The Evolving Role of Rules and Ontologies on the Semantic Web

Chris Welty
IBM Research
Great moments in evolution
Outline

• New Ontology Language Standards
  – OWL-2 and RIF
• What’s happenin’
  – Semantic Web and Linked Data
• How I was right, even when I was wrong
  – About the web and the semantic web
• Evolving the Role of Ontologies
OWL-2

• Some enhancements to OWL(1)
  – Self Restriction
  – QCRs
  – Reflexive, Irreflexive, Assymmetric relations
  – Disjoint Properties
  – Relation composition
  – Keys
  – Limited metamodeling
  – More useful dialects EL, QL, RL

• Many restrictions remain
  – Only unary, binary predicates
  – separation between object and data properties
  – no variables
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)
Triples 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)
- OWL/RDF only has binary relations, called “properties”
  - *sigh* A property is supposed to be unary
  - Nodes and edges in a graph, a “triple”
  - Unary relations (nee properties) represented using a special binary relation, rdf:type
- The difference between triples and predicates is simply a matter of taste, they can be used to encode the same information
- Non-monotonic
Objects vs. Triples

status(a1, “active”)
discount(a1, “.1”)

ForAll ?x discount(?x, “.2”) :- status(?x, “active”)

Object (Java, RIF-PRD)
status (a1, “active”)
discount(a1, .2)

Change is non-montonic

Triples (RIF-BLD, OWL)
a1 status “active” .
a1 discount “.1” .
a1 discount “.2” .

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

Same thing in RDF n-triples
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)
OWL vs. RIF

• Mostly the same
  – RIF: operations on datatypes, restrictions on higher arity predicates, optimized for reasoning on data
  – OWL: cardinality restrictions, limited negation, optimized for reasoning on ontology

• More important concerns
  – Style, expertise, tooling, maturity, legacy data
The Semantic Web Vision

• ~80% of web pages are generated from *back end databases*
• Publish the semantics (schema?) as well as the data
• URIs provide a web-based form of identity
  – It’s the semantic WEB, not the SEMANTIC web
• NOT: humans will markup their web pages
• NOT: NLP will populate the SW from web pages
Linked Data Cloud
Linked Data Cloud
Linked Data Cloud
Linked Data Cloud
Linked Data
What is it?

• Lots of data
• Domains include computer science publication, medical informatics, social networking, geospatial
• Many entities are *linked*
  – information about the same entity is stored in different places
  – Same URI -> same entity
  – Different documents, in different places, can reference the same URI
• Not a whole lot of ontology
  – Much of what’s there is not-so-good
How I was right, even when I was wrong...

- The semantic web has been around for about a decade
- Prediction/Warning? — “In a few years, several independent standards will be competing, all of which will be equivalent in value to the initial WC3 effort but none of which will be dominant. The result of this proliferation will be a proliferation of bad ontologies.”
  - Ultimately SW will fail
- What I missed
  - It's not just tags
  - KA does scale – people want to share their knowledge
  - A lot of people don’t care about reasoning
  - Better not perfect
  - KA not needed – the actual vision
- Why I was right:
  - Reasoning doesn’t really scale (exptime *is* incomplete)
  - Bad ontologies do lead to bad systems, but it doesn’t always matter
I’m not the only one...

• The web has been around for about two decades
• Prediction (CHI research in the late 90s):
  – Hypertext is decades old, HTML "just adds tags"
  – Will not scale
  – Makes a horrible UI and will lead to bad systems
  – Ultimately will fail
• What they missed
  – It’s not just tags
  – It did scale – became the way of sharing information
  – A lot of people don’t care about good UIs
  – Better not perfect
• Why they were right:
  – It really was a bad UI, so it kept getting better
  – Good UI do matter sometimes
    • a lot of money now spent on good web design
    • Shift over time from “roll it out” to “think it out”
Role of Ontologies on the SW

• Current LoD trends seem to indicate ontologies have little role in the SW
  – Emphasis on *simple* is better (look at FOAF)
  – Quality a second priority, get it out there

• Doesn’t *always* matter, but...
  – For some cases it does
  – Publications, social network, general knowledge: not so much
  – Medical, intel: much

• The evolving role
  – More applications where ontologies matter
  – Money spent on good ontology design
  – Like the web, a shift over time from “roll it out” to “think it out”

• ...what are elements of Ontology Quality?
Its about the instances

• For every class, think about what an instance of it is
  – What is an instance of “ECM Classification”?  
• Classes do not describe their subclasses
• Criteria for individuation must remain constant within a taxonomy
  – Instance of a class is also an instance of every superclass
  – Explore the “boundary conditions”
• “Leaf Nodes” of a hierarchy have no special significance
  – Don’t switch to instances
Common Pitfalls

• Composition (part of)
  – Arm subclass body
• Constitution
  – Statue subclass marble
• Disjunction
  – (class Car partial (all hasPart CarPart)
  – (Engine subclass CarPart)
  – (Tire subclass CarPart)
• Polysemy
  – Book subClass PhysicalObject
  – Book subClass ConceptualCreation
• Meaningless organizational nodes
  – FictionalBookbyLatinAmericanAuthor subClass FictionalBook
• Instance
  – PinotNoir instanceof Grape
• Temporality
  – YoungElvis instanceOf Elvis
The linguistic tests

• If P subclass Q, you should be able to say “P is a kind of Q”
• If a instanceOf P, you should be able to say, “a is a P”
• If a instanceof P subClassOf Q, you should be able to say “a is a Q”
• For every instance, there should be a class it is (rigidly) an instance of that is its natural label
• You should not find it natural to say, if P subclassOf Q, “P has Q”, “P might be Q”, “P was Q”
What’s in a name

• Don’t argue about what specific terms mean
  – Common software architecture argument: “What is a bridge?”
• Try and find the distinctions that matter
  – Assign them labels later
• Avoid “ish” “-thing” & “other-” classes
  – Find good names that will avoid meaning creep
  – Other- classes create a maintenance nightmare
• Classes describe their instances
  – Remember the linguistic tests
• The superclass is not part of the name
  – So don’t assume it is (e.g. Best_Practices subClassOf Document)
Kinds of classes

• Backbone Types
  – Rigid (unchangeable)

• Roles
  – Identify a part played in a relation

• Classification nodes
  – Merely to group/organize subclasses
Backbone

• Classes in the backbone ("backbone types") are unchangeable
• Think of them as the natural labels for their instances
• Compare “being software” with “being expensive”
Ontology vs. Classification

• Ontology is supposed to be about *objective* reality
• Classifications are entirely subjective
  – E.g. Dewey decimal
• For artifacts, however, the distinction can blur
  – Concentrate on the rigid truths
Does quality matter?

- Although ontology is itself hundreds of years old, we are only now crafting large machine-processable ontologies
- Ontology engineering is fairly new
- About at the 1980s software engineering phase
  - But moving faster
- Same arguments apply
  - Good ontologies are easier to maintain, check, integrate, debug, extend
  - Good ontologies support reuse
  - Good ontologies also support a new class of knowledge-based systems
- Studies have shown 8-30% performance increases of good ontologies vs. bad
Measuring Quality (Static)

• Static evaluation is really human evaluation
  – Ontologies are knowledge – someone has to verify the knowledge is correct

• Example instances for each class
  – “vetted” up the taxonomy

• Assistants for linguistic tests, and names

• Tests for the standard pitfalls

• Tests for the backbone
Automated Metrics

- Typically linguistic
  - Statistical
  - Based on some resource (WordNet)
- Typically based on class labels
- Statistical linguistic evaluation
  - Normalized labels against the common use
  - Language is inherently fuzzy – ontologies try to capture distinctions (the statue and marble)
- Resource based evaluation
  - Ends up pushing the resources distinctions on the ontology
  - “Movie” in WN is a hyponym of “photographic paper”
Conclusion

• New Ontology Language Standards
  – http://www.w3.org/2005/rules/
  – http://www.w3.org/2007/OWL/

• Semantic Web trends
  – http://linkeddata.org/
  – http://dbpedia.org/

• Ontologies will have an increasing role in SW

• Quality will matter
  – Ontologies are *engineered artifacts*!