

The RuleML 0.87 Release

UML Model, Validation Stability, and Abridged Syntax

David Hirtle

Co-op student, NRC IIT e-Business

August 12, 2004

Overview

- Introduction
 - example
- RuleML 0.87
 - background
 - UML model
 - type/role distinction
 - slot changes
 - stripe-skipping
 - validation stability
 - demo
- Future Work

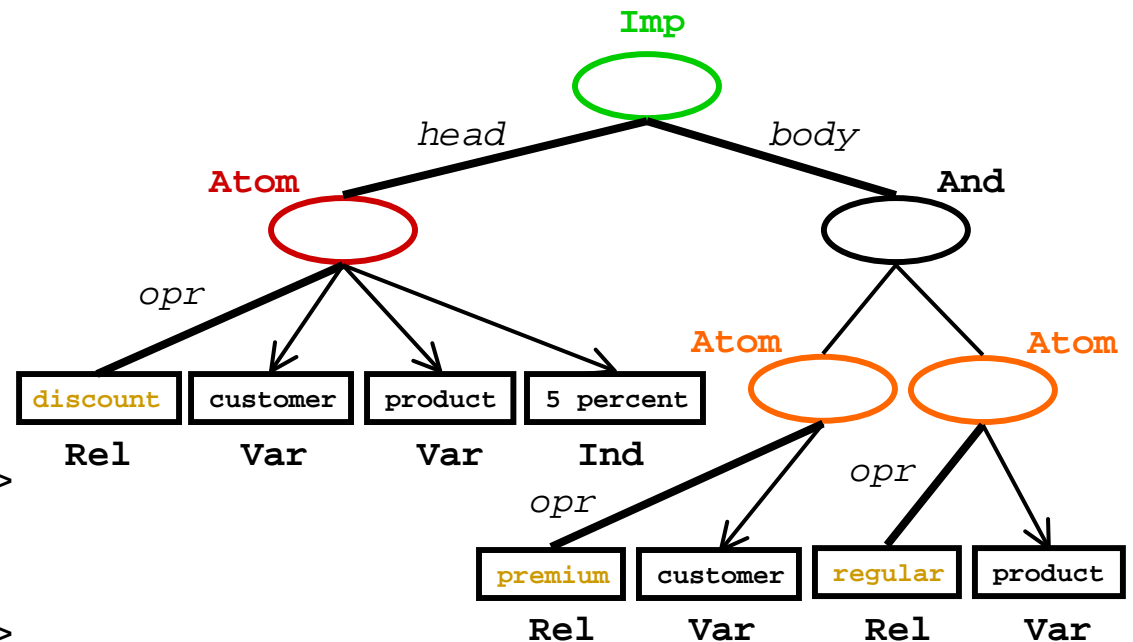
Introduction

- rules are essential for the Semantic Web
 - derivation rules (implicational-inference)
 - reaction rules (event-condition-action)
 - transformation rules (functional-equational)
- rule interchange is important for e-Business
- Rule Markup Initiative's goal is a canonical language (RuleML) for interoperable rule markup
 - XSLT translators to other SW languages, e.g. RDF
- collaborating with W3C and other standards bodies

Introduction – Example

"The **discount** for a *customer* buying a *product* is **5 percent** if the *customer* is **premium** and the *product* is **regular**."

```
<Imp>
  <head>
    <Atom>
      <opr><Rel>discount</Rel></opr>
      <Var>customer</Var>
      <Var>product</Var>
      <Ind>5.0 percent</Ind>
    </Atom>
  </head>
  <body>
    <And>
      <Atom>
        <opr><Rel>premium</Rel></opr>
        <Var>customer</Var>
      </Atom>
      <Atom>
        <opr><Rel>regular</Rel></opr>
        <Var>product</Var>
      </Atom>
    </And>
  </body>
</Imp>
```

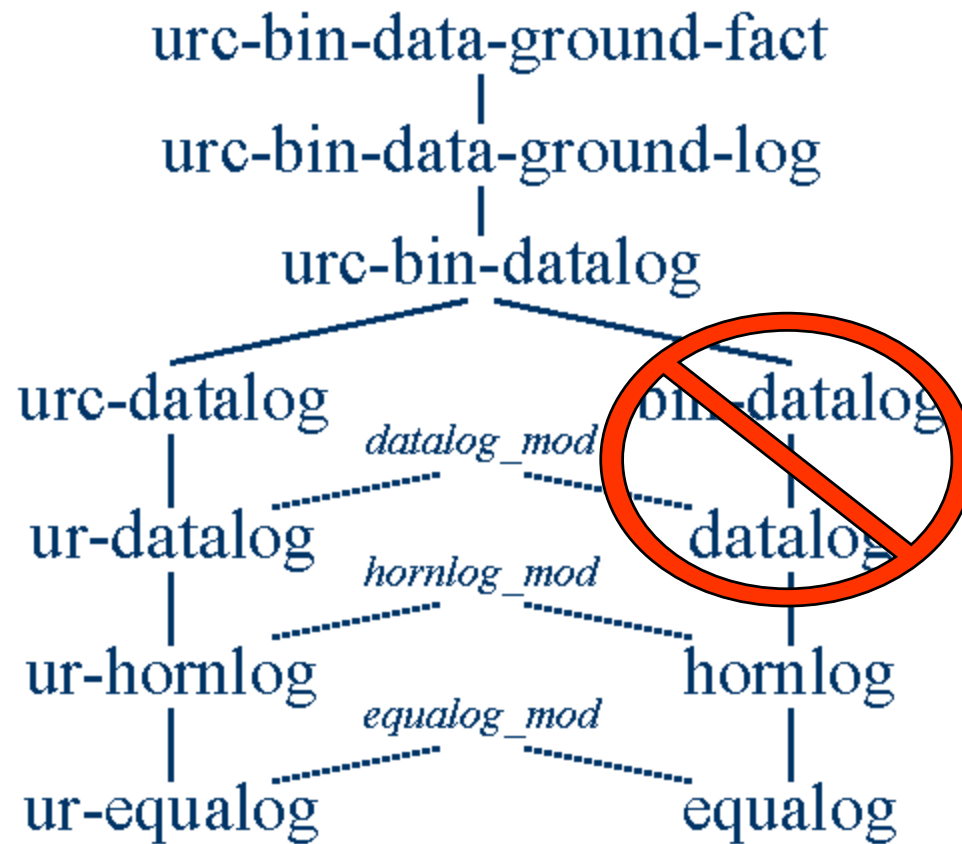


RuleML 0.87

- release announced today
- full specification: www.ruleml.org/0.87
 - XML Schemas: www.ruleml.org/0.87/xsd
 - Examples: www.ruleml.org/0.87/exa
 - Stylesheet: www.ruleml.org/0.87/xslt
- highlights
 - UML model for system of sublanguages
 - new type/role tag distinction
 - slot changes
 - “stripe-skipping” syntax
 - validation stability

RuleML 0.87 - Background

- transition from DTDs to XML Schema was problematic
 - many issues were resolved
 - then modularization in 0.85 revealed a new one...
- W3C XML Schema expressiveness gap
 - cannot extend ranges by decreasing lowerbound
 - impossible to go from binary to “zero or more” arguments
 - e.g. from binary datalog sublanguage to regular datalog



Rooted DAG will be extended with branches for further sublanguages

(version 0.85)

ruleml

[www.ruleml.org/0.85/dtd/Inheritance_diagram.gif]

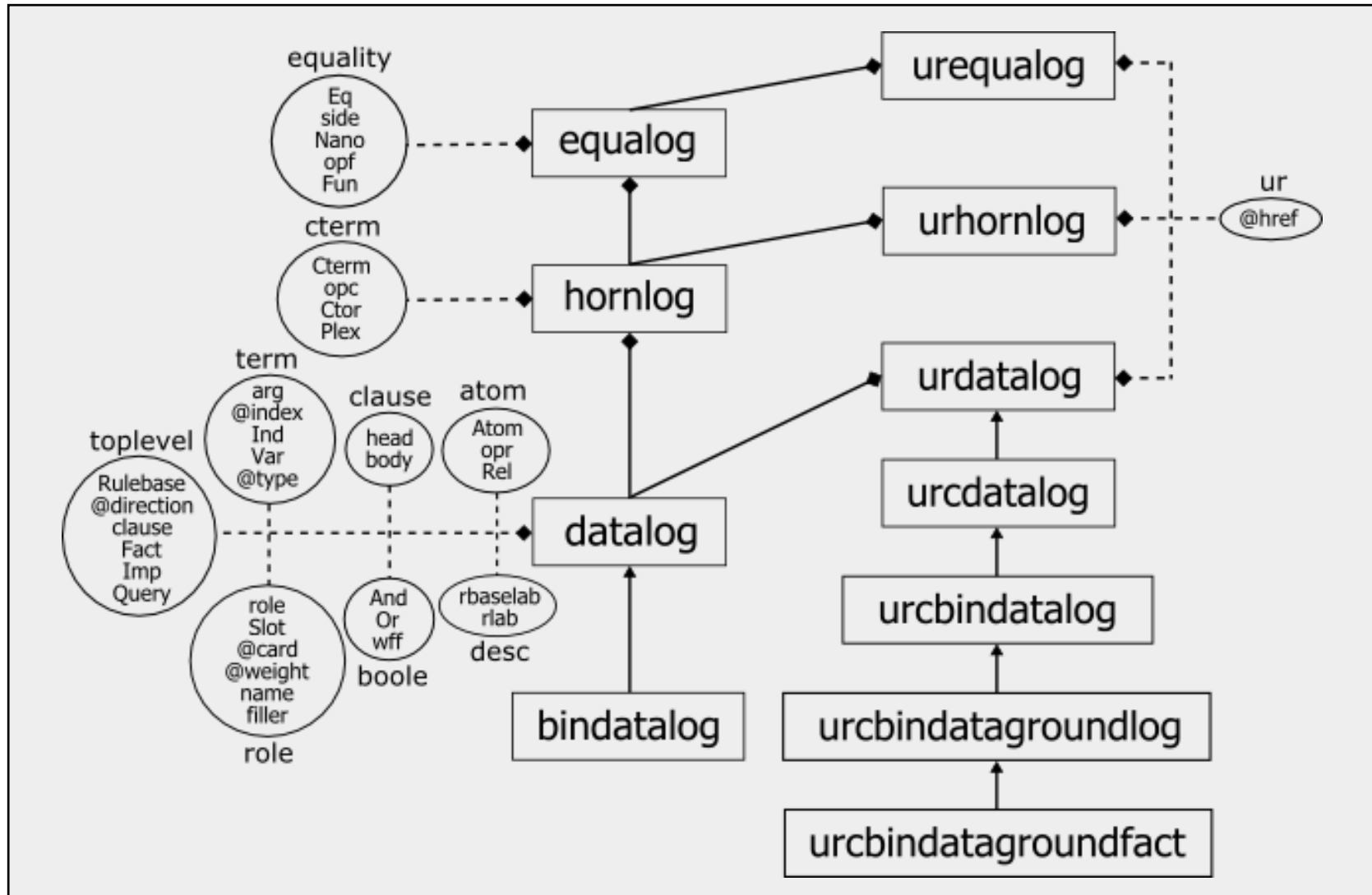
RuleML 0.87 – Background cont'd

- transition from DTDs to XML Schema was problematic
 - many issues were resolved
 - then modularization in 0.85 revealed a new one...
- W3C XML Schema expressiveness gap
 - cannot extend ranges by decreasing lowerbound
 - impossible to go from binary to “zero or more”
 - e.g. from binary datalog sublanguage to regular datalog
- evaluated three alternative versions of modularization
 - see www.ruleml.org/modularization for details
 - the “winner” was further refined and represented in UML

RuleML 0.87 - UML Model

- graphical conventions:
 - rectangle - schema drivers (actual sublanguages)
 - oval - elementary modules
 - UML-like aggregation arrows
 - e.g. datalog is part of hornlog
 - UML-like inheritance arrows
 - e.g. bindatalog is a datalog
- element and attribute definitions grouped into modules
 - not intended to be directly validated against
 - allow others to “borrow” specific parts of RuleML
- top-down expressiveness ordering

[www.ruleml.org/modularization/ruleml_m12n_uml.png]



RuleML 0.87 - Type/ Role Distinction

- role tags distinguished with “_” prefix since 0.8
 - e.g. role tags: `_head`, `_opr`
 - type tags: `imp`, `var`
- now switching to Java-style case convention
 - role tags begin with lowercase letter e.g. `head`, `opr`
 - type tags begin with uppercase letter e.g. `Imp`, `Var`
- XSLT stylesheet automatically upgrades 0.86 to 0.87
 - see [comparison of input and output files](#) using [HTMLDiff](#)

RuleML 0.87 - Slot Changes

- to accommodate F-logic...
 - slot names as subelements, not attributes
 - e.g. `<slot name="instrument"><Ind>bass</Ind></slot>`
becomes
`<slot><Ind>instrument</Ind><Ind>bass</Ind></slot>`
 - variables and complex terms as slot names
 - e.g. `<slot><Var>Property</Var><Ind>bass</Ind></slot>`
 - (**Property** can be bound to **instrument** or **fish**)
- slot evolves from a role to a type
 - so with new naming convention:

`<slot><Var>Property</Var><Ind>bass</Ind></slot>`

RuleML 0.87 - Stripe-Skipping

- alternating type/role tags called “striped syntax”
 - e.g. <Imp><head><Atom><opr><Rel>
- result is quite verbose
 - but important for compatibility with OO modeling and RDF
- “Stripe-Skipping” to the rescue
 - cf. [RuleML 0.8](#) and Sandro Hawke, W3C: [StripeSkipping](#)
 - role tags become optional (can be “reconstructed” anyway)
 - result is (combinable) compact and expanded forms

RuleML 0.87 - Stripe-Skipping (2)

Compact:

```
<Imp>
  <head>
    <Atom>
      <Rel>discount</Rel>
      <Var>customer</Var>
      <Var>product</Var>
      <Ind>5.0 percent</Ind>
    </Atom>
  </head>
  <body>
    <And>
      <Atom>
        <Rel>premium</Rel>
        <Var>customer</Var>
      </Atom>
      <Atom>
        <Rel>regular</Rel>
        <Var>product</Var>
      </Atom>
    </And>
  </body>
</Imp>
```

Expanded:

```
<Imp>
  <head>
    <Atom>
      <opr><Rel>discount</Rel></opr>
      <arg index="1"><Var>customer</Var></arg>
      <arg index="2"><Var>product</Var></arg>
      <arg index="3"><Ind>5.0 percent</Ind></arg>
    </Atom>
  </head>
  <body>
    <And>
      <Atom>
        <opr><Rel>premium</Rel></opr>
        <arg index="1"><Var>customer</Var></arg>
      </Atom>
      <Atom>
        <opr><Rel>regular</Rel></opr>
        <arg index="1"><Var>product</Var></arg>
      </Atom>
    </And>
  </body>
</Imp>
```

RuleML 0.87 - Validation Stability

- emphasized in 0.86 and 0.87, but ...
- [XML Schema spec](#) is complex and difficult to implement
 - current tools (e.g. validators) can be misleading
 - “I suspect it is true that there is no single schema processor which correctly enforces all the constraints defined in the schema specification. ... Certainly the fact that it gets through XML Spy (or any other product) is no proof of validity...”
Michael Kay, [xmlschema-dev@w3.org mailing list](mailto:xmlschema-dev@w3.org)
- modularity of RuleML XSDs adds to the challenge
 - discovered (and reported) issues with several tools, e.g. XML Spy
 - useful as a benchmark for finding validator bugs

RuleML 0.87 - Validation Stability (2)

- determining if RuleML XSDs are valid is not trivial
- our approach: use a variety of validators
 - [W3C XML Schema Validator \(XSV\)](#) (stable, free, online)
 - [Altova XML Spy](#)
 - [Saxon-SA](#)
 - [Microsoft XML Core Services \(MSXML\)](#)
 - [Xerces2 Java Parser \(Xerces-J\)](#) } via [Stylus Studio](#)
- see www.ruleml.org/0.87/#Validation for full results

RuleML 0.87 - Demo

[W3C XML Schema Validator \(XSV\)](#)

Steering Committee

- presented to RuleML Steering Committee during teleconference
 - Monday, August 9, 2004 1:00pm ADT
- Committee members:
 - Asaf Adi (IL)
 - Harold Boley (CA)
 - Mike Dean (USA)
 - Andreas Eberhart (DE)
 - Benjamin Grosf (USA)
 - Michael Kifer (USA)
 - Steve Ross-Talbot (UK)
 - Bruce Spencer (CA)
 - Said Tabet (USA)
 - Gerd Wagner (NL)
- work was approved

Future Work

- fully compact and fully expanded role tag normal forms
 - possible XSLT between these and earlier “mixed” form
- first-order logic extensions
- reaction rules, transformation rules
- abstract syntax
- glossary of terms
- guarded Horn Logic
 - suggested by Wolfgang Nejdl, U Hannover
- CLP (Corteous Logic Programs) overrides facts
 - work by Benjamin Grosf, MIT

Questions/ Comments?