OIL: an Ontology Language for the Semantic Web

Ian Horrocks\textsuperscript{1}, Stefan Decker\textsuperscript{2}, Dieter Fensel\textsuperscript{3}, Frank van Harmelen\textsuperscript{3,4}, Sergey Melnik\textsuperscript{2}, Michel Klein\textsuperscript{3}, Jeen Broekstra\textsuperscript{4}

and others too numerous to mention

\textsuperscript{1} University of Manchester, UK
\textsuperscript{2} AIFB, University of Karlsruhe, Germany
\textsuperscript{3} Vrije Universiteit Amsterdam, Holland
\textsuperscript{4} Aldministrator Nederland B.V.
Requirements for an Ontology Language

☞ Well designed
  ➟ Intuitive to human users
  ➟ Adequate expressive power

☞ Well defined
  ➟ Clearly specified syntax (obviously)
  ➟ Formal semantics (equally important)
  ➟ Adequate expressive power

☞ Compatible with existing (web) standards
Standards for Ontology Languages

- Proposals already exist for W3C standard *schema* languages
  - XMLS (XML Schema)
  - RDFS (RDF Schema)

- Both have been touted as (standard) web *ontology* languages

- However, both suffer from
  - Expressive inadequacy — lack of basic modelling primitives
  - Poorly (un) defined semantics
Proposed Common Core: OIL

- Simple and intuitive **Frame Language** syntax
  - Many users are frightened by logic-based syntax (I know I am!)
  - Rich range of modelling primitives
  - Can still function as a basic frame language
  - Facilitates construction/adaption of tools

- Semantics defined by mapping to expressive **Description Logic**
  - Well defined formal properties (decidability, complexity)
  - Enriched expressive power (boolean connectives, etc.)
  - Can provide reasoning services to support ontology design

- Compatibility provided by layering on top of **RDFS**
  - Class hierarchy etc. accessible to any RDFS-aware agent

Frames + DL + WWW ⇒ OIL
Why Reasoning Support?

☞ Reasoning support is key feature of OIL

☞ Reasoning is important
  ➤ as design support tool
  ➤ for large ontologies
  ➤ with multiple authors
  ➤ for integrating and sharing ontologies

☞ Because it allows
  ➤ Establishing inter-ontology relationships
  ➤ Checking for consistency
  ➤ Checking for (unexpected) implied relationships

“The Semantic Web needs a logic on top” (Henry Thompson)
OIL Language Overview

OIL restricts frame languages:

☞ No defaults
☞ Limited axioms/rules
☞ Ontology only (limited form of individuals)

Main reasons for this:

☞ Reasoning support
☞ Semantics
OIL extends frame languages:

- Defined classes (necessary and **sufficient** conditions)
- Enhanced slot constraints
  - Restriction to class as well as value
  - Existential and universal restrictions
  - Cardinality constraints with optional class qualifier
  - Boolean expressions as well as class names
  - Sub-slots as well as sub-classes
  - Properties on slots (transitive, symmetrical)
  - Inverse slots
  - ...

- Concrete data types
  - Integers and strings, with min, max, ranges etc.

- Additional kinds of axiom
  - Disjointness, disjoint-coverings, equivalence etc.
OIL by Example

slot-def part-of
  subslot-of structural-relation
  inverse has-part
  properties transitive

class-def defined herbivore
  subclass-of animal
  slot-constraint eats
    value-type plant OR
    slot-constraint part-of
      has-value plant
      min-cardinality 2 vegetable
  disjoint herbivore carnivore

% part-of is a slot
% sub-slot of structural-relation
% inverse is has-part
% it is transitive

% herbivore exactly defined as:
% sub-class of animal
% that eats
% only plants
% or parts of
% plants
% and \( \geq 2 \) types of vegetable
% herbivore and carnivore disjoint
Semantics via translation to \textit{SHIQ} DL:

<table>
<thead>
<tr>
<th>OIL</th>
<th>Equivalent \textit{SHIQ}</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textbf{slot-def} part-of</td>
<td>% part-of $\sqsubseteq$ structural-relation</td>
</tr>
<tr>
<td>\textbf{subslot-of} structural-relation</td>
<td>% has-part $\triangleright$ part-of$^-$</td>
</tr>
<tr>
<td>\textbf{inverse} has-part</td>
<td>% part-of $\in \mathbb{R}_+$</td>
</tr>
<tr>
<td>\textbf{properties} transitive</td>
<td></td>
</tr>
<tr>
<td>\textbf{class-def defined} herbivore</td>
<td>% herbivore $\triangleright$</td>
</tr>
<tr>
<td>\textbf{subclass-of} animal</td>
<td>% animal $\sqsubseteq$</td>
</tr>
<tr>
<td>\textbf{slot-constraint} eats</td>
<td>% $\forall$eats.(plant $\sqsubseteq$</td>
</tr>
<tr>
<td>\textbf{value-type} plant OR</td>
<td>% $\exists$part-of.plant) $\sqsubseteq$</td>
</tr>
<tr>
<td>\textbf{min-cardinality} 2 vegetable</td>
<td>% $\geq 2.eats$vegetable</td>
</tr>
<tr>
<td>\textbf{disjoint} herbivore carnivore</td>
<td>% herbivore $\sqsubseteq$ $\neg$carnivore</td>
</tr>
</tbody>
</table>
How to Put Ontologies on the Web
Compatibility via RDFS delivery syntax:

```xml
<rdf:Property rdf:ID="has-part">
  <rdfs:subPropertyOf rdf:resource="#structural-relation"/>
  <oil:inverseRelationOf rdf:resource="#is-part-of"/>
</rdf:Property>

<rdfs:Class rdf:ID="herbivore">
  <rdf:type rdf:resource="http://www.ontoknowledge.org/oil/rdfs-schema/#DefinedClass"/>
  <rdfs:subClassOf rdf:resource="#animal"/>
  <oil:hasSlotConstraint>
    <oil:valueType>
      <oil:hasProperty rdf:resource="#eats"/>
      <oil:hasClass>
        <oil:OR>
          <oil:hasOperand rdf:resource="#plant"/>
          <oil:hasOperand>
            <oil:has-value>
              <oil:hasProperty rdf:resource="#is-part-of"/>
              <oil:hasClass rdf:resource="#plant"/>
            </oil:has-value>
          </oil:hasOperand>
        </oil:OR>
      </oil:hasClass>
    </oil:valueType>
  </oil:hasSlotConstraint>
</rdfs:Class>
```
One of the key ideas behind OIL:

Don’t make the core language too large

Core language should contain only “consensus” primitives
Additional expressive power provided by language extensions

These could include:
- Rules
- Additional algebraic properties on slots
- Limited second order features
- Modules, import, etc.
- ...
Extensions will (hopefully) have similar relationship with OIL core that OIL has with RDFS:

☞ Build on top of OIL core

☞ Use RDFS and OIL core modelling primitives wherever possible

☞ Maximise backward compatibility with RDFS and OIL core
DAML and OIL

- US DAML initiative also developing RDFS based ontology language
- Similar constructs to OIL but different RDFS encoding
- Joint US/EU Committee on Agent Markup Languages now established

? Ultimate aim is OIL/DAML based W3C standard ?
OIL Infrastructure

Reasoning services provided by CORBA FaCT system

- Currently via OIL ↔ FaCT translators (XSL)
- CORBA OIL coming soon

Frame ontology editors being built/adapted to OIL

- Protege editor (Stanford)
- OntoEdit (Karlsruhe)
- OilEd (Manchester)

Additional infrastructure urgently required
OntoWeb Thematic Network

EU Proposal to fund Semantic Web “network of excellence”

55 members from industry and academia in Europe and around the world (including US and Japan)

key objectives

- Technology transfer
- Represent and co-ordinate ontology-related research
- Disseminate information, research and application results
- Represent EU ontology community and co-operating with related initiatives like DAML in the US
- Cooperate with language and content standardisation efforts

http://www.ontoweb.org/

See: http://www.cs.man.ac.uk/~horrocks/Luxembourg.html for notes on EU funding for Semantic Web
WonderWeb Research Project

EU Proposal to fund Semantic Web infrastructure research

6 partners from Europe and US

- University of Manchester, UK (coordinator)
- Vrije Universiteit Amsterdam, Netherlands
- LADSEB-CNR, Italy
- University of Karlsruhe, Germany
- InfoLab, Stanford University, USA
- Interprice Technologies GmbH, Germany

Key objectives

- Ontology languages and standardisation
- Integration/reconciliation techniques for migration and sharing
- Foundational ontologies for range of application domains
- Technical infrastructure and tools for development and deployment

http://www.cs.man.ac.uk/~horrocks/WonderWeb/