W3C Rules Interchange Format
(Basic Logic Dialect)

Chris Welty
IBM Research &
RIF WG co-Chair
Rules, rules, rules...
RIF Background

• Semantic web approach
  – interoperability requires a formal semantics

• The OWL WG approach
  – Start with something (DAML+OIL, OWL1.1)

• Literally *hundreds* of rule system implementations
  – JSR-94, JESS, ISO-Prolog, CLIPS, OPS...
  – Already several “SW” rule languages
    • SWRL, RuleML, WRL, SWSL, KAON2, TRIPLE, JenaRules...
  – Everyone wants “their” system/technique/theory to be the SW rule standard

• Workshop April ‘05
  – Outcome: *Nash Equilibrium*...
Enter the Vendors

• “Rule systems Vendors”
  – $1B/year existing market
  – iLog (IBM), Fair Isaac, Haley, …

• Database vendors
  – Oracle, IBM

• OMG PRR effort
  – Simple production rules
  – Event-condition-action
  – Vendors understand the value of standardization
  – Interchange already a priority
  – …a common semantics?
Credits

- Editors
  - Michael Kifer (U. Stonybrook), Harold Boley (NRCC), Jos de Bruijn (U. Bolzano), Axel Polleres (DERI Galway), Christian de Sainte-Marie (Ilog/IBM), Gary Hallmark (Oracle), Dave Reynolds (HP)

- Active WG members
  - Adrian Paschke (REWERENCE), Hassan Aït-Kaci (ILOG), John Hall (OMG), Jos de Bruijn (FUB), Leora Morgenstern (NYU), Mike Dean (SRI), Shanghai Ke(Ilog/IBM), and Stella Mitchell (IBM)

- WG Team
  - Chris Welty (IBM), Christian de Sainte Marie (ILOG), and Sandro Hawke (W3C/MIT)
What is a Rule?

• IF <condition> THEN <conclusion>
  – <condition> aka rule body, antecedant
  – <conclusion> aka rule head, consequent

• In RIF BLD (Basic Logic Dialect)
  – Condition and conclusion are monotonic
  – IF (child-of(y,x) AND brother-of(z,x)) THEN (uncle-of(z,y))
  – You cannot change the value of any predicate, all statements are either true or false (as in OWL or RDF)

• In RIF PRD (Production Rule Dialect)
  – Condition and conclusion are non-monotonic
  – IF (customer.level = “gold”) THEN (customer.discount = 10%)
  – This will change the value of customer.discount if it was e.g. 5% before the rule (as in a programming language)
RIF Design

RIF BLD
- Equality in conclusions
- Membership in conclusions
- External Functions
- Frame subclass
- Open Lists
- “logic” functions

RIF PRD
- Conclusion “actions”
- Negation
- Frames-as-objects
- Retraction
- Subclass
- Membership in conclusion

RIF CORE
- Horn (monotonic)
- Datatypes & builtins
- External functions
- Frames, class membership (in conditions)
- Equality (in conditions)
- Ground lists
- Existential quantification (in conditions)
RIF Documents

• RIF Core, BLD, PRD
• DTB – datatypes and builtins
  – Defines all the required datatypes and pre-defined predicates and functions on them
• SWC – RDF & OWL Compatibility
  – Defines syntax and semantics for combining RIF rules with RDF graphs and OWL-2
• FLD – Framework for Logic Dialects
  – Defines how new dialects that extend BLD can be described
Negation

“One must understand, [rule systems] include a version of negation that is operationally well-defined. The problem is to make sense of it.”

Fitting, 1999

“And agree on it. Thus RIF logic dialects (CORE and BLD) do not have negation.”

Welty, just now
Objects and Frames

id [ slot -> value ]

a1 [ status -> “active”
    discount -> .1 ]
ForAll ?x ( ?x [ discount -> .2 ] :-
    ?x [ status -> “active” ] )

Object (PRD, Java)
- a1 [ status -> “active”
    discount -> .2 ]

Frame (Core, BLD)
- a1 [ status -> “active”
    discount -> .1
    discount -> .2 ]

Change is non-montonic

Change is montonic. All changes result in the KB getting larger.
BLD Overview

• Definite Horn rules
  – Disjunction of atoms with exactly one positive literal
    \[ A : - B \text{ and } C \text{ and } D \ldots \]
• Equality, functions, lists, and a standard first-order semantics
• Syntactic features
  – frames
  – internationalized resource identifiers (IRIs) as identifiers
  – XML Schema data types and builtins
• XML (1.0) syntax with normative XMLS definition
• Non-normative presentation syntax
• Metadata and imports
Symbols & Datatypes

- Used to identify constants, variables, functions, predicates
- "literal"^^<symspacenamespace-identifier>
  - Notable symbol spaces: xsd:string, rif:local, rif:iri
  - “Chris”^^<xsd:string>
  - “http://www.w3.org/1999/02/22-rdf-syntax-ns#type”^^<rif:iri>
  - “Person1”^^rif:local

- Datatype literals have lexical space and value space
  - Taken from XML Schema Datatypes (xsd)
  - Lexical space defines the syntax of the constant
  - Value space defines the domain
  - Values, value spaces are disjoint
    - “Chris”^^<xsd:string> != “Christopher”^^<xsd:string>
    - “1”^^<xsd:float> != “1”^^<xsd:int>
    - “1”^^<xsd:float> == “1.0”^^<xsd:float>

Denotes the same entity across documents, rule sets, the web, ...
May denote a different entity in different RIF documents
Frames

- In Core & BLD frame syntax is available
  \[ id \ [ slot \rightarrow value] \]
- ...but frames are NOT objects, they are RDF triples
  \(<id> <slot> <value>\).
- Like RDF, you cannot \textit{change} a triple, you can only \textit{add a new one}
  
  – This is what \textit{monotonic} means
- RIF frames also give you membership (type) and subclass (BLD only)
  \[ id \ # \ class \ ## superclass \ [ slot \rightarrow value] \]
  \(<id> <slot> <value>\).
  \(<id> <rdf:type> <class>\).
  \(<class> <rdfs:subclassOf> <superclass>\).
Frames and Predicates

- Traditionally, logic uses *predicates* to indicate properties and relations on entities:
  - Person(Chris)
  - Father-of(Chris, Rachel)
  - Employee-of(Chris, IBM, 2004)
  - Degree(Chris, RPI, 1995, PhD, Ed-Rogers)

- RDF only has *binary* relations, which it calls “properties”:
  - *sigh* A property is supposed to be unary
  - Nodes and edges in a graph
  - Unary relations (nee properties) represented using a special binary relation, rdf:type
  - Frames represent unary and binary relations with special syntax, basically the same as RDF

- The difference between frames and predicates is simply a matter of taste, they can be used to encode the same information
RIF BLD

• BLD rule:
  – (\textbf{Forall} var* <conclusion> :- <condition>)
  – Conclusions may contain conjunction (aka AND)
  – Conditions may contain conjunction, disjunction (OR), and existential quantification

• Restrictions on conclusion
  – No existential, disjunction, external functions

•Existentials

\texttt{\textbf{Exists} \ ?y ( P ( \ ?x \ ?y ) )}

  – “There exists a value for \ ?y such that P(\ ?x \ ?y) is true”

\texttt{Parent(\ ?x) :- Exists \ ?y ( HasChild(\ ?x \ ?y))}

  – In Core & BLD, exists can only appear in conditions
Structure

• Rules occur in Groups

\[
\text{Group(} \ (\text{Forall } ?x \ _Q(?x) :- \ _P(?x)) \\
(\text{Forall } ?x \ _Q(?x) :- \ _R(?x)) \ \text{)}
\]

• Groups occur in Documents

\[
\text{Document(} \\
\text{Group(} (\text{Forall } ?x \ _Q(?x) :- \ _P(?x)) \\
(\text{Forall } ?x \ _Q(?x) :- \ _R(?x)) \ \text{)} \\
\text{Group(} (\text{Forall } ?y \ _R(?y) :- \ <\text{ex:op}>(?y)) \ \text{)})
\]
Simple Example
Database mapping

  my:actorIn(?aname ?mname) :-
  And( dbp:starring(?movie ?actor) rdfs:label(?movie ?mname) rdfs:label(?actor ?aname))))
Simple Example w/ Frames
Database mapping

Document( Prefix(dbp http://dbpedia.org(property/))
Prefix(my http://mydata.org/resource#)
Prefix(rdfs http://www.w3.org/2000/01/rdf-schema#)
Group ( Forall ?mname ?aname ?movie ?actor
    my:actorIn(?aname ?mname) :-
        And(?movie[dbp:starring -> ?actor
            rdfs:label -> ?mname]
            ?actor[rdfs:label ?aname])))
Builtins and Negative Guards
Entailing disjunction

Document( Prefix(ex http://example.com/example#)
  Prefix(pred http://.../rif-builtin-predicate#)
  Group( ex:p(ex:a)
    Forall ?x
      (1=2 :-
        And (ex:p(?x)
          External(pred:isNotInteger(?x))
          External(pred:isNotString(?x)))) ) )

Entails:

Or ( External(pred:isInteger(ex:a))
     External(pred:isString(ex:a)) )
Compatibility With RDF Frames, membership, subclass


Document( Prefix(ex http://example.com/example#)
  Import(<http://example.org/mygraph>
    <http://www.w3.org/2007/rif-import-profile#Simple>)
  Group ( ex:A ## ex:B    ex:k # ex:A ) )

Entails:
And ( ex:i # ex:A ex:k[rdf:type->ex:A]
  ex:i # ex:B
  ex:i[rdf:type->ex:B] )
XML Syntax

And (Exists ?Buyer (cpt:purchase(?Buyer ?Seller
cpt:book(?Author bks:LeRif)
curr:USD(49)))
?Seller=?Author )

<And><formula><Exists><declare><Var>Buyer</Var></declare>
 <formula><Atom><op><Const type="&rif;iri">&cpt;purchase</Const></op>
 <args ordered="yes"><Var>Buyer</Var><Var>Seller</Var>
 <Expr><op><Const type="&rif;iri">&cpt;book</Const></op>
 <args ordered="yes">
  <Var>Author</Var>
  <Const type="&rif;iri">&bks;LeRif</Const></args>
 </Expr>
 <Expr><op><Const type="&rif;iri">&curr;USD</Const></op>
 <args ordered="yes">
  <Const type="&xsd;integer">49</Const></args>
 </Expr>
 </Atom></formula>
</Exists></formula>
<formula><Equal>
 <side><Var>Seller</Var></side>
 <side><Var>Author</Var></side>
</Equal></formula>
</And>
Metadata

• Can hang off any “class element”

<And>
<meta> <Frame>
<object><Const type="rif:local">pd</Const></object>
<slot ordered="yes">
<Const type="&rif;iri">&dc;publisher</Const>
<Const type="&rif;iri">&w3;W3C</Const>
</slot> </Frame> </meta>

<formula>...</formula>
</And>
Metadata

• Can hang off any “class element”

```
    <Group>
    <meta>  <Frame>
    <object><Const type="rif:local">pd</Const></object>
    <slot ordered="yes">
        <Const type="&rif;iri">&dc;publisher</Const>
        <Const type="&rif;iri">&w3;W3C</Const>
    </slot>  </Frame>  </meta>
```

...
Metadata

• Can hang off any “class element”

<Document>
<meta> <Frame>
<object><Const type="rif:local">pd</Const></object>
<slot ordered="yes">
  <Const type="&rif;iri">&dc;publisher</Const>
  <Const type="&rif;iri">&w3;W3C</Const>
</slot> </Frame> </meta>

...

</Document>
Logic Functions (BLD)

Document(
  Prefix(pred <http://www.w3.org/2007/rif-builtin-predicate#>)
  Prefix(func <http://www.w3.org/2007/rif-builtin-function#>)
  Prefix(ex <http://example.org/example#>)

Group (
  ex:factorial(0) = 1

  Forall ?N ( 
    ex:factorial(?N) = 
    External(func:numeric-multiply( 
      ?N 
      ex:factorial( 
        External( 
          func:numeric-subtract(?N 1) ) ) ) 
    :- External(pred:numeric-greater-than(?N 0) ) ) )
)
Status

• RIF BLD, RDF&OWL in Last Call
  – Last call indicates “final version for external review”
  – Expected “call for implementation” (Candidate Recommendation) any day now
  – Roughly 10 implementation commitments

• See http://w3.org/2005/rules