OWL: a Reasonable Ontology Language?

Ian Horrocks
<iian.horrocks@comlab.ox.ac.uk>
Information Systems Group
Oxford University Computing Laboratory
What is an Ontology?
What is an Ontology?

An explicit specification of a conceptualization
What is an Ontology?

A model of (some aspect of) the world
What is an Ontology?

A model of (some aspect of) the world

- Introduces **vocabulary** relevant to domain, e.g.:
  - Anatomy
What is an Ontology?

A model of (some aspect of) the world

- Introduces **vocabulary** relevant to domain, e.g.:
  - Anatomy
  - Cellular biology
What is an Ontology?

A model of (some aspect of) the world

- Introduces **vocabulary** relevant to domain, e.g.:
  - Anatomy
  - Cellular biology
  - Aerospace
What is an Ontology?

A model of (some aspect of) the world

- Introduces **vocabulary** relevant to domain, e.g.:
  - Anatomy
  - Cellular biology
  - Aerospace
  - Photography
What is an Ontology?

A model of (some aspect of) the world

- Introduces **vocabulary** relevant to domain, e.g.:
  - Anatomy
  - Cellular biology
  - Aerospace
  - Photography
  - Pizzas
  - ...
What is an Ontology?

A model of (some aspect of) the world

• Introduces **vocabulary** relevant to domain

• Specifies *relative meaning* (aka semantics) of terms

Heart is a muscular organ that is part of the circulatory system
What is an Ontology?

A model of (some aspect of) the world

• Introduces **vocabulary** relevant to domain

• Specifies *relative meaning* (aka semantics) of terms

  Heart is a muscular organ that is part of the circulatory system

• **Formalised** e.g. using suitable logic

  \[
  \text{Heart} \sqsubseteq \text{MuscularOrgan} \sqsubseteq \exists \text{isPartOf. CirculatorySystem}
  \]
What are Ontologies Used For?

• Coherent **shared view** of domain
  – Help identify and resolve disagreements

• Ontology-based **Information Systems**
  – User-centric view of data that is independent of logical/physical schema
  – Answers reflect knowledge & data, e.g.:

Now... *that* should clear up a few things around here
What are Ontologies Used For?

\[ Q(x) \leftarrow \text{Patient}(x) \land \text{suffersFrom}(x, y) \land \text{VascularDisease}(y) \]

i.e., “Patients suffering from Vascular Disease”
What are Ontologies Used For?

\[ Q(x) \leftarrow \text{Patient}(x) \land \text{suffersFrom}(x, y) \land \text{VascularDisease}(y) \]

i.e., “Patients suffering from Vascular Disease”

\[
\begin{align*}
\text{John} : & \text{Patient} \\
\exists & \text{suffersFrom. HeartDisease}
\end{align*}
\]
What are Ontologies Used For?

\[ Q(x) \leftarrow \text{Patient}(x) \land \text{suffersFrom}(x, y) \land \text{VascularDisease}(y) \]

i.e., “Patients suffering from Vascular Disease”

\[
\begin{align*}
\text{John} : \text{Patient} \quad & \quad \exists \text{suffersFrom. HeartDisease} \\
+ \\
\text{Heart} \sqsubseteq \text{MuscularOrgan} \quad & \quad \exists \text{isPartOf. CirculatorySystem} \\
\text{HeartDisease} \equiv \text{Disease} \quad & \quad \exists \text{affects. Heart} \\
\text{VascularDisease} \equiv \text{Disease} \quad & \quad \exists \text{affects. (}\exists \text{isPartOf. CirculatorySystem})
\end{align*}
\]
What are Ontologies Used For?

• Coherent **shared view** of domain
  - Help identify and resolve disagreements

• **Ontology-based Information Systems**
  - User-centric view of data that is independent of logical/physical schema
  - Answers reflect knowledge & data, e.g.:
    “Patients suffering from Vascular Disease”
  - Query expansion/navigation/refinement
  - Incomplete and semi-structured data
  - ...

**More “intelligent” applications**
What are Ontologies Used For?

• Coherent user-centric view of domain
  - Help identify and resolve disagreements
• Ontology-based Information Systems
  - View of data is independent of logical/physical schema
  - Answers reflect schema & data, e.g.:
    “Patients suffering from Vascular Disease”
  - Query expansion/navigation/refinement
  - Incomplete and semi-structured data
  - More “intelligent” applications

Now... *that* should clear up a few things around here
What is the Semantic Web?
What is the Semantic Web?

• According to TBL circa 1998:
  “... a consistent logical web of data ...” in which “... information is given well-defined meaning ...”
What is the Semantic Web?

- According to TBL circa 1998:
  
  “… a consistent logical web of data …” in which
  “… information is given well-defined meaning …”

- By now has evolved into:
  
  “a platform for distributed applications and sharing (linking) data”
What is the Semantic Web?

• According to TBL circa 1998:
  “... a consistent logical web of data ...” in which
  “... information is given well-defined meaning ...”

• By now has evolved into:
  “a platform for distributed applications and sharing (linking) data”
  – RDF provides uniform syntactic structure for data
  – OWL provides machine readable schemas (ontologies)
What is the Semantic Web?

• According to TBL circa 1998:
  “... a consistent logical web of data ...” in which
  “... information is given well-defined meaning ...”

• By now has evolved into:
  “a platform for distributed applications and sharing (linking) data”
  – RDF provides uniform syntactic structure for data
  – OWL provides machine readable schemas (ontologies)

i.e., a large distributed ontology based information system
A Brief History of OWL

• RDF standard first published 1999; revised 2004
• RDF extended to **RDFS**, a primitive ontology language
  – classes and properties; sub/super-classes (and properties); range and domain (of properties)
• But RDFS **lacks** important **features**, e.g.:
  – existence/cardinality constraints; transitive/inverse properties; localised range and domain constraints, …
• And RDF(S) has “higher order flavour” with no (later **non-standard**) **formal semantics**
  – difficult to understand or to provide reasoning support
A Brief History of OWL

- EU On-To-Knowledge project developed OIL
A Brief History of OWL

- EU On-To-Knowledge project developed OIL
- DAML program developed DAML-ONT
- Efforts soon merged to produce DAML+OIL
  - Further development carried out by “Joint EU/US Committee”
A Brief History of OWL

• EU On-To-Knowledge project developed OIL
• DAML program developed DAML-ONT
• Efforts soon merged to produce DAML+OIL
  - Further development carried out by “Joint EU/US Committee”
• DAML+OIL submitted to W3C as basis for standardisation
  - WebOnt WG developed OWL (2004)
  - OWL WG developed OWL 2 (2009)
• OWL (2) based on SHOIN (SROIQ) Description Logics!? 
What are Description Logics (DLs)?

- Fragments of **first order logic** designed for KR
- Useful computational properties
  - **Decidable** (essential)
  - Low complexity (desirable)
- Succinct and **variable free syntax**

\[
\text{Heart} \sqsubseteq \text{MuscularOrgan} \sqcap \\
\exists \text{isPartOf}. \text{CirculatorySystem}
\]

\[
\forall x. [\text{Heart}(x) \rightarrow \text{MuscularOrgan}(x) \land \\
\exists y. [\text{isPartOf}(x, y) \land \\
\text{CirculatorySystem}(y)]]
\]
Why base OWL on a (Description) Logic?

Can exploit the results of 20+ years of DL research

- Well defined (model theoretic) semantics

<table>
<thead>
<tr>
<th>Constructor</th>
<th>DL Syntax</th>
<th>Example</th>
<th>FOL Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>intersectionOf</td>
<td>$C_1 \sqcap \ldots \sqcap C_n$</td>
<td>Human $\sqcap$ Male</td>
<td>$C_1(x) \land \ldots \land C_n(x)$</td>
</tr>
<tr>
<td>unionOf</td>
<td>$C_1 \sqcup \ldots \sqcup C_n$</td>
<td>Doctor $\sqcup$ Lawyer</td>
<td>$C_1(x) \lor \ldots \lor C_n(x)$</td>
</tr>
<tr>
<td>complementOf</td>
<td>$\neg C$</td>
<td>$\neg$Male</td>
<td>$\neg C(x)$</td>
</tr>
<tr>
<td>oneOf</td>
<td>${x_1} \sqcup \ldots \sqcup {x_n}$</td>
<td>${john} \sqcup {mary}$</td>
<td>$x = x_1 \lor \ldots \lor x = x_n$</td>
</tr>
<tr>
<td>allValuesFrom</td>
<td>$\forall P.C$</td>
<td>$\forall$hasChild.Doctor</td>
<td>$\forall y. P(x, y) \rightarrow C(y)$</td>
</tr>
<tr>
<td>someValuesFrom</td>
<td>$\exists P.C$</td>
<td>$\exists$hasChild.Lawyer</td>
<td>$\exists y. P(x, y) \land C(y)$</td>
</tr>
<tr>
<td>maxCardinality</td>
<td>$\leq np$</td>
<td>$\leq 1$hasChild</td>
<td>$\exists^n y. P(x, y)$</td>
</tr>
<tr>
<td>minCardinality</td>
<td>$\geq np$</td>
<td>$\geq 2$hasChild</td>
<td>$\exists^ny. P(x, y)$</td>
</tr>
</tbody>
</table>
Why base OWL on a (Description) Logic?

Can exploit the results of 20+ years of DL research

- Well defined (model theoretic) semantics
- **Formal properties** well understood (complexity, decidability)

I can’t find an efficient algorithm, but neither can all these famous people.

[Garey & Johnson. Computers and Intractability]
Why base OWL on a (Description) Logic?

Can exploit the results of 20+ years of DL research

- Well defined (model theoretic) **semantics**
- **Formal properties** well understood (complexity, decidability)
- Practical **reasoning algorithms**

| □-rule | if 1. \((C_1 \sqcap C_2) \in \mathcal{L}(v), v \) is not indirectly blocked, and  
|        | 2. \{C_1, C_2\} \notin \mathcal{L}(v)  
|        | then \(\mathcal{L}(v) \rightarrow \mathcal{L}(v) \cup \{C_1, C_2\}\). |
| ⊃-rule | if 1. \((C_1 \sqcup C_2) \in \mathcal{L}(v), v \) is not indirectly blocked, and  
|        | 2. \{C_1, C_2\} \cap \mathcal{L}(v) = \emptyset  
|        | then \(\mathcal{L}(v) \rightarrow \mathcal{L}(v) \cup \{E\} \) for some \(E \in \{C_1, C_2\}\). |
| ∃-rule | if 1. \(\exists r.C \in \mathcal{L}(v_1), v_1 \) is not blocked, and  
|        | 2. \(v_1\) has no safe \(r\)-neighbour \(v_2\) with \(C \in \mathcal{L}(v_1)\),  
|        | then create a new node \(v_2\) and an edge \((v_1, v_2)\)  
|        | with \(\mathcal{L}(v_2) = \{C\}\) and \(\mathcal{L}(v_1, v_2) = \{r\}\). |
| ∀-rule | if 1. \(\forall r.C \in \mathcal{L}(v_1), v_1 \) is not indirectly blocked, and  
|        | 2. there is an \(r\)-neighbour \(v_2\) of \(v_1\) with \(C \notin \mathcal{L}(v_2)\)  
|        | then \(\mathcal{L}(v_2) \rightarrow \mathcal{L}(v_2) \cup \{C\}\). |
| ∀+-rule | if 1. \(\forall r.C \in \mathcal{L}(v_1), v_1 \) is not indirectly blocked, and  
|        | 2. there is some role \(r'\) with \(\text{Trans}(r')\) and \(r' \equiv r\)  
|        | 3. there is an \(r'\)-neighbour \(v_2\) of \(v_1\) with \(\forall r'.C \notin \mathcal{L}(v_2)\)  
|        | then \(\mathcal{L}(v_2) \rightarrow \mathcal{L}(v_2) \cup \{\forall r'.C\}\). |
| choose-rule | if 1. \(\leq n r.C \in \mathcal{L}(v_1), v_1 \) is not indirectly blocked, and  
|           | 2. there is an \(r\)-neighbour \(v_2\) of \(v_1\) with \(\{C, \neg C\} \cap \mathcal{L}(v_2) = \emptyset\)  
|           | then \(\mathcal{L}(v_2) \rightarrow \mathcal{L}(v_2) \cup \{E\} \) for some \(E \in \{C, \neg C\}\). |
| ≥-rule | if 1. \(\geq n r.C \in \mathcal{L}(v), v \) is not blocked, and  
|        | 2. there are not \(n\) safe \(r\)-neighbours \(v_1, \ldots, v_n\) of \(v\)  
|        | with \(C \in \mathcal{L}(v_i)\) and \(v_i \neq v_j\) for \(1 \leq i < j \leq n\). |
Why base OWL on a (Description) Logic?

Can exploit the results of 20+ years of DL research

- Well defined (model theoretic) **semantics**
- **Formal properties** well understood (complexity, decidability)
- Practical **reasoning algorithms**
- Effective **implemented systems**
What did OWL ever do for us?
What did OWL ever do for us?

**Ontologies before:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Original Language</th>
<th>defined</th>
<th>primitive</th>
<th>artificial</th>
<th>∑</th>
<th>de-defined</th>
<th>primitive</th>
</tr>
</thead>
<tbody>
<tr>
<td>CKB</td>
<td>SB-ONE</td>
<td>23</td>
<td>57</td>
<td>58</td>
<td>138</td>
<td>2</td>
<td>46</td>
</tr>
<tr>
<td>Companies</td>
<td>BACK</td>
<td>70</td>
<td>45</td>
<td>81</td>
<td>196</td>
<td>1</td>
<td>39</td>
</tr>
<tr>
<td>FSS</td>
<td>SB-ONE</td>
<td>34</td>
<td>98</td>
<td>75</td>
<td>207</td>
<td>0</td>
<td>47</td>
</tr>
<tr>
<td>Espresso</td>
<td>SB-ONE</td>
<td>0</td>
<td>145</td>
<td>79</td>
<td>224</td>
<td>11</td>
<td>41</td>
</tr>
<tr>
<td>Wisber</td>
<td>TURQ</td>
<td>50</td>
<td>81</td>
<td>152</td>
<td>283</td>
<td>6</td>
<td>18</td>
</tr>
<tr>
<td>Wines</td>
<td>CLASSIC</td>
<td>50</td>
<td>148</td>
<td>237</td>
<td>435</td>
<td>0</td>
<td>10</td>
</tr>
</tbody>
</table>
What did OWL ever do for us?

Ontologies after:
What did OWL ever do for us?

Ontologies after:

Welcome to the Protege Ontology Library!

**OWL ontologies**

- **AIM@SHAPE Ontologies**: Ontologies pertaining to digital shapes. Source: [AIM@SHAPE NoE](http://www.aimeurope.org) - Advanced and Innovative Models And Tools for the development of Semantic-based systems for Handling, Acquiring, and Processing knowledge Embedded in multidimensional digital objects.
- **amino-acid.owl**: A small OWL ontology of amino acids and their properties. Source: [Amino Acid Ontology Web site](http://www.aminoacidontology.com).
- **Basic Formal Ontology (BFO)**
- **bhakti.owl**: An OWL ontology for the transcendental states of consciousness experienced by practitioners of bhakti-yoga, a form of Vedic consciousness engineering.
- **Biochemical Ontologies**: Over 30 ontologies for knowledge representation and reasoning across scientific domains. Ontologies are normalized into non-disjoint primitive skeletons and
What did OWL ever do for us?

**Tools** before:

```lisp
> (load-tkb "demo.kb" :verbose T)
..........................................................
..........................................................
> (classify-tkb :mode :stars)
pppppppppppccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
What did OWL ever do for us?

Tools after:
What did OWL ever do for us?

“Profile” before:

2000 International Workshop on Description Logics - DL2000

RWTH Aachen, Germany
August 17 - August 19, 2000

A copy of the proceedings Proceedings is available for free.

Call for Participation

The 2000 International Workshop on Description Logics continues the tradition of international workshops devoted to discussing developments and applications of knowledge representation formalisms based on Description Logics. Demonstrations of systems and DL-based applications will be possible and people interested are encouraged to get in touch with the organizers.

DL2000 will precede ECAI2000 (14th European Conference on Artificial Intelligence) which will be held in Berlin, Germany, August 20-25, 2000. DL2000 overlaps with ICCS2000 which will be held in Darmstadt, Germany, August 13-18, 2000. There is an agreement with the ICCS organizers that DL-related sessions at the ICCS conference will be scheduled on non-overlapping days.

DL2000 is supported by the Graduiertenkolleg Informatik und Technik of the University of Technology in Aachen (RWTH).
What did OWL ever do for us?

“Profile” after:

**Tuition Fee:** $2,450
What did OWL ever do for us?

Applications before:
What did OWL ever do for us?

Applications after:
What did OWL ever do for us?

**Applications** after:

- eScience, eCommerce, geography, engineering, defence, …
- Major impact in healthcare and life sciences
- Mainstream technology supported by, e.g., **Oracle** 11g
- Increasing impact in business applications
What did OWL ever do for us?

Peter and Ian before:
What did OWL ever do for us?

Peter and Ian after:
Where We Are Now

• OWL (2) ontology language a \texttt{W3C} standard
• OWL (2) based on AI research (in particular DLs)
• Wide range of tools and infrastructure now available
• High profile applications
• Support from mainstream technology vendors
So everybody’s happy?
So everybody’s happy?

Of course not!
It’s Too Complicated
It’s Too Complicated

It is too complicated, and users will never understand it or be able to use it!
It’s Too Complicated

It is too complicated, and users will never understand it or be able to use it!

– Many people are now using it!
– Naive users can manage with a small subset (c.f. SQL, MS-Word, ...)
– “Lite” subsets only useful if they confer some computational advantage
It’s Too Complicated

- Many people are now using it!
- SQL is also very complicated, but naive users can manage with a small subset.
- "Lite" subsets only useful if they confer some real advantage.
It’s Too Complex
It’s Too Complex

Complexity is too high, and it won’t scale!
It’s Too Complex

Complexity is too high, and it won’t scale!

- What do we mean by “scale”?
  - Reasoning with whole web doesn’t make sense
- Even so, scalability is a real problem
  - $SROIQ$ satisfiability/subsumption is 2NEXPTIME-complete
It’s Too Complex

Thanks to: Arthur Gordon, Alison Gurlitz, Stephen Lam and Eugene Moy
It’s Too Complex

So is OWL reasoning doomed to failure?

– High complexity doesn’t mean that **bad** performance is guaranteed
  • Just that we can’t guarantee **good** performance
– Highly optimised implementations (may) work well in practice
– Main problem is relatively low “**robustness**”
  • Optimisations exploit features of **typical** ontologies
  • Small changes in ontology can lead to large changes in performance – “it worked OK yesterday”
– Large **data sets** may also be problematical
– Users/applications can choose tractable subsets (**profiles**) if greater scalability and/or robustness is needed
It’s Too Complex

OWL 2 profiles:

- **OWL 2 EL**
  - polynomial (combined) complexity
  - highly effective “one pass” classification algorithms

- **OWL 2 RL**
  - polynomial (combined) complexity
  - convenient rule-extended database implementation

- **OWL 2 QL**
  - AC^0 (data) complexity (< logspace)
  - highly scalable query rewriting implementation
It should have been based on .........*

* Insert favourite logic/KR-formalism
It should have been based on ..........*

* Insert favourite logic/KR-formalism
It should have been based on ..........*

✔ More natural/intuitive and easy to understand
✔ Can describe arbitrary relational structures
✔ UNA and CWA semantics is more intuitive/appropriate
✔ Better scalability

* Insert favourite logic/KR-formalism
It should have been based on ..........*

✔ More natural/intuitive and easy to understand
✔ Can describe arbitrary relational structures
✔ UNA and CWA semantics is more intuitive/appropriate
✔ Better scalability

✘ Less natural/intuitive and easy to understand
✘ Can’t describe unbounded structures
✘ UNA and CWA inappropriate in Web setting
✘ Poor at dealing with incomplete information

* Insert favourite logic/KR-formalism
It should have been based on ..........*

* Insert favourite logic/KR-formalism
It should have been based on ..........*

- Need to deal with vague concepts, e.g., “tall”
- Information may also be vague/noisy, e.g., the Web
- Strictly extends “crisp” languages (1 = true; 0 = false)

* Insert favourite logic/KR-formalism
It should have been based on .........*

✔ Need to deal with vague concepts, e.g., “tall”
✔ Information may also be vague/noisy, e.g., the Web
✔ Strictly extends “crisp” languages (1 = true; 0 = false)

✘ Developing ontologies may be more difficult
✘ How will fuzzy values be determined/agreed?
✘ Reasoner implementations still prototypical
✘ Practicality still an open question

* Insert favourite logic/KR-formalism
It should have been based on ..........*

* Insert favourite logic/KR-formalism
It should have been based on ...........

- Expressive superset of most other languages
- FOL reasoners now highly capable
  and Specialised reasoners can be used for subsets
- Undecidability not important
  and little different from high complexity

* Insert favourite logic/KR-formalism
It should have been based on ..........*

✔ Expressive superset of most other languages
✔ FOL reasoners now highly capable
  and Specialised reasoners can be used for subsets
✔ Undecidability not important
  and little different from high complexity

✘ Reasoners are *much* less robust
✘ Poor at proving non-subsumption (normal case)
✘ Difficult to recognise subsets
✘ Incomplete answers typically used in unsound way

* Insert favourite logic/KR-formalism
Undecidability -v- High Complexity
Undecidability -v- High Complexity

• Can think of undecidable as a very high complexity class
  – Result is very low robustness of reasoner performance

  Users have to make do with imperfect tests which sometimes fail to yield results” ... “analogous to 404 errors on the Web
Undecidability -v- High Complexity

• Can think of undecidable as a very high complexity class
  – Result is very low robustness of reasoner performance
    Users have to make do with imperfect tests which sometimes fail to yield results” ... “analogous to 404 errors on the Web

• But in practice
  – Even SOTA FOL theorem provers are not very effective for non-theorems/non-subsumption
  – Vast majority tests are non-subsumptions, so answer to most tests is “don’t know” (almost every link gives a 404 error)
  – Users expect/demand (fast and) complete reasoning; otherwise they simply won’t use the reasoner
Incompleteness -v- Incorrectness
Incompleteness -vs- Incorrectness

- Applications often treat failure to prove “yes” as “no”
  - and incomplete reasoners often don’t even distinguish
Incompleteness -v- Incorrectness

- Applications often treat failure to prove “yes” as “no”
  - and incomplete reasoners often don’t even distinguish

Isn’t this just negation as failure?
Incompleteness -v- Incorrectness

- Applications often treat failure to prove “yes” as “no”
  - and incomplete reasoners often don’t even distinguish

Isn’t this just negation as failure?

- Absolutely not!
  - Failure in NAF means failure of entailment
    - $\neg \phi$ is true if $\phi$ is not entailed
  - It doesn’t mean failure of an incomplete reasoner to prove that $\phi$ is entailed
  - Treating “don’t know” as “no” is simply incorrect
It’s Not Expressive Enough
It’s Not Expressive Enough

I need to express .......,* which I can’t express in OWL

* Insert favourite expressive feature
It’s Not Expressive Enough

I need to express .......,* which I can’t express in OWL

✔ There are many things that can’t be expressed in OWL
✔ Some of them would certainly be very useful

* Insert favourite expressive feature
It’s Not Expressive Enough

I need to express .......,* which I can’t express in OWL

✔ There are many things that can’t be expressed in OWL
✔ Some of them would certainly be very useful

✘ It’s too complicated
✘ It’s too complex
✘ It should have been based on .......

* Insert favourite expressive feature
Conclusions?

• There is no “right choice” of ontology language
  “you pays your money, and you takes your choice”

• Standardisation requires some choice

• Claim: OWL was a (not totally un-)reasonable choice:
  – good compromise between expressive power and robust tool performance
  – has allowed for the development of a range of tools, infrastructure and applications that could previously only have been dreamt of
THE END?
Ongoing Research

• Optimisation/Profiles
  – [Kazakov], [Glimm et al], [Faddoul et al], [Savo et al]

• Query answering
  – [Kontchakov et al], [Konev et al], [Baader et al]

• Diagnosis and repair
  – [Horridge et al], [Peñaloza et al]

• Extensions
  – [Motik et al], [Artale et al]

• ...
Ongoing Standardisation Efforts

- Standardised query language
  - SPARQL standard for RDF
  - Currently being extended for OWL, see http://www.w3.org/TR/sparql11-entailment/

- RDF
  - Revision currently being considered, see http://www.w3.org/2009/12/rdf-ws/
Thank you for listening
Thank you for listening

Any questions?