RAWE: A Web Editor for Rule Markup in LegalRuleML

Monica Palmirani[1], Luca Cervone[1] Octavian Bujor[1], Marco Chiappetta[1]

[1]CIRSFID, University of Bologna.
{monica.palmirani, luca.cervone, octavian.bujor, marco.chiappetta}@unibo.it

Abstract. This paper presents a Web editor (RAWE: Rules Advanced Web Editor) for marking up legal rules starting from legally binding texts. The Web editor exploits the legal information embedded in the Akoma Ntoso markup, in combination with and XML techniques, so as to help the legal-knowledge engineer model legal rules and convert them into LegalRuleML, an OASIS XML standard candidate.

Keywords: Legal Reasoning, Akoma Ntoso, LKIF-core, LegalRuleML.

1. Introduction

This paper presents a Web editor for marking up legal texts in a legal document's main structure, normative references, and legal metadata using the Akoma Ntoso [2] [13] [25] XML standard, now undergoing the OASIS standardization process. The same Web editor exploits the legal information embedded in legal markup, in combination with XML techniques, to help the legal-knowledge engineer model legal rules using a logic formalism and convert them into LegalRuleML [1] [23][24], another OASIS XML standard candidate. The two standards—Akoma Ntoso and LegalRuleML—are complementary in implementing the legal-knowledge modelling and representation of legal documents. The main goal of the RAWE Web editor is to provide a tool capable of managing in an integrated way the advantages of the Akoma Ntoso and of LegalRuleML, applying the isomorphism principle [3][9][22] to connect, as far as possible, legally binding textual provisions with the logic formalism expressed using rules. Usually, AI&Law experts are too focused on the task of applying a logic formalism to achieve isomorphism, but the legal experts (judges, lawyers, and administrators) are interested in verifying the results of the legal reasoning engine and in finding evidence in the legally binding text.

Secondly, a legal text changes over time, and so the rules need to be updated accordingly. If the isomorphism principle is not applied, it is quite difficult to determine whether those rules need to be updated. The RAWE editor helps to maintain text and rules aligned and to minimize manual markup activity.

Thirdly, the aim of the RAWE is to show how it is possible to export LegalRuleML in RDF serialization to favour Linked Open Data interoperability. Finally, in the future, the same editor will export LegalRuleML files in other

proprietary languages, like SPINdle [15] or Carneades [8][11], so as to permit legal reasoning.

2. From Open Text to Open Rules

The first point to be made in clarifying the goals RAWE would like to achieve is to draw a distinction among three conceptual layers: norms (abstract mandatory commands concerning rights or duties), textual provisions (sequences of texts), and rules (rendering of the text into logical rules).

A **norm**, following Kelsen's definition [14], is an abstract mandatory command concerning rights or duties. A norm is usually expressed in writing using legal texts or in an oral way (e.g., a social norm, an oral contract) or in other representations (e.g., symbolic road signs).

Textual provisions (or simply *provisions)* are the instantiation of general norms in one possible textual representation (a sentence, article, or paragraph).

Legal rules are interpretations of one or more provisions formalized using logical rules in the form of antecedent and consequent. Sometimes several provisions will form a single rule, or a single provision may include multiple rules.

Usually, in the state of the art, AI&Law scholars focus their attention only on the rule modelling and on the foundational logical theory, and apart from the isomorphism principle [3], the connection with the text over time and the ontology aspects have been neglected. There is an important theoretical debate in the AI&Law community on the interpretation of the legal textual provisions expressed in natural language and on the canonization of rules using logical formalisms [4]. The prevalent theory is now oriented towards *hybrid interpretation* [27] (rather than pure *textualism*, or pure *interpretation*). We want to make visible in the text the "evidence" that there is a minimal but reasonable interconnection, following the legal theory of interpretation, with a logical rule in a formal representation. This exercise sometimes forces the legal-knowledge expert to split the original provision into two or more rules, or to duplicate the rules, or to compress several sentences into a single rule. In this scenario, we have to manage an N:M relationship among norms, textual provisions, and the ontology that we want to capture and represent maintaining a strong separation among these three levels.

Nevertheless, it is obvious that the isomorphism approach alone presents some exceptions and limitations that need to be balanced in a reasonable way. We have at least three cases where the legal rules have no textual link: (i) when we have implicit rules deriving from the general principles of the legal system (e.g., *lex superior*, *lex specialis*, *lex posterior*); (ii) when the legal-knowledge engineer includes a personal interpretation as a summary of his/her expertise; and (iii) when the legal-reasoning engine produces rules. In these cases the Web editor provides metadata to distinguish those rules deriving from the legal text from those that are a free interpretation of the rules' author. The RAWE editor permits multiple interpretations of the same legal text and makes it possible to follow the isomorphism principle, but also to derogate from it if need be.

Finally, the Web editor exports all the metadata in RDF format to favour the interconnection of Legal Open Data with Linked Open Data. The goal is to release RDF triples about the rule knowledge base, in such a way as to connect that with other datasets available in the Linked Open Data Cloud. This permits more-effective filters of the legal resources in the Semantic Web domain (e.g., geo-localizing legal resources on the map using the jurisdiction and the temporal metadata filter to find the legal rules relevant to a given context, such as environment law or construction law).

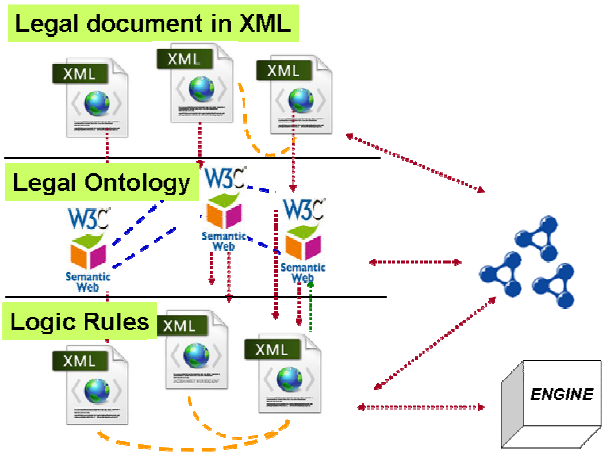


Fig. 1 – *Scenario of relationships among different layers in legal knowledge modelling*

3. Related Work

The AI&Law community itself [27] has spent the last two decades modelling legal norms using different logics and formalisms, usually fed manually to a legal-reasoning engine. Some visual tools [11] or editors [15][19] in the past have been developed to model rules, but the methodology used starts from a reinterpretation of the legal source text by a legal-knowledge engineer who extracts the norms, applies models and theory using a logic representation, and finally represents them with a particular formalism. The RAWE approach is different: it starts from the legal text marked up in some Legal XML standard and, exploiting the text's regularity, detects some metadata that is also useful for modelling rules.

Over the last decade, several Legal XML standards have arisen for describing legal texts (Akoma Ntoso, CEN Metalex [5] [16]) and rules (RuleML, RIF, SWRL, etc.), but the two communities are mostly separated, and they pursue their goal separately. In the meantime, the Semantic Web, and in particular legal ontology research, combined with the NLP extraction of semantics, has given a great impulse to the modelling of legal concepts [17][18][20][7][6][12][26]. In this scenario there is an urgent need to close the gap between the text description, represented using XML

techniques, and the norms formalized with logical rules, this in order to realize an integrated and self-contained representation. There are three main reasons:

- Legal knowledge is currently presented in a disjointed way in the original text that inspired the logical modelling. This disconnection between legal-document management and the logical representation of the embedded rules strongly affects the real usage of the legal-document knowledge in favour of citizens, public administrations, and businesses (e.g., contracts, insurance regulation, banking soft law).
- Management of changes undergone over time by legal documents—especially acts, regulations, and contracts—that by nature are variable and subject to frequent modifications, significantly affecting the coordination between the text and the rules that should be remodelled.
- The legal validity of the text as authentically approved by the competent entities (e.g., contractors) should be preserved across all manipulations. On the other hand, it is important to connect legal document resources, which themselves include many legality values (e.g., authenticity, integrity, evidence in trial, written form, etc.), with the multiple interpretations coming from legal-knowledge modelling.

Certainly, one of the main challenges over the last five years has been to acquire the ability to capture, with the help of NLP techniques, all the relevant legal knowledge embedded in a legal document and to represent it in an appropriate formal model. However, there hasn't been significant progress on the state of the art in this respect, especially in languages other than English. So it is important to improve the user interface technique to help the legal-knowledge expert to easily model legal rules and prepare an environment for a future NLP integration. RAWE is the only Web editor in the state of the art that can model legal texts and rules in a coordinated and consistent way using a WYSIWYG interface exploiting two important legal XML standards: Akoma Ntoso and LegalRuleML.

4. Akoma Ntoso and LegalRuleML Synergy

As mentioned before, Akoma Ntoso and LegalRuleML are two XML standards for modelling and representing legal documents. RAWE can coordinate the knowledge captured with these two standards so as to help the end user mark up the legal rules using a logic formalism enriched with temporal parameters.

Akoma Ntoso is specifically designed to model a legal document's structure and legal metadata, like the preface, preamble, sections, conclusions, normative references, dates, and signatures. The Akoma Ntoso metadata block additionally defines the conditions under which the legal textual fragment is valid, effective and in force, while also defining jurisdiction, the document's authority, and other relevant legal metadata, like modifications. All those metadata are also significant in defining the context of a legal rule, helping the legal reasoning engine filter the rules pertinent to a particular case (e.g., infringement of the rule at a given date in 1999).

The following example displays as `<temporalGroup id="t5">` the block that defines the interval of efficacy and enforceability of Section 504 of the US Code.

```
<akomaNtoso>
 <act name="act">
  <meta>
       ... meta data about the legal document ...
   </meta>
   <coverPage>
      Cover page content
   </coverPage>
   <preface>
      ... the preface of the document ...
   </preface>
   <preamble>
       ... the preamble of the document ...
   </preamble>
   <body>
      <clause id="tit17-chp5-sec504-clsc" period="#t5">
       ... the normative part of the document ...
   </body>
   </act>
</akomaNtoso>
```

LegalRuleML is designed to model in logical formalism the norms expressed in a legal text. It does so especially using deontic operators: obligation, right, permission, prohibition. LegalRuleML is also intended to define the context for each rule by providing a set of metadata like the temporal parameters, the original textual sources, the jurisdiction, the author, and the authority of the rules. The fragment below shows the main structure of a LegalRuleML document composed of different metadata blocks defining the author who modelled the text into rules (`<lrml:Agents>`), recording the original legal resources IRI (`<lrml:References>`), and providing the temporal parameters (`<lrml: TemporalCharacteristics>`), the context, and each rule's date of creation (`<lrml:Context key="ruleInfo1" hasCreationDate ="#t8">`). The `<lrml:Contex>` provides the environment in which the rules are valid (time, author, jurisdiction, etc.).

`<lrml:LegalRuleML>`	

`<lrml:Agents>`
`<lrml:Agent key="aut1"`
`sameAs="&unibo;/person.owl#m.palmirani"/>`
`</lrml:Agents>`
	general definition of Agent with value an uri to m.palmirani

`<lrml:References>`
`<lrml:Reference refersTo="ref2"`
`refID="/us/USCode/eng@/main#tit17-sec504-clsc-pnt1"`
`refIDSystemName="AkomaNtoso2.0-2012-10"/>`
`</lrml:References>`
	definition of legal text fragment

`<lrml:TimeInstants>`
`<ruleml:Time key="t6">`
`<ruleml:Data xsi:type="xs:dateTime">1999-12-`
`09T00:00:00.0Z</ruleml:Data>`
`</ruleml:Time>`
`</lrml:TimeInstants>`
	definition of instant time

`<lrml:TemporalCharacteristics key="tblock1">` `<lrml:TemporalCharacteristic key="e2-e">`	definition of intervals and

<lrml:forRuleStatus iri="&lrmlv;#Efficacious"/> <lrml:hasStatusDevelopment iri="&lrmlv;#Ends"/> <lrml:atTimeInstant keyref="#t6"/> </lrml:TemporalCharacteristic> </lrml:TemporalCharacteristics> <lrml:Context key="ruleInfo1" hasCreationDate="#t8"> <lrml:appliesTemporalCharacteristics keyref="#tblock1"/> <lrml:appliesStrength iri="&lrmlv;defeasible"/> <lrml:appliesRole> <lrml:Role iri="&lrmlv;#Author"> <lrml:filledBy keyref="#aut1"/> </lrml:Role> </lrml:appliesRole> <lrml:appliesAuthority keyref="#congress"/> <lrml:appliesJurisdiction keyref="&jurisdictions;us"/> <lrml:appliesSource keyref="#ref2"/> <lrml:toStatement keyref="#rule1"/> </lrml:Context> <lrml:hasStatements key="rulebase-v2"> <lrml:ConstitutiveStatement key="rule1"> <ruleml:if> ...</ruleml:if> <ruleml:then>... </ruleml:then> </lrml:ConstitutiveStatement> </lrml:hasStatements>... </lrml:LegalRuleML>	situations definition of the rule context rule base block

Entering all the <lrml:Contex> information manually for each rule is a really time-consuming task, especially when the legal text has gone through several modifications over time. Moreover, it is difficult to maintain consistency between legal textual provisions and rules in the dynamicity of the legal system. For this reason the RAWE Web editor exploits the information embedded in the Akoma Ntoso text proposition (e.g., section, article), and it reuses those data to define the context of the rules when accurately connected to the legal provision.

The following example presents a fragment of Section 504 of the US Code concerning copyright infringement and the related rules. Section 504 is presented in the version updated at time t5, which in Akoma Ntoso is defined in the <temporalGroup id="t5"> block.

When the end-user selects a portion of the legal text with the mouse in the Web editor window, all the related metadata recorded in Akoma Ntoso are detected and exported in LegalRuleML to model the rules.

The following example shows in the two standards (i) the correspondence among temporal events; (ii) the correspondence among temporal intervals; and (iii) how it is possible to reuse the Akoma Ntoso information in LegalRuleML (compact form).

i) Event definition in the Akoma Ntoso metadata block	**Event definition in LegalRuleML, automatically extracted from the Akoma Ntoso text using mouse-over** <lrml:TimeInstants>

<eventRef source="#rp5" id="e6" type="amendment" date="1999-12-09"/>	<ruleml:Time key="t6"> <ruleml:Data xsi:type="xs:dateTime">1999-12-09T00:00:00.0Z</ruleml:Data> </ruleml:Time> </lrml:TimeInstants>
Intervals definition in Akoma Ntoso in the metadata block <temporalData source="#palmirani"> <temporalGroup id="t5"> <timeInterval refersTo="#inforce" start="e6"/> <timeInterval refersTo="#efficacy" start="e6"/> </temporalGroup> </temporalData>	**Intervals definition in LegalRuleML** <lrml:TemporalCharacteristics key="tblock1"> <lrml:TemporalCharacteristic key="e2-e"> <lrml:forRuleStatus iri="&lrmlv;#Efficacious"/> <lrml:hasStatusDevelopment iri="&lrmlv;#Ends"/> <lrml:atTimeInstant keyref="#t6"/> </lrml:TemporalCharacteristic> </lrml:TemporalCharacteristics>
	Context of rule1 in LegalRuleML, automatically built using Akoma Ntoso information <lrml:Context key="ruleInfo1" > <lrml:appliesTemporalCharacteristics keyref="#tblock1"/> <lrml:appliesStrength iri="&lrmlv;defeasible"/> <lrml:appliesAssociations> <lrml:Associations key="sourceBlock1"> <lrml:Association> <lrml:appliesSource keyref="#sec504-clsc-lst1-pnt2"/> <lrml:toTarget keyref="#rule1"/> </lrml:Association> </lrml:Associations> </lrml:appliesAssociations> <lrml:toRuleText keyref="#rule1"/> </lrml:Context>
Text in Akoma Ntoso <clause id="tit17-chp5-sec504-clsc"> <num>(c)</num> <heading>Statutory Damages.</heading> <list id="tit17-chp5-sec504-clsc-lst1">	**Rule definition in LegalRuleML connected to the textual provision selected by mouse-over** <lrml:Penalty key="rule3-penalty1"> <lrml:Obligation key="rule3-penalty1-obl1"> <ruleml:And> <ruleml:Atom key="rule3-penalty1-obl1-axm1"> <ruleml:Rel

```
     <point id="tit17-chp5-
sec504-clsc-lst1-pnt1">
      <num>(1)</num>
       <content>
<p>-Except as provided by clause
(2) of this subsection, the
copyright owner may elect, at any
time before final judgment is
rendered, to recover, instead of
actual damages and profits, an
award of statutory damages for
all infringements involved in the
action, with respect to any one
work, for which any one infringer
is liable individually, or for
which any two or more infringers
are liable jointly and severally,
in a sum of not less than <span
period="#t5">$750</span> or more
than <span
period="#t5">$30,000</span> as
the court considers just. For the
purposes of this subsection, all
the parts of a compilation or
derivative work constitute one
work.</p>
       </content>
      </point>
```

```
iri="&lrmlv;payFine"> min Pay
</ruleml:Rel>

  <ruleml:Var>X</ruleml:Var>
              <ruleml:Ind>750
</ruleml:Ind>
       </ruleml:Atom>
       <ruleml:Atom key="rule3-
penalty1-obl1-axm1">
              <ruleml:Rel
iri="&lrmlv;payFine"> Pay max
</ruleml:Rel>

  <ruleml:Var>X</ruleml:Var>
              <ruleml:Ind>30,000
</ruleml:Ind>
       </ruleml:Atom>
      </ruleml:And>
    </lrml:Obligation>
   </lrml:Penalty>
```

5. From LegalRuleML Meta-model to RDF Serialization

LegalRuleML was designed based on a meta-model[1] that defines relationships among different classes of the elements in the XML-schema. For helping this approach the technical author of the XML-schema (Tara Athan) implemented also several rdfs schemas. The following fragment of rdfs schema shows the relationship among the element `<lrml:Role>` and the property `<lrml:appliesRole>`. Following this approach all the elements that start with lower case are edges and the elements that start with upper case are nodes of a graph.

```
<rdfs:Class rdf:about="#Role">
    <rdfs:isDefinedBy rdf:resource="&lrmlmm;#"/>
    <rdfs:label>Role</rdfs:label>
    <rdfs:comment>The class of roles played by agents relative to
LegalRuleML things.</rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Thing"/>
</rdfs:Class>
<rdf:Property rdf:about="#appliesRole">
    <rdfs:isDefinedBy rdf:resource="&lrmlmm;#"/>
    <rdfs:label>appliesRole</rdfs:label>
```

[1] Meta model is now under revision and the authors take this version from the OASIS repository: https://tools.oasis-open.org/version-control/browse/wsvn/legalruleml/trunk/schemas/?rev=71&sc=1

```
    <rdfs:comment>A role applied to the targets by
      the subject association or rule context.
    </rdfs:comment>
    <rdfs:domain rdf:resource="#AssociationOrContext"/>
    <rdfs:range rdf:resource="#Role"/>
</rdf:Property>
```

Using this meta-model it is possible to extract some relationships among elements. Some assertions in RDF format about the knowledge base rules are possible especially from the `<lrml:Context>`. These assertions build a set of RDF triples useful for improving information retrieval of the legal rules, and related legal textual sources, in the Semantic Web. The contextualization of the legal rules (e.g. Jurisdiction, Author, Authority, etc.) permits to create enriched connection with the Linked Open Data Cloud (e.g. geo-localization of the legal rules on the maps):

```
<rdf:RDF xmlns:rdf="&rdf;#" xmlns:rdfs="&rdfs;#" xmlns:xs="&xs;"
xmlns:rulemlmm="&rulemlmm;#" xmlns:lrmlmm="&lrmlmm;#">
  <rdf:Description rdf:about="www.example.2.1.1.xml#rule1">
   <lrmlmm:appliesRole>
      <lrmlmm:Role rdf:about="&lrmlv;#Author">
       <lrmlmm:filledBy
rdf:resource="http://monica.palmirani.cirsfid.unibo.it"/>
      </lrmlmm:Role>
   </lrmlmm:appliesRole>
   <lrmlmm:appliesSource rdf:resource="&akn;#sec504-clsc-pnt1"/>
   <lrmlmm:appliesSource rdf:resource="&akn;#sec504-clsc-pnt1"/>
   <lrmlmm:appliesStrength rdf:resource="&lrmlv;defeasible"/>
   <lrmlmm:appliesJurisdiction rdf:resource="&jurisdictions;us"/>
   <lrmlmm:appliesAuthority rdf:resource="&authorities;congress"/>
  </rdf:Description>
</rdf:RDF>
```

The same mechanism should be applied to the other assertions included in the `<lrml:Context>`.

6. RAWE Functionality

RAWE permits the following functionalities:

- Authentication of the end-user and customization of the environment according with the personal profile (e.g., legal system, legal tradition, legal guidelines);
- Multilanguage interface and environment;
- Customized interface and buttons on the basis of the user profile;
- Mark-up of a legal text with Akoma Ntoso standard using parsers to automatically detect the normative references, dates, metadata, and structure of legal documents;
- Record of the XML files in the eXist repository [21];
- Tree of the marked-up elements;
- On-the-fly view in Akoma Ntoso and in LegalRuleML;
- Conversion and export in PDF, XML, ePub, or RDF format;
- Web editor environment with WYSIWIG interface;
- Undo function;

- Contextual functionalities based on the XML tree and XML-schemas;
- Mouse-over for detecting the metadata of a portion of legal text and reuse for modelling legal rules;
- Toolbar for marking up the document's structure;
- Toolbar for marking up legal rules.

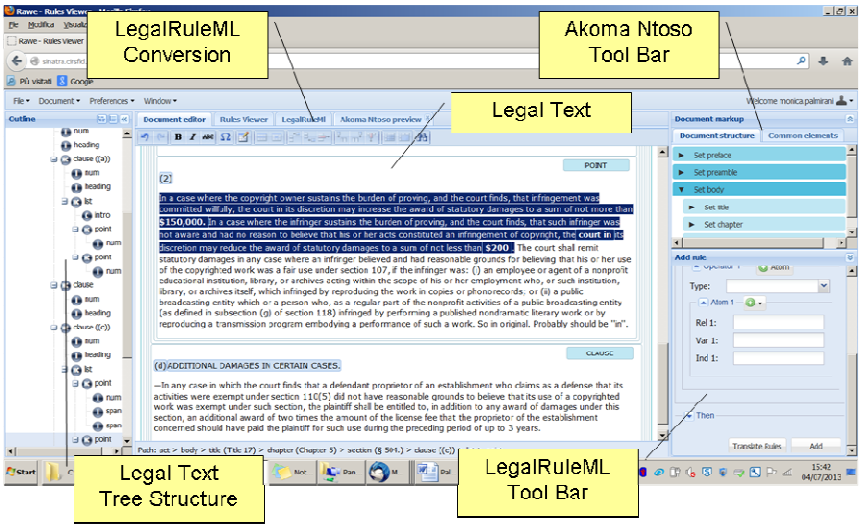


Fig. 2 – *RAWE* Web editor for marking up legal texts and normative rules

There are some critical points that we have faced in the RAWE implementation using HCI techniques:

- ***Contextual Composition of the Rule***. In LegalRuleML we have three groups of rules: Prescriptive, Constitutive and Behaviors. Each group permits some particular modeling following the legal theory (e.g. Prescriptive rule is a sequence of deontic operators, Penalty needs a separate regime, Constitutive rule doesn't include deontic operators, etc.). For this reason RAWE needs to take in consideration the LegalRuleML prescriptive grammar constraints and lead the end user to compose the rules correctly.
- ***Reparation*** is a binary relationship between a penalty and a prescriptive rule or violation. So we found a smart interface way to select the two parts of the relationship and to connect them to each other.
- ***Metadata in Context***. If we need to refine or readjust the context and the related metadata, we need a new toolbar and panel. RAWE permits to readjust the metadata imported by Akoma Ntoso and to add new ones.
- ***Extra isomorphism rules***. Sometimes we need to include extra rules not directly linked to the legal text. RAWE permits to model this particular situation.

However other some critical issues need to be addressed in the future:

- ***Ontology***. Some elements of the rule modeling need to be enriched with the definitions of an external vocabulary or ontology (e.g. LKIF[10]).
- ***Key***. We need to create a naming convention to harmonize the ID definition.

- ***Meta-Rules***. In the future LegalRuleML will be also be able to manage meta-rules (rules about other rules), and we need to find a mechanism for linking rules as antecedents and consequents.
- ***Multiple interpretation***. In this version of the editor is not possible to have multiple interpretations of the same legal textual document fragment.
- ***Granularity***. For now the granularity of the isomorphism is on the rule. In the future we will be able to also manage the same functionality on the body, head, and atom.

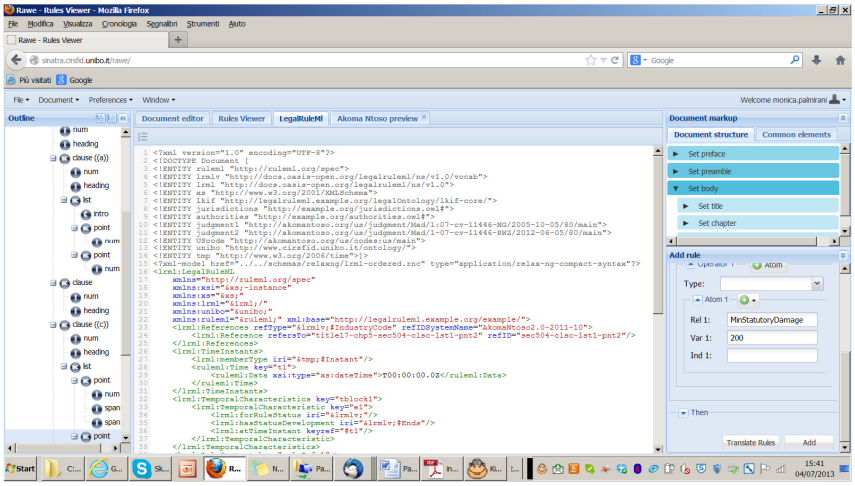


Fig. 3 – *RAWE* conversion of a rule in the LegalRuleML standard

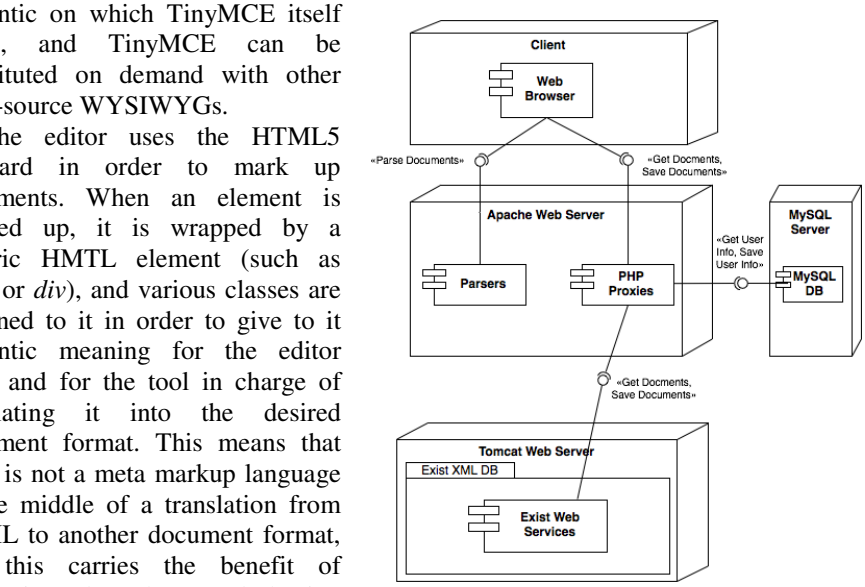
7. The RAWE Architecture

RAWE is a specialized Web editor developed using several open-source technologies, such as Sencha ExtJS 4.1 and TinyMCE.

Sencha ExtJS is an MVC framework that makes it possible to build an extraordinarily rich Web application. It supplies the instruments with which to easily develop the core of the application based on the Model View Controller pattern, and, moreover, it comes with a big range of user interface widgets. The other core strength of ExtJS lies in its component design. If the developer needs a new component that is not yet developed, the default components can be extended and the result is encapsulated in the default components. ExtJS is also completely cross-browser, so it is possible to deliver the application on a wide range of browser and operating systems. The latest smart phone and tablet browser are also covered, so it is possible to use ExtJS-based applications with touch screens and gestures.

TinyMCE is a platform-independent Web-based Javascript HTML WYSIWYG editor control. It can convert HTML text area fields or other HTML elements into editor instances. We integrated it into ExtJS, developing a new component for the

framework. The component retains all the functionality of the TinyMCE editor, but the effects of those functionalities are intercepted by the core of the ExtJS application. With this strategy, each event handled by the editor simply fires other events handled by the other components of the application. This means that there is no specific semantic on which TinyMCE itself relies, and TinyMCE can be substituted on demand with other open-source WYSIWYGs.



The editor uses the HTML5 standard in order to mark up documents. When an element is marked up, it is wrapped by a generic HMTL element (such as *span* or *div*), and various classes are assigned to it in order to give to it semantic meaning for the editor itself and for the tool in charge of translating it into the desired document format. This means that there is not a meta markup language in the middle of a translation from HTML to another document format, and this carries the benefit of preventing data loss and having immediate access to the HTML version of the document without further conversions.

8. Conclusion

We have presented RAWE, a Web editor for marking up legal rules exploiting the previous markup of legal texts in Akoma Ntoso. RAWE is developed to enable application of the isomorphism principle; nevertheless, it is also open to the addition of rules not properly linked with the legal textual provisions, this in order to permit multiple interpretations or the inclusion of implicit rules. RAWE transforms all the rules in LegalRuleML and it saves them in a native XML repository, *eXist*. It is also possible to export the outcomes to a XML file. Finally, RAWE can convert in RDF the `<lrml:Context>` for creating a repository capable, in the future, of implementing an endpoint SPARQL for managing a better filter of legal resources in the Linked Open Data Cloud. Future work will be focused on the critical points stressed in the paper for managing advanced features.

References

[1] Athan T., Boley H., Governatori G., Palmirani M., Paschke A., Wyner A.: OASIS LegalRuleML. In Bart Verheij, ed, Proceedings of 14th International Conference on Artificial Intelligence and Law (ICAIL 2013). ACM, 2013.

[2] Barabucci G., Cervone L., Palmirani M., Peroni S., Vitali F.: Multi-layer Markup and Ontological Structures in Akoma Ntoso. In: LNCS 6237/2010, pp. 133-149, Springer, 2010.
[3] Bench-Capon T. and Coenen F.: Isomorphism and legal knowledge based systems. Artificial Intelligence and Law, 1(1):65–86, 1992.
[4] Boella G., Governatori G., Rotolo A., Torre L.V.D.: A Formal Study on Legal Compliance and Interpretation. ;In AICOL Workshops(2009), Springer, 162-183, 2011.
[5] Boer A., Hoekstra R., de Maat E., Hupkes E., Vitali F., Palmirani M., Rátai B.: CEN Metalex Workshop Agreement (2009-08-28 proposal). http://www.metalex.eu/WA/proposal.
[6] Breuker J., Boer A., Hoekstra R., Van Den Berg C.: Developing Content for LKIF: Ontologies and Framework for Legal Reasoning, in Legal Knowledge and Information Systems, JURIX 2006, pp.41-50, ISO Press, Amsterdam, 2006.
[7] Brighi R., Lesmo L., Mazzei A., Palmirani M., Radicioni D.: Towards Semantic Interpretation of Legal Modifications through Deep Syntactic Analysis. JURIX 2008: 202-206, 2008.
[8] Ceci, M., and Gordon, T. F.: Browsing case-law: An application of the carneades argumentation system. In Proceedings of the RuleML2012@ECAI Challenge, H. Aït-Kaci, Y.-J. Hu, G. J. Nalepa, M. Palmirani, and D. Roman, Eds., vol. 874, pp. 79-95.
[9] Gordon T. F., Governatori G., Rotolo A.: Rules and Norms: Requirements for Rule Interchange Languages in the Legal Domain. RuleML 2009: pp. 282-296, Springer, 2009.
[10] Gordon T. F.: Constructing Legal Arguments with Rules in the Legal Knowledge Interchange Format (LKIF). In: Computable Models of the Law, Languages, Dialogues, Games, Ontologies (2008), pp. 162-184, Springer, 2008.
[11] Gordon, T. F.: The Carneades web service. In Computational Models of Argument – Proceedings of COMMA 2012, B. Verheij, S. Szeider, and S. Woltran, Eds., IOS Press, pp. 517-518.
[12] Hoekstra R., Breuker J., Di Bello M., Boer A.: The LKIF Core Ontology of Basic Legal Concepts. In: Casanovas P., Biasiotti M.A., Francesconi E., Sagri M.T. (eds.), Proceedings of LOAIT 2007, 2007.
[13] http://www.akomantoso.org/ naming convention of the URI
[14] Kelsen H.: Reine Rechtslehre, 2d. ed., Wien, 1960.
[15] Lam H., Governatori G.: The Making of SPINdle. RuleML 2009 proceeding, pp. 315-322, 2009.
[16] Lupo C., Vitali F., Francesconi E., Palmirani M., Winkels R., de Maat E., Boer A., and Mascellani P: General xml format(s) for legal sources - Estrella European Project IST-2004-027655. Deliverable 3.1, Faculty of Law, University of Amsterdam, Amsterdam, The Netherlands, 2007.
[17] Mazzei A., Radicioni D., Brighi R.: NLP-based extraction of modificatory provisions semantics. ICAIL 2009: pp. 50-57, ACM, 2009.
[18] Mommers L.: Ontologies in the Legal Domain. In: Poli R., Seibt J. (eds.), Theory and Applications of Ontology: Philosophical Perspectives, Springer 2010, pp. 265-276, 2010.
[19] Palmirani M., Brighi R.: An XML Editor for Legal Information Management. Proceeding of the DEXA 2003, Workshop on E-Government, Praga, 1-5 September, pp. 421-429. Springer-Verlag Berlin Heidelberg, 2003.
[20] Palmirani M., Brighi R.: Model Regularity of Legal Language in Active Modifications. AICOL Workshops 2009: pp. 54-73, Springer, 2009.
[21] Palmirani M., Cervone L.: Legal Change Management with a Native XML Repository. A cura di G. Governatori. Legal Knowledge and Information Systems. JURIX 2009. The Twenty-Second Annual Conference. Rotterdam. 16th-18th December 2009,pp. 146-156, Amsterdam: ISO press, 2009.
[22] Palmirani M., Contissa G., Rubino R: Fill the Gap in the Legal Knowledge Modelling. In Proceedings of RuleML 2009, pp. 305-314, Springer, 2009.
[23] Palmirani M., Governatori G., Rotolo A., Tabet S., Boley H., Paschke A.: LegalRuleML: XML-Based Rules and Norms. RuleML America 2011: 298-312, Springer, 2011.
[24] Palmirani M., Governatori G., Rotolo A., Tabet S., Boley H., Paschke A.: Legal-RuleML: XML-Based Rules and Norms. RuleML America, Springer, 2011, pp.298-312.
[25] Palmirani M.: Legislative Change Management with Akoma-Ntoso, in Legislative XML for the Semantic Web, Springer, Law, Governance and Technology Series Volume 4, 2011, pp 101-130.
[26] Sartor G.: Legal Concepts as Inferential Nodes and Ontological Categories. In Artif. Intell. Law 17(3) 2009, pp. 217-251, 2009.
[27] Sartor G.: Legal Reasoning: A Cognitive Approach to the Law. Vol. 5. Treatise on Legal Philosophy and General Jurisprudence. Berlin: Springer, 2005.
[28] Vitali F., Palmirani M.: Akoma Ntoso Release Notes. [http://www.akomantoso.org]. Accessed 5 July 2013.