OASIS LegalRuleML

RuleML2013, Seattle
12th July
Outline

- Introduction to LegalRuleML
  - Motivations, Goals, Principles
  - Design principles
  - LegalRuleML Syntax

- Meta-Model of LegalRuleML (Tara Athan)

- Use Case: “Section 29 Consumer Code of Australia” (Guido Governatori)

- Use Case: Patent Law (Adrian Paschke)
Motivations

- Legal texts are the privileged sources for norms, guidelines and rules that often feed different concrete Web applications.
  - Legislative documents, Contracts, Judgements

- The ability to have proper and expressive conceptual, machine readable models of the various and multifaceted aspects of norms, guidelines, and general legal knowledge is a key factor for the development and deployment of successful applications.
The LegalRuleML TC, set up inside of OASIS at Jan 12, 2012 (www.oasis-open.org) with 25 members, aims to produce a rule language for the legal domain:

- Based on the legal sources (text, pictures, etc.) of norms
- Oriented to legal people
- Compact in the syntax annotation
- Neutral respect any logic
- Flexible and extensible
State of the art and background

- RuleML
  - RuleML doesn’t manage temporal metadata, penalty-reparation, temporal defeasibility
- LKIF-rule [Gordon 2008]
  - LKIF-rule doesn’t implement the temporal metadata, specific deontic operators, temporal defeasibility, penalty-reparations
- RIF
  - RIF does not provide direct support for adequate representation of legal rules and legal reasoning. The current RIF dialects are not expressive enough, since they do not support e.g. logic-based negation, non-monotonic reasoning, events and temporal metadata etc.
RuleML Family of Sublanguages
Requirements

- Support for modelling different types of rules:
  - Constitutive rules (e.g. definitions)
  - Prescriptive rules (e.g. obligation, permission, etc.)
  - Other type of statements:
    - Penalty
    - Reparation
    - Override
    - Facts
- Implement isomorphism [Bench-Capon and Coenen, 1992]
- Model legal procedural rules
Design Principles (1/2)

Multiple Semantic Annotations:

- A legal rule may have multiple semantic annotations where each annotation can represent a different legal interpretation.
- Each such annotation can appear in a separate annotation block as internal or external metadata.

Tracking the LegalRuleML Creators:

- As part of the provenance information, a LegalRuleML document or any of its fragments can be associated with its creators.

Linking Rules and Provisions:

- LegalRuleML includes a mechanism, based on IRI, that allows N:M relationships among the rules and the textual provisions
  - avoiding redundancy in the IRI definition and errors in the associations
  - LegalRuleML is independent respect any Legal Document XML standard, IRI naming convention
Design Principles (2/2)

Temporal Management:
- Provisions, references, rules, applications of rules and physical entities change in time, and their histories interact in complicated ways. LegalRuleML must represent these temporal issues in unambiguous fashion.

Formal Ontology Reference:
- LegalRuleML is independent from any legal ontology and logic framework. It includes a mechanism, based on IRIs, for pointing to reusable classes of a specified external ontology.

LegalRuleML is based on RuleML:
- LegalRuleML reuses and extends concepts and syntax of RuleML wherever possible, and also adds novel annotations. RuleML includes also Reaction RuleML.

Mapping:
- Investigate the mapping of LegalRuleML metadata to RDF triples for favouring Linked Data reuse.
Open Document, Open Rules, Open Data

Legal document in XML

Legal Ontology

Logic Rules

Combine rules with other dataset
Interoperability and interchange
Retrieve rules and documents

Linked Open Data

ENGINE
LegalRulML Approach

Metadata of Context

Digital Millennium Copyright Act
NEW VERSION

2013

Rules as interpretation of the text

Metadata of Context T2
Outcome of the LegalRuleML TC

- Two formats: compact and normal
- XSD
- RelaxNG
- Metamodel in RDFs
- Example
- Glossary
- Documentation
  http://sinatra.cirfsid.unibo.it/XSDocViewer/
LegalRuleML main blocks

Metadata
  Legal Sources
  References
  Agents
  Authority
  Time Instants
  Temporal Characteristics
  Jurisdiction
  Role

Context association of metadata with rules

Context different author association of metadata with rules

Context different time and jurisdiction association of metadata with rules

<lrml:Penalty key="rule1">
  <lrml:if> ...</lrml:if>
  <lrml:then>... </lrml:then>
</lrml:Penalty>...
Textual References

Rule Context
parameters like agents, times, sources

Association
between Text and Rules
N:M relationship

Rules

Document Structure:
Metadata, Contexts, Rulebases

<lrml:LegalRuleML>
  <lrml:References>
    <Reference> ...
    </lrml:References>
  ...
  <lrml:Context key="ruleInfo1-v2">
    <lrml:Association>
      <lrml:appliesSource keyref="#sec2.1-list1-itm31-par1-v2"/>
      <lrml:toTarget keyref="#rulebase1-v2"/>
    </lrml:Association>
  </lrml:Context>
  <lrml:hasStatements key="rulebase-v2">
    <lrml:ConstitutiveStatement key="rule1a-v2">
      <ruleml:Rule>
        <ruleml:if> ...
        <ruleml:then>...
      </ruleml:Rule>
    </lrml:ConstitutiveStatement>
  </lrml:hasStatements>
</lrml:LegalRuleML>
Normal and Compact version

- Meta-model is built on the RDF principles
- Nodes and Edges define the relationships among <subject, predicate, object>

```
<lrml:hasStatement>
  <lrml:ConstitutiveStatement key="cs1">
    <lrml:hasTemplate>
      <ruleml:Rule key=":ruletemplate1" closure="universal">
    </ruleml:Rule>
  </lrml:ConstitutiveStatement>
</lrml:hasStatement>
```

```
<lrml:ConstitutiveStatement key="cs1">
  <ruleml:Rule key=":ruletemplate1" closure="universal">
```

NORMAL

COMPACT
LegalRuleML main blocks

Metadata
- Legal Sources
- References
- Agents
- Authority
- Time Instants
- Temporal Characteristics
- Jurisdiction
- Role

Context

association of metadata with rules

<lrml:Penalty key="rule1">
    <lrml:if> ... </lrml:if>
    <lrml:then> ... </lrml:then>
</lrml:Penalty>...
<lrml:LegalSources>
  <lrml:LegalSource key="ref1"
    sameAs="http://www.law.cornell.edu/uscode/text/17/504#psection-1"/>
</lrml:LegalSources>

<lrml:References>
  <lrml:Reference refersTo="ref2"
    refID="/us/USCode/eng@/main#title17-sec504-clsc-pnt1" refIDSystemName="AkomaNtoso2.0-2012-10"/>
</lrml:References>
Agents and Authorities

<lrml:Agents>

    <lrml:Agent key="aut1"
    sameAs="&unibo;/person.owl#m.palmirani"/>
    <lrml:Agent key="aut2"
    sameAs="&unibo;/person.owl#g.governatori"/>

</lrml:Agents>

<lrml:Authorities>

    <lrml:Authority key="congress"
    sameAs="&unibo;/org.owl#congress">
    <lrml:type iri="&lrmlv;Legislature"/>
    </lrml:Authority>

</lrml:Authorities>

Agent - an entity that acts or has the capability to act.

Authority - any body with the power to create, endorse, or enforce legal norms.
Temporal Events and Temporal Situations

Event that define the validity of the rules

Type of event: In force Efficacy
LegalRuleML main blocks

Metadata
  Legal Sources
  References
  Agents
  Authority
  Time Instants
  Temporal Characteristics
  Jurisdiction
  Role

Context

association of metadata with rules

<lrml:Penalty key="rule1">
  <lrml:if> ...</lrml:if>
  <lrml:then>... </lrml:then>
</lrml:Penalty>...
The **Association** construct implements the association between metadata and rules:

- N-arity relationship without redundancy
- Fine granularity
applies relationship: Jurisdiction and Role

```xml
<lrml:Association>
  <lrml:appliesJurisdiction keyref="&jurisdictions;us"/>
  <lrml:appliesRole>
    <lrml:Role iri="&lrmlv;#Author">
      <lrml:filledBy keyref="#aut1"/>
    </lrml:Role>
  </lrml:appliesRole>
  <lrml:toTarget keyref="#rule1"/>
</lrml:Association>
```
<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="#sec504-clsc-pnt1"/>
  <lrml:toStatement keyref="#rule1"/>
</lrml:Context>
LegalRuleML main blocks

Metadata
 Legal Sources
 References
 Agents
 Authority
 Time Instants
 Temporal Characteristics
 Jurisdiction
 Role

Context
 association of metadata with rules

<lrml:Penalty key="rule1">
  <lrml:if> ...</lrml:if>
  <lrml:then>... </lrml:then>
</lrml:Penalty>...
Deontic operators

- Obligation, Right, Permission, Prohibition, etc.

  <lrml:Prohibition key="prh2">
  <lrml:Obligation key="ob1">
  <lrml:Permission key="per1">
  <lrml:Right key="rgh1">
  <lrml:Compliance key="cmp1">
  <lrml:Violation key="vlt1">

- Penalty, Reparation, Behaviors
Deontic operators

**Bearer** - an entity that to which the deontic specification is primarily directed.

\[
\begin{array}{c}
\text{lrml:Right} \\
\text{lrml:slot} \\
\text{lrml:Bearer iri="&deontic-ontology;#oblbsub1"/}
\end{array}
\]

\[
\begin{array}{c}
\text{lrml:Var}X
\end{array}
\]

\[
\begin{array}{c}
\text{lrml:slot} \\
\text{lrml:AuxiliaryParty iri="&deontic-ontology;#oblbAdd1"/}
\end{array}
\]

\[
\begin{array}{c}
\text{lrml:Var}Y
\end{array}
\]

**AuxiliaryParty** - an entity in addition to the bearer of a deontic specification.

\[
\begin{array}{c}
\text{lrml:Atom} \\
\text{lrml:Rel iri="#copyright"/}
\end{array}
\]

\[
\begin{array}{c}
\text{lrml:Var}X
\end{array}
\]

\[
\begin{array}{c}
\text{lrml:Var}book
\end{array}
\]

\[
\begin{array}{c}
\text{lrml:Atom}
\end{array}
\]

\[
\begin{array}{c}
\text{lrml:Right}
\end{array}
\]
Penalty

Set of obligations/rights

<lrml:PenaltyStatement key="pen1">
  <lrml:SuborderList key="behav1">
    <lrml:Obligation key="oblig101">
    </lrml:Obligation>
    <lrml:Obligation key="oblig102">……
    </lrml:Obligation>
    <lrml:Obligation key="oblig103">……
    </lrml:Obligation>
  </lrml:SuborderList>
</lrml:PenaltyStatement>

(¬A => B)  
(¬B => C)  
(¬C => D)  

Obligation101

→ Obligation102

→ Obligation103
<lrml:ReparationStatement key="rep1">-
  <lrml:Reparation key="assoc1">
    <lrml:appliesPenalty keyref="#pen1"/>
    <lrml:toPrescriptiveStatement keyref="#ps1"/>
  </lrml:Reparation>
</lrml:ReparationStatement>
### Defeasibility

<table>
<thead>
<tr>
<th>Condition</th>
<th>Rule</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body always head</td>
<td>Body -&gt; head</td>
<td>strict</td>
</tr>
<tr>
<td>Body sometimes head</td>
<td>Body =&gt; head</td>
<td>defeasible</td>
</tr>
<tr>
<td>Body not complement head</td>
<td>Body &gt; head</td>
<td>defeater</td>
</tr>
</tbody>
</table>

R2 > R1

```xml
<lrml:OverridesStatement>
  <lrml:Overrides under="#ps1" over="#ps2"/>
</lrml:OverridesStatement>
```
Defeasibility qualification

1.1 In the Context block

```xml
<lrml:Context key="ruleInfo1">
  <lrml:appliesStrength>
    <lrml:Defeasible/>
  </lrml:appliesStrength>
  <lrml:toStatement keyref="#cs1"/>
</lrml:Context>
```

1.2

```xml
<lrml:appliesStrength iri="&defeasible-ontology;#defeasible"/>
```

2 Inline in the Rule

```xml
<lrml:hasStrength>
  <lrml:Defeasible key="str1" iri="&defeasible-ontology;#defeasible1"/>
</lrml:hasStrength>
```
<lrml:hasStatement>
  <lrml:FactualStatement key="fact1">
    <lrml:hasTemplate>
      <ruleml:Atom key=":atom11">
        <ruleml:Rel iri="#rel5"/>
        <ruleml:Ind iri="#JohnDoe"/>
      </ruleml:Atom>
    </lrml:hasTemplate>
  </lrml:FactualStatement>
</lrml:hasStatement>
Where to find material of the tutorial

- Examples SVN: https://tools.oasis-open.org/version-control/browse/wsvn/legalruleml/trunk/?rev=77&sc=1
- Documentation of the LegalRuleML TC: https://www.oasis-open.org/committees/tc_home.php?wg_abbr_ev=legalruleml
- lrml navigable technical documentation: http://sinatra.cirsfid.unibo.it/XSDocViewer/
Thank you for your attention!
and joint to LegalRuleML TC
Questions?
monica.palmirani@unibo.it