Highly-Modular Relax NG Schemas for Customized Access to RuleML Knowledge Bases

Tara Athan¹, Harold Boley²

¹Athan Services, W. Lafayette, Indiana

²Institute for Information Technology, National Research Council; Faculty of Computer Science, University of New Brunswick, Canada

OMG Technical Meeting, Ontology PSIG
Cambridge, MA, 21 June 2012
Re-conceptualization and Re-engineering: Goals

- **Language Extensions**
  - Decreased positional sensitivity
  - More flexibility in defining sublanguages

- **Greater Reliability**

- **Greater Automation**
  - Testing, documentation, conversion
Relationship of RNC and XSD: Syntactic Inclusion

- **Relaxed Serialization (RNC)**
  - More positional independence
- **Original Serialization (XSD)**
  - Optional Stripes
  - Some positional independence
- **Normal Serialization (RNC)**
  - Fully-striped
  - Canonical Position
Modularization:
“Original Fifteen” (non-SWSL)

- RuleML XSDs use directed tree-based modularization
- RuleML Relax NG uses lattices
- Lattice vertices can be assigned codes
  - Bitwise-dominance indicates containment
    \[ 1111 = 001111 < 101111 \]
Modularization: Original Fifteen FOL

- RuleML XSDs use directed tree-based modularization
- RuleML Relax NG uses lattices
- Lattice vertices can be assigned codes
  - Bitwise-dominance indicates containment
    \[1111 = 001111 < 101111\]
Selection Form

Instructions

Make a selection from the form below, then click "Refresh Schema" to update the Schema URL. The main module is also displayed below the form. To reset the form to the default (supremum) values, click "Reset Form".

Reset Form  Refresh Schema

Schema URL = http://ruleml.org/1.0/relaxng/schema_rnc.php?backbone=x3f&default=x7&termseq=x7&lng=x1&propo=x3ff&implies=x7&terms=xf3f&quant=x7&expr=xf&serial=xf
### MYNG Customization Form Part 1

<table>
<thead>
<tr>
<th>Expressivity</th>
<th>Treatment of Attributes With Default Values (Check One)</th>
<th>Term Sequences: Number of Terms (Check One)</th>
<th>Language (Check One)</th>
<th>Serialization Options (Check Zero or More)</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Backbone&quot;</td>
<td>Atomic Formulas</td>
<td>English Abbreviated Names</td>
<td>Unordered Groups</td>
<td></td>
</tr>
<tr>
<td>(Check One)</td>
<td>Ground Fact</td>
<td>None</td>
<td>Stripe-Skipping</td>
<td></td>
</tr>
<tr>
<td>Atomic Formulas</td>
<td>Ground Fact</td>
<td>Binary (Zero or Two)</td>
<td>Explicit Datatyping</td>
<td></td>
</tr>
<tr>
<td>Ground Fact</td>
<td>Ground Fact</td>
<td>Polyadic (Zero or More)</td>
<td>Schema Location</td>
<td></td>
</tr>
<tr>
<td>Fact</td>
<td>Datalog</td>
<td></td>
<td>Attribute</td>
<td></td>
</tr>
<tr>
<td>Ground Fact</td>
<td>Horn Logic</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Datalog</td>
<td>Logic</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Horn Logic</td>
<td>Disjunctive</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Disjunctive</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### MYNG Customization Form Part 2

<table>
<thead>
<tr>
<th>Propositional Options (Check Zero or More)</th>
<th>Implication Options (Check Zero or More)</th>
<th>Term Options (Check Zero or More)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRIs</td>
<td>Equivalences</td>
<td>Object Identifiers</td>
</tr>
<tr>
<td>Rulebases</td>
<td>Inference Direction</td>
<td>Slots</td>
</tr>
<tr>
<td>Entailments</td>
<td></td>
<td>Slot Cardinality</td>
</tr>
<tr>
<td>Degree of Uncertainty</td>
<td></td>
<td>Slot Weight</td>
</tr>
<tr>
<td>Strong Negation</td>
<td></td>
<td>Equations</td>
</tr>
<tr>
<td>Weak Negation</td>
<td></td>
<td>Oriented Equations</td>
</tr>
<tr>
<td>Non-Material</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quantification Options (Check Zero or More)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Implicit Closure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Generalized Lists</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set-valued Expressions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interpreted Expressions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expression Options (Check Zero or More)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Usage

The Schema URL may be used directly for online validation - copy and paste as required by the validator. For a demonstration of validation using the online service Validator.nu, see How to Validate with the RuleML Parameterized Relax NG Schema. Some scripts and processing instructions may require that the character "&" be replaced by "&amp;". Clicking on the Schema URL downloads a copy of the schema driver into a file named "custom_driver.rnc". To use the schema driver locally (offline), a local copy of the modules directory is also necessary - this may be downloaded as a zip archive from the RuleML 1.0 Relax NG Directory.
start = Node.choice | edge.choice
#
# ROOT NODE AND PERFORMATIVES INCLUDED
#
include "modules/performative_expansion_module.rnc" inherit = ruleml {start |!= notAllowed}
#
# ATOMIC FORMULAS INCLUDED
#
include "modules/atom_expansion_module.rnc" inherit = ruleml {start |!= notAllowed}
#
# CONJUNCTIONS AND DISJUNCTIONS INCLUDED
#
include "modules/andor_expansion_module.rnc" inherit = ruleml {start |!= notAllowed}
#
# IMPLICATIONS INCLUDED
#
include "modules/implication_expansion_module.rnc" inherit = ruleml {start |!= notAllowed}
#
# QUANTIFICATION OVER VARIABLES INCLUDED
#
include "modules/quantification_expansion_module.rnc" inherit = ruleml {start |!= notAllowed}
#
# ATTRIBUTES WITH DEFAULT VALUES ARE INITIALIZED
#
include "modules/default_inf_expansion_module.rnc" inherit = ruleml {start |!= notAllowed}
#
# ATTRIBUTES WITH DEFAULT VALUES ARE ABSENT OR OPTIONAL

18/06/12 RuleML in Relax NG Athan, Boley 10/16
RNC as Content Model

- **XSD**

```xml
<xsd:element name="RuleML">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element minOccurs="0" ref="ruleml:oid"/>
      <xsd:choice minOccurs="0" maxOccurs="unbounded">
        <xsd:element ref="ruleml:act"/>
        <xsd:element ref="ruleml:Assert"/>
        <xsd:element ref="ruleml:Retract"/>
        <xsd:element ref="ruleml:Query"/>
      </xsd:choice>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

- **RNC**

```plaintext
RuleML = element
RuleML { oid?,
         ( act
          | Assert
          | Retract
          | Query)* } 
```
Serializations Compared

- **RNC normal**

Atom = `element Atom { attribute closure { "universal" | "existential" }?, oid?, degree?, op, arg*, repo?, slot*, resl? }

- **RNC relaxed**

Atom = `element Atom { attribute closure { "universal" | "existential" }?, (oid? & degree?), ((op|Rel) & (arg|arg.content)* & repo? & slot* & resl?) }
Syntactic Monotonicity

• Definition:
  • Grammar containment implies syntactic containment

• Relax NG (like XSD) is not monotonic
  • redefinition
  • interleave combine “&=”

• $xy.rnc$
  start = x
  x = element x{ x.main }
  x.main = y?
  y = element y{ text }

• $xy_redefine.rnc$
  include $xy.rnc$
  { x.main = y+ }

• $xy_interleave.rnc$
  include $xy.rnc$
  x.main &= y
Schema Design Pattern: Sufficient to Achieve Monotonicity

- Segregated Names
  - Choice combine
  - No combine
  - Interleave combine
    - &= empty
    - &= ...
    - &= ...

- Joins by union, not redefinition

\[\text{Equal-node.choice} \models \text{Equal.Node.def}\]
\[\text{Equal.Node.def} = \]
\[\text{element Equal} \{
\text{(Equal-datt.choice} \& \text{reEqual.attlist), Equal.header, Equal.main}\}\]
\[\text{Equal.header} \&= \]
\[\text{SimpleFormula.header}?\]
\[\text{Equal.main} \models \]
\[\text{leftSide-edge.choice, rightSide-edge.choice}\]
Expressivity of Schema Design Pattern

- Any valid RNC schema can be expressed using the schema design pattern
- Any language lattice where each language has a valid RNC schema can be modularized using the schema design pattern

```
RuleML =
    element RuleML
    {...}
act =
    element act {...}
...
```
Measurable Outcomes: Increased customizability

- Over fifty freely combinable modules
  - Decoupling elements such as `<Atom>`
- More than $2^{50} > 10^{15}$ grammars
- generating an estimated 300,000 different (and meaningful) languages.