Shared content-vocabularies: Ontologies

- Formal, explicit specification
- of a shared conceptualisation
- Consensual knowledge
- Abstract model of some domain
- machine processable
- concepts, properties, relations, functions
What’s inside an ontology?

- Classes + class-hierarchy
- instances
- slots/values
- inheritance (multiple? defaults?)
- restrictions on slots (type, cardinality)
- properties of slots (symm., trans., ...)
- relations between classes (disjoint, covers)
- reasoning tasks: classification, subsumption
Stack of languages

- **XML**: Surface syntax, no semantics
- **XML Schema**: Describes structure of XML documents
- **RDF**: Datamodel for “relations” between “things”
- **RDF Schema**: RDF Vocabulary Definition Language
- **OWL**: A more expressive Vocabulary Definition Language
Bluffer’s guide to RDF (1)

- **Object -> Attribute -> Value**

  - **triples**

  - Objects are **web-resources**

  - Value is again an Object:
    - triples can be **linked**
    - data-model = graph
Every identifier is a URL
= world-wide unique naming!

Has XML syntax

Any statement can be an object
• graphs can be nested

NYT claims

pers05

ISBN...

<rdf:Description rdf:about="#pers05">
<authorOf>ISBN...</authorOf>
</rdf:Description>
What does RDF Schema add?

- Defines **vocabulary** for RDF
- Organizes this vocabulary in a **typed hierarchy**
  - Class, subClassOf, type
  - Property, subPropertyOf
  - domain, range
Things RDF(S) can’t do

- (in)equality
- enumeration
- boolean algebra
  - Union, complement
- number restrictions
  - Single-valued/multi-valued
  - Optional/required values
- inverse, symmetric, transitive

...
Use Cases for ontologies (from OWL requirements doc.)

- Web portal
  - ontology-based

- Multi-media collections
  - annotating, searching

- Corporate Website
  - knowledge management

- Documentation
  - engineering & design

- Agents & Services

- Ubiquitous computing
  - interoperability
Design Goals for OWL

- Shareable
- Changing over time
- Interoperability between ontologies
- Inconsistency detection (requires a logic)
- Balancing expressivity and complexity
- Ease of use
- Compatible with existing standards
- Internationalisation
Requirements for OWL

- Ontologies are objects on the Web with their own meta-data, versioning, etc.
- Ontologies are extendable
- They contain classes, properties, data-types, range/domain, individuals
- Equality (for classes, for individuals)
- Classes as instances
- Cardinality constraints
- XML syntax
Objectives for OWL

+ layered language
+ complex datatypes
+ digitial signatures
± decidability
± local unique names

- default values
- closed world option
- property chaining
- arithmetic
- string operations
- partial imports
- view definitions
- procedural attachment
Layered language

- **OWL Lite:**
  - Classification hierarchy
  - Simple constraints

- **OWL DL:**
  - Maximal expressiveness
  - While maintaining tractability
  - Standard formalisation

- **OWL Full:**
  - Very high expressiveness
  - Loosing tractability
  - Non-standard formalisation
  - All syntactic freedom of RDF (self-modifying)
**Language Layers**

**OWL Light**
- (sub)classes, individuals
- (sub)properties, domain, range
- conjunction
- (in)equality
- cardinality 0/1
- datatypes
- inverse, transitive, symmetric
- hasValue
- someValuesFrom
- allValuesFrom

**OWL DL**
- Negation
- Disjunction
- Full Cardinality
- Enumerated types

**OWL Full**
- Allow meta-classes etc

**RDF Schema**

- Full
- DL
- Lite
Language Layers: Full

- No restriction on use of vocabulary (as long as legal RDF)
  - Classes as instances (and much more)

- RDF style model theory
  - Reasoning using FOL engines
    - via axiomatisation

  - Semantics should correspond with OWL DL for suitably restricted KBs
Language Layers: DL

Use of OWL vocabulary restricted
- Can’t be used to do “nasty things” (i.e., modify OWL)
- No classes as instances
- Defined by abstract syntax

Standard FOL model theory (definitive)
- Direct correspondence with FOL
- Reasoning via DL engines
- Reasoning for full language via FOL engines
  - No need for axiomatisation (unlike full)
  - Would need built in datatypes for performance
Language Layers: Lite

- No explicit negation or union
- Restricted cardinality (0/1)
- No nominals (oneOf)
- Semantics as in DL
  - Reasoning via DL engines (+datatypes)
    - E.g., FaCT, RACER, Cerebra
### OWL: constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Abbreviation</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>intersectionOf</td>
<td>$C_1 \land \ldots \land C_n$</td>
<td>Human $\land$ Male</td>
</tr>
<tr>
<td>unionOf</td>
<td>$C_1 \lor \ldots \lor C_n$</td>
<td>Doctor $\lor$ Lawyer</td>
</tr>
<tr>
<td>complementOf</td>
<td>$\neg C$</td>
<td>$\neg$Male</td>
</tr>
<tr>
<td>oneOf</td>
<td>${x_1 \ldots x_n}$</td>
<td>{john, mary}</td>
</tr>
<tr>
<td>toClass</td>
<td>$\forall P.C$</td>
<td>$\forall$hasChild.Doctor</td>
</tr>
<tr>
<td>hasClass</td>
<td>$\exists P.C$</td>
<td>$\exists$hasChild.Lawyer</td>
</tr>
<tr>
<td>hasValue</td>
<td>$\exists P.{x}$</td>
<td>$\exists$citizenOf.{USA}</td>
</tr>
<tr>
<td>minCardinalityQ</td>
<td>$\geq n \ P.C$</td>
<td>$\geq 2$hasChild.Lawyer</td>
</tr>
<tr>
<td>maxCardinalityQ</td>
<td>$\leq n \ P.C$</td>
<td>$\leq 1$hasChild.Male</td>
</tr>
<tr>
<td>cardinalityQ</td>
<td>$= n \ P.C$</td>
<td>$= 1$hasParent.Female</td>
</tr>
</tbody>
</table>

+ XML Schema datatypes:
  - int, string, real, etc
## OWL: Axioms

<table>
<thead>
<tr>
<th>Axiom</th>
<th>Abbreviation</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>subClassOf</td>
<td>$C_1 \sqsubseteq C_2$</td>
<td>Human $\sqsubseteq$ Animal $\land$ Biped</td>
</tr>
<tr>
<td>sameClassAs</td>
<td>$C_1 \equiv C_2$</td>
<td>Man $\equiv$ Human $\land$ Male</td>
</tr>
<tr>
<td>subPropertyOf</td>
<td>$P_1 \sqsubseteq P_2$</td>
<td>hasDaughter $\sqsubseteq$ hasChild</td>
</tr>
<tr>
<td>samePropertyAs</td>
<td>$P_1 \equiv P_2$</td>
<td>cost $\equiv$ price</td>
</tr>
<tr>
<td>sameIndividualAs</td>
<td>$x_1 \equiv x_2$</td>
<td>President_Bush $\equiv$ G_W_Bush</td>
</tr>
<tr>
<td>disjointWith</td>
<td>$C_1 \sqsubseteq \neg C_2$</td>
<td>Male $\sqsubseteq \neg$ Female</td>
</tr>
<tr>
<td>differentIndividualFrom</td>
<td>${x_1} \sqsubseteq \neg{x_2}$</td>
<td>{john} $\sqsubseteq \neg$ {peter}</td>
</tr>
<tr>
<td>inverseOf</td>
<td>$P_1 \equiv P_2^{-}$</td>
<td>hasChild $\equiv$ hasParent$^{-}$</td>
</tr>
<tr>
<td>transitiveProperty</td>
<td>$P^+ \sqsubseteq P$</td>
<td>ancestor$^+$ $\sqsubseteq$ ancestor</td>
</tr>
<tr>
<td>uniqueProperty</td>
<td>Thing $\sqsubseteq \leq 1P$</td>
<td>Thing $\sqsubseteq \leq 1$ hasMother</td>
</tr>
<tr>
<td>UnambiguousProperty</td>
<td>Thing $\sqsubseteq \leq 1P^{-}$</td>
<td>Thing $\sqsubseteq \leq 1$ isMotherOf$^{-}$</td>
</tr>
</tbody>
</table>

* Axioms (mostly) reducible to subClass/PropertyOf
Different syntaxes

- **RDF**
  - Official exchange syntax
  - Hard for humans

- **UML**
  - Large user base
  - Masahiro Hori, IBM Japan

- **XML**
  - Not the RDF syntax
  - Better for humans
  - Masahiro Hori, IBM Japan

- **“Abstract” syntax**
  - Human readable/writeable
Things OWL doesn’t do

- default values
- closed world option
- property chaining
- arithmetic
- string operations
- partial imports
- view definitions
- procedural attachment
Illustrating OWL in its abstract syntax
Class(professor partial)
Class(associateProfessor partial academicStaffMember)

DisjointClasses(associateProfessor assistantProfessor)
DisjointClasses(professor associateProfessor)

Class(faculty complete academicStaffMember)
DatatypeProperty(age range(xsd:nonNegativeInteger))

ObjectProperty(lecturesIn)

ObjectProperty(isTaughtBy
domain(course)
  range(academicStaffMember))

SubPropertyOf(isTaughtBy involves)

ObjectProperty(teaches
  inverseOf(isTaughtBy)
domain(academicStaffMember)
  range(course))

EquivalentProperties(lecturesIn teaches)

ObjectProperty(hasSameGradeAs Transitive Symmetric
domain(student)
  range(student))
Individual(949318 type(lecturer))

Individual(949352
type(academicStaffMember)
value(age "39"^^&xsd;integer))

ObjectProperty(isTaughtBy Functional)

Individual(CIT1111
type(course)
value(isTaughtBy 949352)
value(isTaughtBy 949318))

DifferentIndividuals(949318 949352)
DifferentIndividuals(949352 949111 949318)
Class(firstYearCourse partial
   restriction(isTaughtBy allValuesFrom (Professor)))

Class(mathCourse partial
   restriction(isTaughtBy hasValue (949352)))

Class(academicStaffMember partial
   restriction(teaches someValuesFrom (undergraduateCourse)))

Class(course partial
   restriction(isTaughtBy minCardinality(1)))

Class(department partial
   restriction(hasMember minCardinality(10))
   restriction(hasMember maxCardinality(30)))
Class(course partial complementOf(staffMember))

Class(peopleAtUni complete unionOf(staffMember student))

Class(facultyInCS complete intersectionOf(faculty restriction(belongsTo hasValue (CSDepartment))))

Class(adminStaff complete intersectionOf(staffMember complementOf(unionOf(faculty techSupportStaff))))
EnumeratedClass(weekdays Monday Tuesday Wednesday Thursday Friday Saturday Sunday)