Bridging Research and Practice – Making Real-World Application of Ontologies and Rules

- Lecture Notes -

Harold Boley
Holger Knublauch
Adrian Paschke

RuleML 2006 Tutorial

Athens, GA, 11 Nov 2006

Who are we?

- Harold Boley (NRC, Canada)
  - Leader of Semantic Web Laboratory
  - Work on RFML, RuleML, RIF, POSL, FindXpRT, and OO jDREW
- Holger Knublauch (TopQuadrant, USA)
  - Technical Director of Product Development
  - Developer of Protégé-OWL and TopBraid Composer
- Adrian Paschke (TU Munich, Germany)
  - Co-Chair of Reaction RuleML Technical Group
  - Work on PROVA and Contract Management for rule-based Service Level Agreements (RBSLA)

Agenda

- HB: Introduction (10 minutes)
- HB: RuleML Language Family (30 minutes)
  - Modularization, Interpreted functions, HiLog, Modals, ICs
- AP: Reaction RuleML (30 minutes)
  - General, orthogonal ECAP design, 0.1 Release
- *Break (10 minutes)*
- HK: Development Environments (30 minutes)
  - Protégé (with SWRL, Jess), TopBraid Composer (Jena)
- HK: Demo (30 minutes)
  - realEstate Use Case & Solution Archetypes
- HK: Wrap-up (10 minutes)

Introduction to Making Real-World Application of Ontologies and Rules

Bridging …

Research

- Key business needs and challenges motivate solutions based on semantic technology and rules
- Tutorial starts with exploring these motivations by setting specific business contexts, drivers, forces and challenges typically met in information systems across most of industry
- Following this, is an overview and positioning of industry standards for ontologies (RDFS, OWL) and rules (RIF, RuleML, SPARQL, SWRL)
- Ontology-rule integration (SWRL, RDF Gateway) and reactive rules (via Reaction RuleML) will also be covered

… Bridging

Practice

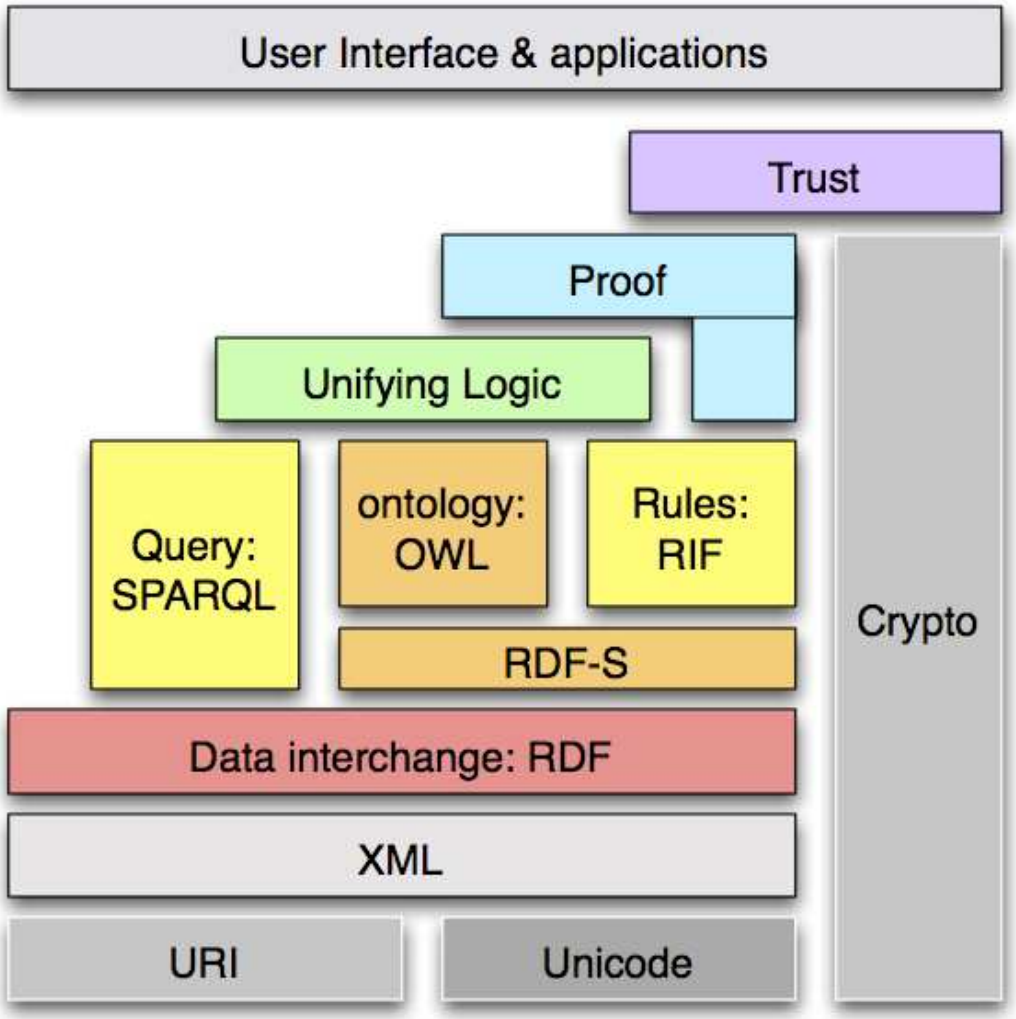
- Tutorial will describe and demonstrate development environments (Protégé, TopBraid Composer (http://www.topbraidcomposer.com) and execution engines (Jess, Jena, Pellet)
- Using use cases and qualifying criteria developed at W3C, some specific solution archetypes will be described
- realEstate Use Case and its solution architecture will be developed in full. Tutorial will evaluate the real-world applicability of the solutions discussed

Intro

The Need for Rules

- Rule-less software is like database-less software
  - Business logic, policies and rules often hard-coded (in Java)
  - Should be developed and maintained in separate ‘rule box’
  - Standardization situation as before the spreading of SQL (well, RuleML is networking with RIF, SBVR/PRR, CL, …)
- Rule-based software is driven by several communities
  - Semantic Web: RIF Charter (→ RDF metadata + OWL ontologies)
  - Web Services: SWSL (→ reactive behavior + processes)
  - Business Rules: Manifesto (→ real-world application)
- Rules need be interchanged over the Web
  - Semantic classification of rules: Datalog ⊂ Horn logic ⊂ …
  - Semantic compatibility & XML syntax => Translation possible

Semantic Web Stack: Tim Berners-Lee 2006



The W3C RIF Working Group

"This Working Group is chartered to produce a **core rule language** plus **extensions** which together allow **rules** to be **translated between rule languages** and thus transferred between rule systems. The Working Group will have to balance the needs of a diverse community — including Business Rules and Semantic Web users — specifying extensions for which it can articulate a **consensus design** and which are sufficiently **motivated by use cases**."

http://www.w3.org/2005/rules/wg/charter

The W3C RIF Use Cases and Requirements

"**Nearly fifty use cases** documenting the need for a RIF were originally submitted. These were grouped into **eight general** categories and then synthesized as much as possible. In the second round, two new use cases were added. The … use case descriptions, guided by this synthesis, provide scenarios that **motivate the need** and **explain the benefits** of a RIF. They are also intended to provide an ongoing reference point for the working group in its goal of providing a precise set of **requirements** for a RIF."

http://www.w3.org/2005/rules/wg/wiki/UCR/Use_Cases

The W3C RIF Use Cases

1. Negotiating eBusiness Contracts Across Rule Platforms
2. Negotiating eCommerce Transactions Through Disclosure of Buyer and Seller Policies and Preferences
3. Collabor. Policy Development for Dynamic Spectrum Access
4. Access to Business Rules of Supply Chain Partners
5. Manag. Inter-Organizational Business Policies and Practices
6. Ruleset Integration for Medical Decision Support
7. Interchanging Rule Extensions to OWL
8. Vocabulary Mapping for Data Integration
9. BPEL Orchestration of Rule-Based Web Services
10. Publishing Rules for Interlinked Metadata

The W3C RIF Phase 1 Requirements (I)

Compliance model
RIF must define a compliance model that will identify required/optional features.
Default behavior
RIF must specify at the appropriate level of detail the default behavior that is expected from a RIF compliant application that does not have the capability to process all or part of the rules described in a RIF document, or it must provide a way to specify such default behavior.
Different semantics
RIF must cover rule languages having different semantics.
Embedded comments
RIF must be able to pass comments.
Embedded metadata
RIF must support metadata such as author and rule name.
Implementability
RIF must be implementable using well understood techniques, and should not require new research in e.g. algorithms or semantics in order to implement translators.
Limited number of dialects
RIF must have a standard core and a limited number of standard dialects based upon that core.
OWL data
RIF must cover OWL knowledge bases as data where compatible with Phase 1 semantics.
RDF data
RIF must cover RDF triples as data where compatible with Phase 1 semantics.

The W3C RIF Phase 1 Requirements (II)

Rule language coverage

RIF must cover the set of languages identified in the Rulesystem Arrangement Framework.

Semantic precision

RIF core must have a clear and precise syntax and semantics. Each standard RIF dialect must have a clear and precise syntax and semantics that extends RIF core.

Dialect Identification

RIF must have a standard way to specify the dialect of the interchanged rule set in a RIF document.

Standard components

RIF implementations must be able to use standard support technologies such as XML parsers and other parser generators, and should not require special purpose implementations when reuse is possible.

Extensible Format

It must be possible to create new dialects of RIF and extend existing ones upwardly compatible.

Translators

For every standard RIF dialect it must be possible to implement translators between rule languages covered by that dialect and RIF without changing the rule language.

XML syntax / XML types

RIF must have an XML syntax as its primary normative syntax.

RIF must support an appropriate set of scalar datatypes and associated operations as defined in XML Schema part 2 and associated specifications.

Merge Rule Sets / Identify Rule Sets

RIF should support the ability to merge rule sets

RIF will support the identification of rule sets.

The W3C RIF Rulesystems Arrangement Framework (RIFRAF)

- Classify wide variety of rule language features
- Introduce expressive discriminators for derivation rules:
  - Purely-Syntactic
  - Syntactic-entailing-Semantic
  - Semantic
  - Pragmatic
- Add discriminators for Event-Condition-Action (ECA) rules and for Types in rule languages
- http://www.w3.org/2005/rules/wg/wiki/Rulesystem_Arrangement_Framework

The W3C RIF Core Design

- First defining a **CORE Condition Language**
- These conditions are then **used as rule bodies** to define a CORE Horn Language
- A **human-oriented syntax**, an **XML syntax**, and the **semantics** of the condition language and of the Horn rule language are given (preliminary)
- http://www.w3.org/2005/rules/wg/wiki/CORE

The W3C RIF Core Design: Human-Readable Syntax

The document RIF Use Cases and Requirements describes the use case "Negotiating eBusiness Contracts Across Rule Platforms", containing this first rule proposed in a hypothetical negotiation by some agent John:

"If an item is perishable and it is delivered more than 10 days after the scheduled delivery date then the item will be rejected."

```
reject(John ?item) :-
  And ( perishable(?item)
        delivered(?item ?deliverydate)
        scheduled(?item ?scheduledate)
        timediff(?diffdate ?deliverydate ?scheduledate)
        greaterThan(?diffdate 10) )
```

The W3C RIF Core Design: XML Syntax

```
<Forall>
  <Var>item</Var>
  <Var>deliverydate</Var>
  <Var>scheduledate</Var>
  <Var>diffdate</Var>
  <Implies>
    <head>
      <Atom>
        <Rel>reject</Rel>
        <Con>John</Con>
        <Var>item</Var>
      </Atom>
    </head>
    .
    .
    .
<body>
  <And>
    <Atom>
      <Rel>perishable</Rel>
      <Var>item</Var>
    </Atom>
    <Atom>
      <Rel>delivered</Rel>
      <Var>item</Var>
      <Var>deliverydate</Var>
    </Atom>
    <Atom>
      <Rel>scheduled</Rel>
      <Var>item</Var>
      <Var>scheduledate</Var>
    </Atom>
    .
    .
    .
    <Atom>
      <Rel>timediff</Rel>
      <Var>diffdate</Var>
      <Var>deliverydate</Var>
      <Var>scheduledate</Var>
    </Atom>
    <Atom>
      <Rel>greaterThan</Rel>
      <Var>diffdate</Var>
      <Con>10</Con>
    </Atom>
  </And>
</body>
</Implies>
</Forall>
```