context blocks (also not shown in the paper) are given with the metadata about the authors of the various Statements. This permits to establish the provenance of the interpretations.

In the following fragment, a particular Alternative – that proposed by the freelancer – is selected, leading to the generation of the corresponding `<ruleml:Rule>` from the rule Template `:freelancer-template`.

```
<lrml:Context key="adjudication">
  <lrml:appliesAssociation keyref="#asn-adjudication"/>
  <lrml:inScope keyref="#freelancer"/>
</lrml:Context>
```

Unlike the first Context block, this one contains an `<lrml:inScope>` element. Such Contexts render interpretations that select one or more Statements as their scope of interpretation. When a Context is processed for presentation or inference, Legal Rules[4] are generated from the `<ruleml:Rule>` Templates of in-scope Statements, annotated and optionally modified semantically by the Associations of the Context.

In this example the external Association `asn-adjudication` links the metadata for the adjudication of the case with a particular rendering of the norm, the rendering `freelancer`, corresponding to the interpretation proposed by the freelancer and confirmed by the judge[5].

5 Conclusions and Future Work

In this paper, we presented the mechanisms for the representation of (mutually incompatible) alternative interpretations of legal sources (textual provisions) in a LegalRuleML document. Specifically, we introduced the `<lrml:Alternatives>` element that is to be used to specify alternative formal renderings of a legal source, where the alternatives in a block are mutually exclusive. The key idea is that each rendering corresponds to an interpretation of the legal source. Using `<lrml:Alternatives>` along with `<lrml:Context>` blocks, we can specify the different formal renderings of a legal source and their metadata that associates them with other elements in a LegalRuleML document, e.g. the interpreter of the legal sources, the time of such interpretations, and the context where

[4] In this paper, we focus on Prescriptive and Constitutive Statements, which always lead to generated Legal Rules. However, in the general case, e.g. `<lrml:FactualStatement>`, something other than a Legal Rule may be generated when a Statement is in scope.

[5] The full example is available from `https://tools.oasis-open.org/version-control/browse/wsvn/legalruleml/trunk/examples/approved/maternity_alternatives_compact.lrml`

such interpretations apply. Furthermore, we presented a real life case that illustrates how to use <lrml:Alternatives> to model the different interpretations of an ambiguous legal source such as arise in a legal dispute. An important advantage of the use of <lrml:Alternatives> is that it reduces redundancies in encoding the formalisation of a legal document. Different interpretations of a legal document can occur for many reasons (e.g., different readings of one norm by the parties involved in a legal dispute, different granularity of representation required by different applications, different interpretations with respect different (sub-)jurisdictions, the change of interpretation of terms over time, ...). The mechanism we have proposed does not force the author of a LegalRuleML document to duplicate and modify the document just to accommodate every different interpretation of a legal source. All the author has to do is to create an <lrml:Alternatives> block in a single LegalRuleML document, add the various alternatives in the block, and refer to it from <lrml:Context> blocks that associate metadata with sets of statements. Then by filtering with respect to the metadata associated with an alternative, one can generate the manifestation of the document corresponding to alternative interpretations selected by the filtering conditions, employing the <lrml:Alternatives> expression to ensure that mutually incompatible alternatives are not simultaneously asserted in the same context.

The LegalRuleML syntax for the metadata collections, e.g. Alternatives, Jurisdictions, LegalSources, was designed to facilitate the exposure of these relationships as Linked Open Data (LOD). The RDF Collection structure is particularly appropriate for this, because it can be closed, indicating that the collection contains only the entities explicitly asserted to belong to it. An XSLT transformation will be developed to convert the LegalRuleML XML into RDF, while an RDFS metamodel will capture additional constraints. We envision that the major benefits of the RDF representation of LegalRuleML are the possibility to integrate the legal knowledge with information stored in other Open Data repositories and triple stores, and the ability to use tools such as SPARQL reasoners for preprocessing LegalRuleML documents before passing data to specialised legal reasoners.

References

1. T. Athan, H. Boley, G. Governatori, M. Palmirani, A. Paschke, and A. Wyner. OASIS LegalRuleML. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Law (ICAIL 2013)*, pages 3–12. ACM, 2013.
2. V. R. Benjamins, P. Casanovas, J. Breuker, and A. Gangemi, editors. *Law and the Semantic Web: Legal Ontologies, Methodologies, Legal Information Retrieval and Applications*. Springer, 2005.
3. T. Berners-Lee. Long live the web: A call for continued open standards and neutrality. *Scientific American*, (22 November):1–6, 2010.
4. B. H. Bix. Legal interpretation and the philosophy of language. In P. Tiersma and L. Solan, editors, *The Oxford Handbook of Language and Law*, pages 145–155. Oxford, 2012.
5. A. Boer, R. Winkels, and F. Vitali. Metalex XML and the legal knowledge interchange format. In P. Casanovas, G. Sartor, N. Casellas, and R. Rubino, editors, *Computable Models of the Law, Languages, Dialogues, Games, Ontologies*, volume 4884 of *LNCSe*, pages 21–41. Springer, 2008.

6. J. Breuker, A. Boer, R. Hoekstra, and K. van den Berg. Developing content for lkif: Ontologies and frameworks for legal reasoning. In T. M. van Engers, editor, *JURIX 2006*, pages 169–174. IOS Press, 2006.
7. J. R. Curran, S. Clark, and J. Bos. Linguistically motivated large-scale NLP with C&C and Boxer. In *ACL*. The Association for Computer Linguistics, 2007.
8. M.-C. de Marneffe, B. MacCartney, and C. D. Manning. Generating typed dependency parses from phrase structure parses. In *IN PROC. INT'L CONF. ON LANGUAGE RESOURCES AND EVALUATION (LREC*, pages 449–454, 2006.
9. F. de Saussure. *Cours de Linguistique Générale*. Payot, Lausanne and Paris, 1916.
10. Estrella Project. The legal knowledge interchange format (LKIF). Technical report, ESTRELLA Project, 2008.
11. E. Francesconi, S. Montemagni, W. Peters, and D. Tiscornia, editors. *Semantic Processing of Legal Texts: Where the Language of Law Meets the Law of Language*, volume 6036 of *LNCS*. Springer, 2010.
12. T. F. Gordon. Constructing legal arguments with rules in the legal knowledge interchange format (LKIF). In P. Casanovas, N. Casellas, R. Rubino, and G. Sartor, editors, *Computable Models of the Law*, number 4884 in LNCS, pages 162–184. Springer Verlag, 2008.
13. T. F. Gordon, G. Governatori, and A. Rotolo. Rules and norms: Requirements for rule interchange languages in the legal domain. In G. Governatori, J. Hall, and A. Paschke, editors, *RuleML*, volume 5858 of *LNCS*, pages 282–296. Springer, 2009.
14. G. Governatori. *Un modello formale per il ragionamento giuridico*. PhD thesis, CIRFID, Università di Bologna, 1997.
15. G. Governatori and A. Rotolo. Norm compliance in business process modeling. In M. Dean, J. Hall, A. Rotolo, and S. Tabet, editors, *RuleML*, volume 6403 of *LNCS*, pages 194–209. Springer, 2010.
16. B. N. Grosof. Representing e-commerce rules via situated courteous logic programs in RuleML. *Electronic Commerce Research and Applications*, 3(1):2–20, 2004.
17. R. Hoekstra, J. Breuker, M. D. Bello, and A. Boer. LKIF core: Principled ontology development for the legal domain. In *Law, Ontologies and the Semantic Web*, pages 21–52, Amsterdam, The Netherlands, 2009.
18. S. Lappin, editor. *The Handbook of Contemporary Semantic Theory*. Blackwell Publishers, 1997.
19. C. Lupo, F. Vitali, E. Francesconi, M. Palmirani, R. Winkels, E. de Maat, A. Boer, and P. Mascellani. General XML format(s) for legal sources - Estrella European Project IST-2004-027655. Technical report, Faculty of Law, University of Amsterdam, Amsterdam, The Netherlands, 2007.
20. OMG. Semantics of Business Vocabulary and Business Rules (SBVR). formal specification, v1.0. Technical report, The Object Management Group, 2008.
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 (RuleML 2011)*, pages 298–312. Springer Berlin Heidelberg, 2011.
22. R. Poscher. Ambiguity and vagueness in legal interpretation. In P. Tiersma and L. Solan, editors, *The Oxford Handbook of Language and Law*, pages 128–144. Oxford, 2012.
23. A. Scalia and B. A. Garner. *Reading Law: The Interpretation of Legal Texts*. West, 2012.
24. L. M. Solan. Linguistic issues in statutory interpretation. In P. Tiersma and L. Solan, editors, *The Oxford Handbook of Language and Law*, pages 87–99. Oxford, 2012.
25. 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 *Proceedings of the 2009 conference on Controlled natural language*, CNL'09, pages 281–289, Berlin, Heidelberg, 2010. Springer-Verlag.