A Semantic Web Application for the Air Tasking Order (ATO)

ICCRTS – Jun 13 2005 2:30-3:00

Albert Frantz, Milvio Franco
In-house Program
Air Force Research Laboratory
AFRL/IFSA Rome NY

Mentor: Prof. Bart Selman, Cornell Univ.
A Semantic Web Application for the Air Tasking Order (ATO) (Briefing Charts)
Agenda

• Semantic Web Technologies
  – Limitations with current WWW
  – Semantic Web Vision
  – Web Ontology Languages (OWL, RDF, RDFS SWRL)
  – Semantic Web Services

• Our Air Tasking Order Time Sensitive Target Application
  – ATO Ontology
  – Demo
limitations of the WWW and technologies:

- HTML mixes content with presentation
  - Primarily display and it is human updated
  - Poor for dynamic content (databases)
- Keyword search – great but information overload
  - Search engines locate information, we search
- XML eXtensible Markup Language – tree based
eXtensible Markup Language (XML)

- XML approach is to “wrap” each data item in start/end tags

  <Aircraft>
  <wingspan> 14.8 meters </wingspan>
  <cruise-speed> 70 knots </cruise-speed>
  <description> UAV </description>
  </Aircraft>

- Limited machine processing: knows it’s an aircraft but, doesn’t know the meaning of aircraft

- Semantic Web languages are based on XML
The Semantic Web Vision

• Semantic Web: “The first step is putting data on the Web in a form that machines can naturally understand, or converting it to that form. This creates what I call a Semantic Web and web of data that can be processed directly or indirectly by machines.” Sir Tim Berners-Lee

• Semantics – is the meaning of words or symbols

• Two parts of Vision:
  1. Make the web a collaborative medium
  2. Machine understandable or processable

• Potential: Query, Electronic Commerce/Business, Scheduling, Biotechnology
Semantic Web Languages

OWL - Web Ontology Language

RDF, RDFS - Resource Description Framework (Schema)

SWRL –Semantic Web Rule Language

- A standard way for understanding the semantics (meaning)
- Enables applications (computers) to use the data

\[ \text{subClassOf} \]: states one class is a subset of another class of items. Example: Fighter is a subClassOf CombatAircraft.

\[ \text{properties} \]: properties are relations between classes, individuals and data Example: Mission1 hasAircraft B52H-1

\[ \text{equivalentClass} \]: one class is equivalent to another class. Example: Platform is an equivalentClass to Aircraft.
Semantic Web Languages

- Web Ontology Language – OWL (W3C Recommendation) - son of DAML
- OWL Lite, DL (Description Logic), Full
- Adds property restrictions, logic, rules and expressiveness for the Semantic Web

```
<owl:Class rdf:ID="Mission1">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:someValuesFrom>
        <owl:Class rdf:ID="B52H_1"/>
      </owl:someValuesFrom>
      <owl:onProperty>
        <owl:ObjectProperty rdf:about="#has_aircraft"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Class rdf:about="#combat_mission"/>
  </rdfs:subClassOf>
</owl:Class>
```
• “Web services are software applications that can be discovered, described, and accessed based on XML and standard Web protocols over intranets, extranets and the internet” “The Semantic Web”, Michael C. Daconta, Leo J. Orbst and Kevin T. Smith

• Semantic web services are web services that can accessed and understood by computers.

• Based on the OWL-S (Web ontology language for Semantic Web services).
Semantic Web illustration

Computers

Semantic Web Services
i.e. Distance Calculations

Ontologies/ Knowledge Base
i.e. ATO Ontology
ATO Ontology Time Sensitive Targeting Demo

- ATO – document that assigns aircraft to tasks
- Show UML design – used ArgoUML
- Show ontology – built with Protégé (Stanford Univ.)
- Show example rules
- Show reasoning – used RACER
Top of ATO Ontology
• The knowledge base has three time sensitive targets
  • SA20 near a mosque
  • SA20 not a mosque
  • Command Post

• Several Combat Missions

• Java application calculates if mission can reach target based on speed distance and time window.

• Reasoning rule states to divert a mission it must have aircraft, with the right weapons, to hit the target and not damage the mosque within critical time
Aircraft Ontology in Protégé
A Constraint Violation
Protégé Rules
Potential Future Work

• Bring Prof. Selman back on through Intelligent Information Systems Institute

• Interface to Operational Net Assessment Ontology
  – Merge ontologies – collaborative
  – Map interface – show effects of diverting missions

• Make Java reachability application a semantic web service

• Demonstrate Resource Allocation
  – Use SWRL (Semantic Web Rule Language)
    • Time and numeric reasoning

• Interim/Final Tech. Report (currently 32 pages)
Backup Slides

• Backup Slides
Semantic Web Technology Layers
Semantic Web Languages
RDF – Resource Description Framework

- Resource Description Framework Language: RDF
- Triple: Subject – Predicate - Object

```xml
<rdf:Description rdf:ID="F15">
  <hasName>
    rdf:resource= "http://www.af.mil/ACOntology#Eagle"
  </hasName>
</rdf:Description>
```

**Subject**: F15

**Predicate**: `hasName`

**Object**: `http://www.af.mil/ACOntology#Eagle`
• Resource Description Framework Schema – RDFS adds classes and properties

• Classes
  
  \[
  \text{<rdfs:Class rdf:about="bomber">}
  \]
  
  \[
  \text{<rdfs:subClassOf rdf:resource="combatAircraft"/>}
  \]
  
  \[
  \text{</rdfs:Class>}
  \]

• Properties
  
  \[
  \text{<rdf:Property rdf:ID="aircraftCanCarryConfiguration">}
  \]
  
  \[
  \text{<rdfs:domain rdf:resource="#aircraft"/>}
  \]
  
  \[
  \text{<rdfs:range rdf:resource="#weapon"/>}
  \]
  
  \[
  \text{</rdf:Property>}
  \]
Computational Issues

• Complexity: time and space complexity

• Decidability: a decidable problem has an algorithm that can solve the problem. May not be decidable in time $t$ or space $s$.

• Completeness: algorithm is guaranteed to find a solution when there is one. (not complete algorithms may return some or none of answers)

• Expressiveness: as logic becomes more expressive in representing concepts computational complexity will typically go up.
• Used US Message Text Format (USMTF) ATO message for ontology design in Unified Modeling Language (UML) -ArgoUML

• Design verified by John Beyerle, C3I Associates.

• Used Protégé to build the ontology and knowledge base. www.protege.stanford.edu

• Used RACER reasoner

• Used Protégé Java API to interface distance calculations to the ontology.
OWL-S

- Web Ontology Language for Semantic Web Services.
- Computer-interpretable description of a service.
- Supports automatic web service discovery, invocation, composition and interoperation and execution monitoring.
- Three parts:
  1. **Service profile**: declarative advertisements of service properties for capabilities and discovering services.
  2. **Process model**: detailed API description of a services operation. How to call service, execute it and what it returns.
  3. **Grounding**: details of how to interoperate with the service, via messages.
- Also developing a Resource ontology: allocation types, capacity types, resource composition
• **Profile**: serviceName, textDescription, contactInformation, hasParameter, hasInput, hasOutput, hasPrecondition, hasEffect, serviceParameters, serviceCategory (category name, taxonomy, value, code)

• **Process Model**: atomicProcess, simpleProcess, compositeProcess, sequence, split, split+join, unordered, choice, if-then-else, iterate, repeatUntil

• **Grounding**: required messages in Web Service Description Language (WSDL)
Semantic Web Rule Language
SWRL

• Extends on OWL-Lite and OWL-DL.
• Unary/Binary Datalog RuleML sublanguages of Rule Markup Language.
• Includes high-level abstract syntax for Horn-like rules.
• Currently in W3C proposal stage.
• If antecedent (body) conditions are true, then consequent (head) conditions must be true.
• hasParent(?x1, ?x2) and hasBrother(?x2, ?x3) => hasUncle(?x1, ?x3)
SWRL Example

```xml
<ruleml:imp>
  <ruleml:_body>
    <swrlx:individualPropertyAtom swrlx:property="hasParent"
      <ruleml:var>x1</ruleml:var>
      <ruleml:var>x2</ruleml:var>
    </swrlx:individualPropertyAtom>
    <swrlx:individualPropertyAtom swrlx:property="hasBrother"
      <ruleml:var>x2</ruleml:var>
      <ruleml:var>x3</ruleml:var>
    </swrlx:individualPropertyAtom>
  </ruleml:body>
  <ruleml:head>
    <swrlx:individualPropertyAtom swrlx:property="hasUncle"
      <ruleml:var>x1</ruleml:var>
      <ruleml:var>x3</ruleml:var>
    </swrlx:individualPropertyAtom>
  </ruleml:head>
</ruleml:imp>
```