

The RuleML System of Families of Languages

T. Athan^{1,3} H. Boley² A. Paschke³

¹Athan Services (athant.com), West Lafayette, Indiana, USA

²Faculty of Computer Science, University of New Brunswick, Fredericton, Canada

³AG Corporate Semantic Web, Freie Universitaet Berlin, Germany

RuleML Symposium, Stony Brook University, 6-9 July 2016

Outline

- 1 System of Families
- 2 Deliberation RuleML
- 3 MYNG
- 4 Consumer RuleML
- 5 Reaction RuleML

The RuleML Families

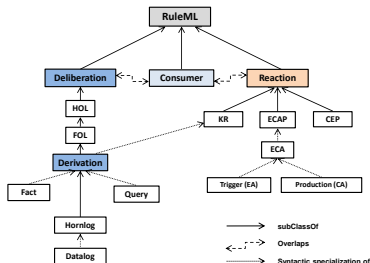


Figure: An overview of the subclass, syntactic specialization, and overlap relationships between the Deliberation, Reaction, and Consumer RuleML families of RuleML 1.02 is given in this diagram.

The RuleML Feature Set

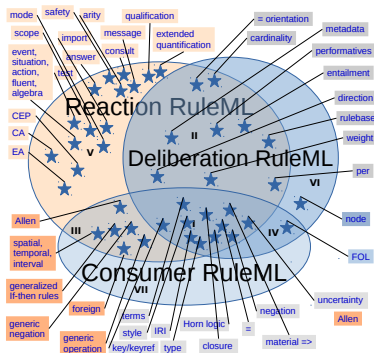


Figure: Details of the syntactic containment relationships between the Reaction, Deliberation and Consumer families are shown in this Venn diagram. The filled ellipses indicate sets of language *constructs*. The stars indicate language *features*, e.g. rules, first-order logic expressivity, or performatives, that may be combined to generate language constructs.

Revised Approach to RuleML Semantics

- RuleML prior to 1.02 assumed **predefined default semantics**,
 - which could be overridden by explicit semantic variant attributes
- Rather than distinguishing one semantics for each language, RuleML 1.02 assumes **no default semantics but by default keeps the semantics unspecified**

Advantages of the new RuleML Semantic System

Safety: semantics must be explicitly specified, avoiding incorrect assumptions

Refinability: semantics can be partially and gradually introduced

Scalability: syntactic extensions can be paired with semantic extensions

Domain Requirements: E.g.:

- Legal: multiple interpretations of legal texts
- Software specification: languages (e.g. UML) without complete formal semantics

Predefined Semantic Styles of RuleML 1.02

- Underspecified profiles (partially constrained)
 - splittableHeadConjunction
 - reasoning
 - active
 - messaging
- Fully-specified semantics
 - Horn-PSOA Tarski
 - Horn Logic Herbrand
 - First-Order Logic Herbrand
 - First-Order Deontic Alethic Logic
 - Reified Classical Situation Calculus

Features of Deliberation RuleML 1.x

- Highly-expressive First-Order logic, with variably polyadic functions and relations
- Flexible hybrid relational-graph syntax
- Oriented and unoriented equality
- Logic programming through weak negation
- Fuzzy and probabilistic logics with uncertainty degree and slot weights
- IRIs and CURIEs as names, in addition to local vocabulary
- Metadata annotations
- Extralogical performatives and metalogic

New Features in Deliberation RuleML 1.02

- Distributed representation (@key/@keyref newly adopted from Reaction RuleML)
- Increased semantic flexibility with references
 - Extended quantification (existing @closure attribute now takes an IRI or CURIE value)
 - Signatures (existing @type attribute now takes an IRI or CURIE value)
 - Semantic specification (@style is newly adopted from Reaction RuleML)

Example

```
<Assert style="rp:splittableHeadConjunction">
  <Implies>
    <Atom><Rel>Wound</Rel></Atom>      <!-- if -->
    <And>                                <!-- then -->
      <Atom><Rel>Bandage</Rel></Atom>
      <Atom><Rel>Medication</Rel></Atom>
    </And>
  </Implies>
</Assert>
```

Example

```
<Assert style="rp:splittableHeadConjunction">
  <Implies>
    <Atom key=":cond"><Rel>Wound</Rel></Atom>
    <Atom><Rel>Bandage</Rel></Atom>
  </Implies>
  <Implies>
    <Atom keyref=":cond"/>
    <Atom><Rel>Medication</Rel></Atom>
  </Implies>
</Assert>
```

MYNG 1.02 Overview

MYNG 1.02 - the Deliberation RuleML Schema Configuration Form

Instructions

To automatically construct a customized language, make selections from the form below. Click "Download ..." for the normative RNC schema or an approximating XSD anchor schema. To view the Relax NG driver schema, click "Generate Schema", then scroll down. To fill the form to the initial (supremum) values, click "Fill Form". To clear the form, specifying a language with the fewest included modules (infimum), click "Clear Form".



Fill Form **Clear Form** **Generate Schema** **Download RNC Schema**

Download XSD Anchor Schema

RNC: myng-b3f-d7-a7-l1-p3ff-l7f-tf3f-q7-ef-s4f

XSD: naffologeq

Figure: MYNG GUI for configuring a customized syntax containing selected features. Provides IRIs or download of the following:

- Precise schemas in Relax NG
- Lenient approximate (anchor) schemas in XSD.

MYNG 1.02 Facets

RNC: myng-b1f-d7-a7-11-p3ff-i7f-tf3f-q7-e1-s4f

XSD: nafnegdishornlogplus

| | | | | |
|--|---|---|---|---|
| <p>Expressivity "Backbone" (Select One)</p> <ul style="list-style-type: none"> <input type="radio"/> Atomic Formulas <input type="radio"/> Ground Fact <input type="radio"/> Ground Logic <input type="radio"/> Datalog <input checked="" type="radio"/> Horn Logic <input type="radio"/> Full First-Order Logic <p>Clear Others</p> <p>Fill Others</p> | <p>Propositional Options (Check Zero or More)</p> <p>Check All</p> <p>Uncheck All</p> <ul style="list-style-type: none"> <input checked="" type="checkbox"/> IRIs <input checked="" type="checkbox"/> Rulebases <input checked="" type="checkbox"/> Entailments <input checked="" type="checkbox"/> Degree of Uncertainty <input checked="" type="checkbox"/> Strong Negation <input checked="" type="checkbox"/> Weak Negation (Negation as Failure) <input checked="" type="checkbox"/> Node Identifiers <input checked="" type="checkbox"/> In-Place Annotation <input checked="" type="checkbox"/> XML base <input checked="" type="checkbox"/> XML id | <p>Implication Options (Check Zero or More)</p> <p>Check All</p> <p>Uncheck All</p> <ul style="list-style-type: none"> <input checked="" type="checkbox"/> Equivalences <input checked="" type="checkbox"/> Inference Direction <input checked="" type="checkbox"/> Non-Material <input checked="" type="checkbox"/> Conjunctive Heads <input checked="" type="checkbox"/> Negative Constraints <ul style="list-style-type: none"> <input checked="" type="checkbox"/> Disjunctive Heads <input checked="" type="checkbox"/> Existential Heads | <p>Term Sequences: Number of Terms (Select One)</p> <ul style="list-style-type: none"> <input type="radio"/> None <input type="radio"/> Unary (Zero or One) <input type="radio"/> Binary (Zero or Two) <input type="radio"/> Unary/Binary (Zero to Two) <input checked="" type="radio"/> Variably Polyadic (Zero or More) | <p>Term Options (Check Zero or More)</p> <p>Check All</p> <p>Uncheck All</p> <ul style="list-style-type: none"> <input checked="" type="checkbox"/> Object Identifiers <input checked="" type="checkbox"/> Slots <ul style="list-style-type: none"> <input checked="" type="checkbox"/> Slot Cardinality <input checked="" type="checkbox"/> Slot Weight <input checked="" type="checkbox"/> Equations <ul style="list-style-type: none"> <input checked="" type="checkbox"/> Oriented Equations <input checked="" type="checkbox"/> Term Typing <input checked="" type="checkbox"/> Data Terms <input checked="" type="checkbox"/> Skolem Constants <input checked="" type="checkbox"/> Reified Terms |
|--|---|---|---|---|

Figure: Fine-grained feature options grouped into related facets.

MYNG 1.02 Facets, continued

| | | | |
|--|--|---|---|
| <p>Quantification Options (Check Zero or More)</p> <p><input type="checkbox"/> Check All <input type="checkbox"/> Uncheck All</p> <ul style="list-style-type: none"> <input checked="" type="checkbox"/> Implicit Closure <input checked="" type="checkbox"/> Slotted Rest Variables <input checked="" type="checkbox"/> Positional Rest Variables | <p>Expression Options (Check Zero or More)</p> <p><input type="checkbox"/> Check All <input type="checkbox"/> Uncheck All</p> <ul style="list-style-type: none"> <input type="checkbox"/> Generalized Lists <input type="checkbox"/> Set-valued Expressions <input type="checkbox"/> Interpreted Expressions | <p>Serialization Options (Check Zero or More)</p> <p><input type="checkbox"/> Check All <input type="checkbox"/> Uncheck All</p> <ul style="list-style-type: none"> <input checked="" type="checkbox"/> Unordered Groups <input checked="" type="checkbox"/> Stripe-Skipping <input checked="" type="checkbox"/> Explicit Datatyping <input checked="" type="checkbox"/> Schema Location Attribute <input checked="" type="checkbox"/> Distributed Syntax | <p>Language (Select One)</p> <ul style="list-style-type: none"> <input checked="" type="radio"/> English Abbreviated Names <input type="radio"/> English Long Names <input type="radio"/> French Long Names |
|--|--|---|---|

Relax NG Schema

Relax NG Query String URL = http://deliberation.ruleml.org/1.02/relaxng/schema_rnc.php?backbone=x1f&default=x7&termseq=x7&lng=x1&propo=x3f&implies=x7f&terms=x7f&quant=x7&expr=x1&serial=x4f

Equivalent Relax NG myng-code URL = http://deliberation.ruleml.org/1.02/relaxng/myng-b1f-d7-a7-11-p3ff-i7f-tf3f-q7-e1-s4f_rnc

XSD Anchor Schema URL = <http://deliberation.ruleml.org/1.02/xsd/nafnegdishornlogplus.xsd>

Figure: Additional facets, followed by referencable IRIs.

MYNG 1.02 Usage

Usage

The RNC and XSD Schema URLs may be used directly for online validation - copy and paste as required by the validator. For a demonstration of RNC 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 "&".

Clicking on the "Download RNC Schema" button downloads a copy of the schema driver into a file named according to the text labeled "RNC". To use the schema driver locally (offline), a local copy of the [modules](#) directory is also necessary - for download instructions, see the [Deliberation RuleML 1.02 Relax NG Directory](#). For more information about the re-engineering of RuleML into Relax NG, which made this modularization possible, see the [MYNG](#) page on the RuleML Wiki.

Figure: IRIs may be used for online validation, or download.

MYNG 1.02 Schema

Schema Driver

```

# Call parameters
# GET parameter: backbone=x1f
# GET parameter: default=x1
# GET parameter: terms=x1f
# GET parameter: lng=x1
# GET parameter: prp=x1f
# GET parameter: impl=x1f
# GET parameter: terms=x1f
# GET parameter: quant=x1
# GET parameter: expr=x1
# GET parameter: serial=x1f
namespace dc = "http://purl.org/dc/elements/1.1/"
namespace dterms = "http://purl.org/dc/terms/"
default namespace rulenz = "http://ruleml.org/spec"

dcterms:title [ "Deliberation RuleML Custom-Built Schema" ]
dcterms:identifier [ "http://deliberation.ruleml.org/1.02/relaxng/schema_rnc.php?backbon"
dterms:isPartOf [ "http://deliberation.ruleml.org/1.02/spec" ]
dterms:creator [ "http://wiki.ruleml.org/index.php/User:Atchant" ]
dct:subject [ "Deliberation RuleML, custom-built" ]
dcterms:description [
  "custom-built main module for a Deliberation RuleML language."
]
dcterms:date [ "2016-07-02T07:27:19-04:00" ]
dcterms:language [ "en" ]

```

Figure: Relax NG driver schema displayed. Modular system depends on custom restriction of Relax NG schema language. Future work includes

- On-the-fly generation of precise XSD schemas,
- Determination of minimal validating schema for a RuleML instance document
- Online normalization and compactification

New Functionality in MYNG 1.02

RNC: myng-101-a7-a7-1-p001-01-0301-a7-af-64f

XSD: mathloges

| | | | | |
|--|---|---|--|---|
| <p>Expressivity "Backbone" (Select One)</p> <p><input type="radio"/> Atomic Formulas</p> <p><input type="radio"/> Ground Fact</p> <p><input type="radio"/> Ground Logic</p> <p><input type="radio"/> Datalog</p> <p><input type="radio"/> Hom Logic</p> <p><input checked="" type="radio"/> Full First-Order Logic</p> <p><input type="button" value="Clear Others"/></p> <p><input type="button" value="Fill Others"/></p> | <p>Propositional Options (Check Zero or More)</p> <p><input type="button" value="Check All"/> <input type="button" value="Uncheck All"/></p> <p><input checked="" type="checkbox"/> IRIs</p> <p><input checked="" type="checkbox"/> Rulebases</p> <p><input checked="" type="checkbox"/> Entailments</p> <p><input checked="" type="checkbox"/> Degree of Uncertainty</p> <p><input checked="" type="checkbox"/> Strong Negation</p> <p><input checked="" type="checkbox"/> Weak Negation (Negation as Failure)</p> <p><input checked="" type="checkbox"/> Node Identifiers</p> <p><input checked="" type="checkbox"/> In-Place Annotation</p> <p><input checked="" type="checkbox"/> XML base</p> <p><input checked="" type="checkbox"/> XML id</p> | <p>Implication Options (Check Zero or More)</p> <p><input type="button" value="Check All"/> <input type="button" value="Uncheck All"/></p> <p><input checked="" type="checkbox"/> Equivalences</p> <p><input checked="" type="checkbox"/> Inference Direction</p> <p><input checked="" type="checkbox"/> Non-Material</p> <p><input checked="" type="checkbox"/> Conjunctive Heads</p> <p><input checked="" type="checkbox"/> Negative Constraints</p> <p><input checked="" type="checkbox"/> Disjunctive Heads</p> <p><input checked="" type="checkbox"/> Existential Heads</p> | <p>Term Sequences: Number of Terms (Select One)</p> <p><input type="radio"/> None</p> <p><input checked="" type="radio"/> Unary (Zero or One)</p> <p><input type="radio"/> Binary (Zero or Two)</p> <p><input type="radio"/> Unary/Binary (Zero to Two)</p> <p><input checked="" type="checkbox"/> Variably Polyadic (Zero or More)</p> | <p>Term Options (Check Zero or More)</p> <p><input type="button" value="Check All"/> <input type="button" value="Uncheck All"/></p> <p><input checked="" type="checkbox"/> Object Identifiers</p> <p><input checked="" type="checkbox"/> Slots</p> <p><input checked="" type="checkbox"/> Slot Cardinality</p> <p><input checked="" type="checkbox"/> Slot Weight</p> <p><input checked="" type="checkbox"/> Equations</p> <p><input checked="" type="checkbox"/> Oriented Equations</p> <p><input checked="" type="checkbox"/> Term Typing</p> <p><input checked="" type="checkbox"/> Data Terms</p> <p><input checked="" type="checkbox"/> Skeleton Constants</p> <p><input checked="" type="checkbox"/> Reified Terms</p> |
|--|---|---|--|---|

Figure: New in MYNG 1.02:
Convenient buttons for
clearing/filling form and facets

Consumer RuleML 1.02 - Embedded

- Designed to be embedded into other languages - lacks extra-logical features:
 - containers (e.g., `<Rulebase>`)
 - performatives (e.g., `<Assert>`, `<Query>`)
 - Consumer , for structure and pragmatics, of containers and performatives from other languages, e.g. from FIPA, SOAP, or domain-specific XML languages such as LegalRuleML.

Consumer RuleML 1.02 - External Resources

- Consumes external resources, from other syntactic and semantic specifications, accessed by attributes linking to external definitions of
 - syntactic constraints (@type),
 - semantic profiles (@style)
 - extended quantifications (@closure).

The external resources may be, for example, definitions in controlled natural language (e.g., in mathematical English) or formal specifications expressed in Reaction RuleML.

Status of LegalRuleML

- LegalRuleML embeds and extends Consumer RuleML 1.02.
- Announcement on LegalRuleML is forthcoming.

Reaction RuleML 1.02 - Highlights

- Rules (Rule) generalizing Implies, e.g., Production (CA) rules, Trigger (EA) and Event-Condition-Action (ECA) rules, distributed rule-based Complex Event Processing (CEP)
- Definition language for interface, signature, scopes, semantic profiles, qualifications, quantification, and corresponding referencing attributes for scope, mode, safety, arity, cardinality, qualifications, quantification
- Generics dedicated to specialization of terms, formulas, operations, connectors with e.g. spatial, temporal, interval, modal types
- Further actions/performatives for semantic import, updates, answers, justifications, tests, etc.
- Message descriptors for agent-based knowledge interchange

Examples - Signature Definition and Generics

```

<signature> <!-- signature for TimeIntervalClass type -->
  <Time>
    <oid><Ind>TimeIntervalClass</Ind></oid>
    <slot><Ind>start</Ind><Var type="xs:datetime"/></slot>
    <slot><Ind>end</Ind><Var type="xs:datetime"/></slot>
  </Time>
</signature>

<!-- Time generic using defined TimeIntervalClass type -->
<Time key=":t1">
  <oid><Ind type="TimeIntervalClass">interval1</Ind></oid>
  <slot><Ind>start</Ind><Data>2011-03-21T10:00:00-04:00</Data></slot>
  <slot><Ind>end</Ind><Data>2012-03-21T10:00:00-04:00</Data></slot>
</Time>

<Neg> <!-- modal operation "it is not necessary that not" -->
<Operation type="rulemlv:AlethicOperator" iri="rulemlv:Necessary">
  <Neg><Time keyref=":t1"/></Neg>
</Operation>
</Neg>

```

Examples Performatives / Actions

```
<!-- semantically import knowledge base-->
<do> <Consult iri="../../kb.rml"/> </do>

<!-- consult enclosed message content -->
<do> <Consult>
  <enclosed><Message> ... </Message></enclosed>
</Consult></do>

<!-- send and receive message -->
<do><Send><Message> ... </Message></Send></do>
<do><Receive><Message> ... </Message></Receive></do>

<!-- preform test with test suite -->
<do>
  <Test>
    <vvi><TestSuite> ... </TestSuite></vvi>
  </Test>
</do>
```

Conclusion

- Release of Consumer RuleML 1.02 in 2015 and contribution to OASIS LegalRuleML
- Recent release of Deliberation RuleML 1.02
- Releases of Reaction RuleML 1.02 and RuleML 1.02 as a whole (including semantics) are forthcoming.
- See <http://ruleml.org> for more details.