An Application of Ontology-based Rules to Situation Awareness

Christopher J. Matheus
Versatile Information Systems, Inc.
Framingham, MA USA
cmatheus@vistology.com

Outline

- Problem Domain: Situation Awareness
- High-level Methodology
- Specific Use Cases
- Issues/Challenges with SWRL
- Rule Language ~~Requirements~~ Wish List

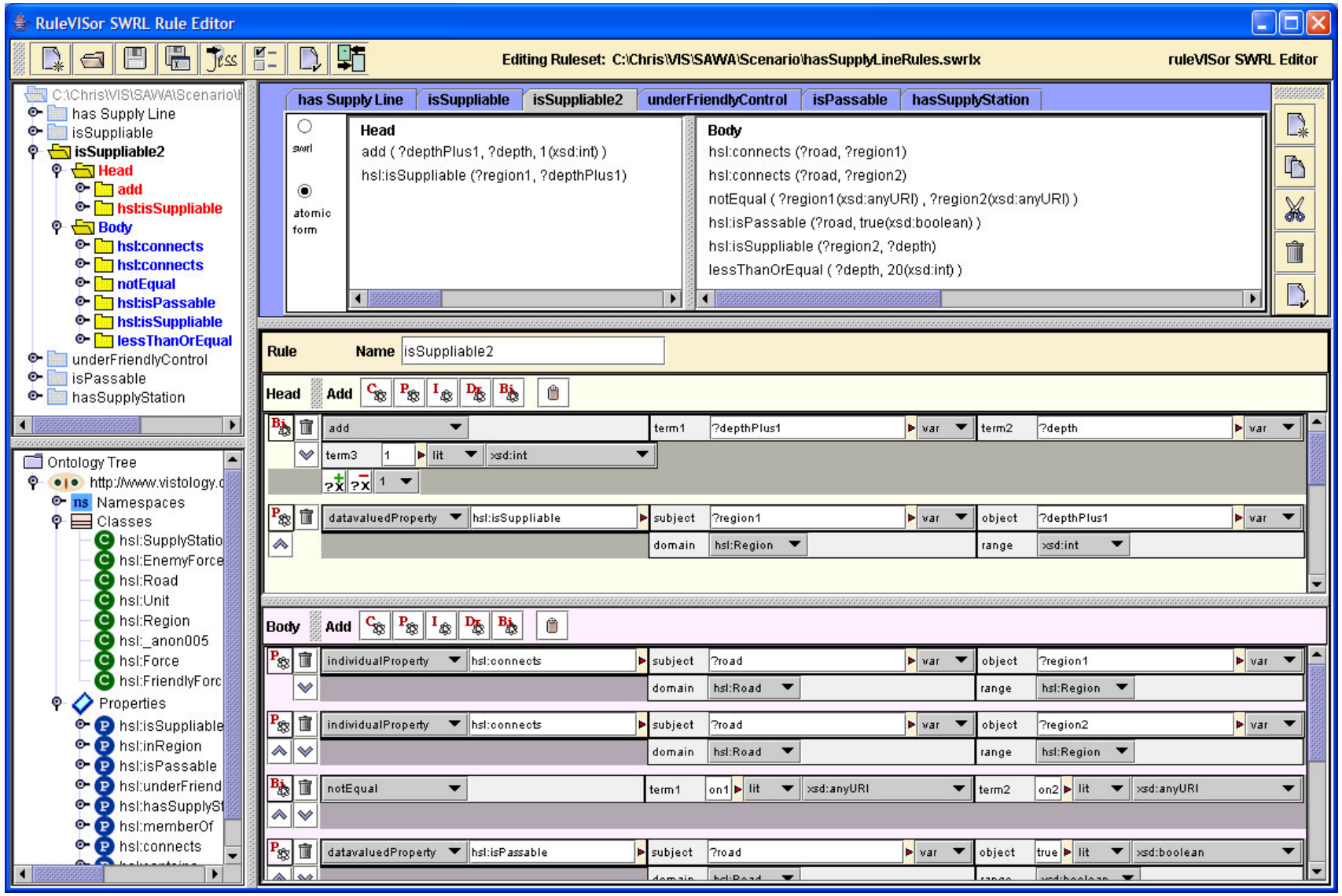
Our Problem Domain

- R&D focus: Formal yet Practical Applications for Situation Awareness & Information Fusion
- Situation Awareness (SAW):
  - an understanding of what's going on in an evolving situation e.g. battlefield, financial markets, crisis management
  - involves fusion of object-level data from multiple sources into meaningful higher-order relations
  - highly context dependent and goal directed
- Requirements for effective SAW apps:
  - domain knowledge about relevant objects and their properties
  - specification of conditions that define higher-order relations
  - a means for reasoning about time-dependent sensor information in the context of the given domain knowledge
  - much in common with SW goals of knowledge representation and processing but with real-time and uncertainty concerns

VIS Use Cases

- SAWA: Situation Awareness Assistant *(AFRL)*
  - Components:
    - Knowledge Management: ezOwl & RuleVISor
    - Runtime: Jess/BaseVISor inference/query engine
  - Domain: supply logistics
- SIXA: Semantic Information eXchange Arch. *(ONR)*
  - ontology-based (C2IEDM/OWL) information mediation
  - reason about track data using pedigree ont & rules
- Situation Development Adviser *(Army)*
  - battlefield ontology
  - doctrinal and heuristic rules of ECOA (SWRL?)

RuleVISor


RuleVISor SWRL Rule Editor
Editing Ruleset: C:\Chris\VIS\SAWA\Scenario\hasSupplyLineRules.swrlx
ruleVISor SWRL Editor
has Supply Line
isSuppliable
isSuppliable2
underFriendlyControl
isPassable
hasSupplyStation
Head
add (?depthPlus1, ?depth, 1(xsd:int))
hsl:isSuppliable (?region1, ?depthPlus1)
Body
hsl:connects (?road, ?region1)
hsl:connects (?road, ?region2)
notEqual (?region1(xsd:anyURI) , ?region2(xsd:anyURI))
hsl:isPassable (?road, true(xsd:boolean))
hsl:isSuppliable (?region2, ?depth)
lessThanOrEqual (?depth, 20(xsd:int))
swrl
atomic form
Rule Name isSuppliable2
Ontology Tree
Namespaces
Classes
Properties

High-Level Methodology

Working with Subject Matter Experts we:

- develop OWL ontologies for describing domain-specific object classes and object properties
- develop SWRL* rules to define relations that are grounded in observable data annotated by ontologies
- convert rules to Jess or BaseVISor rules using XSLT
- establish an input stream of events describing object observations annotated using the domain ontologies
  - all observed values annotated with units, time, certainty, and source derived from an Event ontology
- use Jess/BaseVISor engine to process event stream and detect evolution of higher-order relations

Issues/Challenges with SWRL

- Restriction to binary predicates makes many rules very difficult to construct and understand
  - e.g. criticalPartAtFacility(?Part,?Fac,?Time,?Amt)
  - e.g 9 rules turned into >1000 lines of SWRL code
- Declarative Semantics vs Implementation
  - SWRL built-ins
    - need functional built-ins that specify input and output terms
    - e.g., swrlb:sum(100,?X, ?Y) with unbound vars is infinite
    - practical solution: detect the **one** unbound var to determine the function to compute (multiple unbound vars throws error)
  - No explicit generation/assertion of new facts
    - issue with vars in head that are unbound in the body
    - need assert() and gensym()

criticalPartAtFacility(?Part,?Fac,?Tine,?Amt)

```
<swrlx:classAtom>
    <owlx:Class owlx:name="#CriticalPartAtFacility"/>
    <ruleml:var>?CPFStatement</ruleml:var>
</swrlx:classAtom>
<swrlx:individualPropertyAtom swrlx:property="#criticalPart">
    <ruleml:var>?CPFStatement</ruleml:var>
    <ruleml:var>?Part</ruleml:var>
</swrlx:individualPropertyAtom>
<swrlx:individualPropertyAtom swrlx:property="#criticalFacility">
    <ruleml:var>?CPFStatement</ruleml:var>
    <ruleml:var>?Facility</ruleml:var>
</swrlx:individualPropertyAtom>
<swrlx:datavaluedPropertyAtom swrlx:property="#criticalTime">
    <ruleml:var>?CPFStatement</ruleml:var>
    <ruleml:var>?Time</ruleml:var>
</swrlx:datavaluedPropertyAtom>
<swrlx:datavaluedPropertyAtom swrlx:property="#criticalDeficit">
    <ruleml:var>?CPFStatement</ruleml:var>
    <ruleml:var>?SurplusOrDeficitAmount</ruleml:var>
</swrlx:datavaluedPropertyAtom>
```

Issues/Challenges (continued)

- Time issues
  - usually need to make decisions from partial information
    - requires NAF (could be within scoped context ala N3)
  - need to model time-dependent attributes (e.g. position)
    - more appropriately done as a procedural attachment
  - some computed information is needed only occasional
  - time stamping
    - all data needs to be time stamped
    - asserted inference results also need to be time stamped
    - rules need to be time aware

Top Ten Rule Wish List

1. Rules definable on top of OWL ontologies
2. NAF, perhaps within a scoped context (ala N3)
3. Procedural attachments
4. Explicit representation of non-binary predicates
5. Explicit generation of new facts (assert,gensym)
6. Functionally defined built-ins
7. Graphical means to generate/understand rules
8. Means of generating simple explanations of conclusions
9. Real-time or near-real-time performance
10. Built-in support for reasoning about uncertainty