An Example OWL Ontology
ENC 2004, September 2004
Peter F. Patel-Schneider

Three Variants of OWL

- OWL Full
  - an extension of RDF
  - allows for classes as instances, modification of RDF and OWL vocabularies
- OWL DL
  - the part of OWL Full that fits in the Description Logic framework
  - known to have decidable reasoning
- OWL Lite
  - a subset of OWL DL
  - easier for frame-based tools to transition to
  - easier reasoning

Two Syntaxes for OWL

- RDF/XML documents
  - so that OWL is part of the Semantic Web
  - so that OWL can be an extension of RDF
  - so that RDF applications can parse OWL
- "abstract" syntax
  - easier to read and write manually
  - corresponds more closely to Description Logics and Frames
Living in the Semantic Web and World Wide Web

- Names in OWL are RDF URI references
  - e.g., http://cohse.semanticweb.org/ontologies/people#pet
  - Often (informally) abbreviated via XML qualified names
    - e.g., pp:pet
- Data items belong to XML Schema datatypes
  - E.g., XML Schema integers and strings
  - Generally written in RDF/XML form
    - E.g., "7"8sd:integer, "Susan"8sd:string

How is OWL Used

1. Build an ontology
   - Create the ontology
   - Name classes and provide information about them
   - Name properties and provide information about them
   - (Would be slightly inaccurate to say "define" here)
2. State facts about a domain
   - Provide information about individuals
3. Reason about ontologies and facts
   - Determine consequences of what was built and stated

Creating Ontologies

- Information in OWL is generally in an ontology
  - Ontology—"A branch of metaphysics concerned with the nature and relations of being" [Merriam-Webster Dictionary]
  - An ontology determines what is of interest in a domain and how information about it is structured
  - An OWL ontology is just a collection of information, generally mostly information about classes and properties
- Ontology([name] ...)
- Ontologies can include (import) information from other ontologies
  - Ontology([name] owl:imports(<name>) ...)

Classes

- What is a Class?
  - E.g., person, pet, old
  - A collection of individuals (object, things, ...)
  - A way of describing part of the world
  - An object in the world (OWL Full)
Example Classes

Class(pp:animal partial restriction(pp:eats someValuesFrom(owl:Thing)))
Class(pp:person partial pp:animal)
Class(pp:man complete intersectionOf(pp:person pp:male pp:adult))
Class(pp:animal+lover complete intersectionOf(pp:person restriction(pp:has_pet minCardinality(3))))

Example Classes

Class(pp:vegetarian complete intersectionOf(pp:animal restriction(pp:eats allValuesFrom(complementOf(pp:animal)))
restriction(pp:eats allValuesFrom(complementOf(restriction(pp:part_of someValuesFrom(pp:animal)))))))
DisjointClasses(pp:young pp:adult)

Properties

- What is a Property?
  - e.g., has_father, has_pet, service_number
  - a collection of relationships between individuals (and data)
  - a way of describing a kind of relationship between individuals
  - an object in the world (OWL Full)

Example Properties

ObjectProperty(pp:eaten_by)
ObjectProperty(pp:eats inverseOf(pp:eaten_by)
domain(pp:animal))
ObjectProperty(pp:has_pet domain(pp:person)
range(pp:animal))
ObjectProperty(pp:is_pet_of inverseOf(pp:has_pet))
DataProperty(pp:service_number range(xsd:integer))
SubPropertyOf(pp:has_pet pp:likes)
Individuals

- objects in the world
- belong to classes
- are related to other objects and to data values via properties

Example Individuals

Individual( pp:Tom type(owl:Thing))
Individual( pp:Dewey type( pp:duck))
Individual( pp:Rex type( pp:dog) value( pp:is_pet_of pp:Mick))
Individual( pp:Mick type( pp:male)
  value( pp:reads pp:Daily+Mirror)
  value( pp:drives pp:Q123+ABC))
Individual( pp:The42 type( pp:bus)
  value( pp:service_number "42"\xsd:integer))

The OWL View of Life

OWL is not like a database system
- no requirement that the only properties of an individual are those mentioned in a class it belongs to
- no assumption that everything is known
  - How many pets does Mick have? (Answer: at least one)
- classes and properties can have multiple “definitions”
- statements about individuals need not be together in a document

Using OWL (Building Ontologies)

- determine how the world (domain) should work
  - determine the classes and properties in the domain
  - determine domains and ranges for properties
  - determine characteristics of classes
  - add individuals and relationships as necessary
    - some individuals belong here
    - iterate until “good enough”
    - package all this into an ontology
  - hope that someone else has done most of the work
    * just import all that work
- build the OWL ontology
  - ask whether the ontology is consistent
  - ask whether the classes are coherent
Using OWL (for a Particular Task)

- populate the world (for a particular task)
  - determine the individuals needed for the task
  - determine the relationships between individuals
  - often this will be easy
    * information already in some database, etc.
- write the information in OWL
  - ask whether the information is consistent
  - ask whether other information is entailed

What Follows in the Example Ontology

Class(pp:old+lady complete
  intersectionOf(pp:elderly pp:female pp:person))

Class(pp:old+lady partial
  intersectionOf(
    restriction(pp:has_pet allValuesFrom(pp:cat))
    restriction(pp:has_pet someValuesFrom(pp:animal))))

Every old lady must have a pet cat. (Because she must have some pet and all her pets must be cats.)

What Follows in the Example Ontology

Class(pp:cow partial pp:vegetarian)
ObjectProperty(pp:has_pet domain(pp:person)
  range(pp:animal))

Class(pp:mad+cow complete
  intersectionOf(pp:cow restriction(pp:eats
    someValuesFrom(intersectionOf(pp:brain
      restriction(pp:part_of someValuesFrom pp:sheep)))))

There can be no mad cows.
(Because cows, as vegetarians, don't eat anything that is a part of an animal.)

What Follows in the Example Ontology

Individual(pp:Minnie type(pp:elderly) type(pp:female)
  value(pp:has_pet pp:Tom))

Minnie must be a person (because pet owners are human) and thus is an old lady. Thus Tom must be a cat (because all pets of old ladies are cats).
What Follows in the Example Ontology (extended)

Class(pp:animal+lover complete
     intersectionOf(pp:person
                    restriction(pp:has_pet minCardinality(3))))
Individual(pp:Walt type(pp:person)
           value(pp:has_pet pp:Huey)
           value(pp:has_pet pp:Louie)
           value(pp:has_pet pp:Dewey))
DifferentIndividuals(pp:Huey pp:Louie pp:Dewey)

Walt must be an animal lover. Note that stating that Walt is a person is redundant.

What Follows in the Example Ontology

Class(pp:van partial pp:vehicle)
Class(pp:driver partial pp:adult)
Class(pp:driver complete
     intersectionOf(restriction(pp:drives
                                someValuesFrom(pp:vehicle)
                                pp:person))
Class(pp:white+van+man complete
     intersectionOf(pp:man
                    restriction(pp:drives
                                 someValuesFrom(intersectionOf(pp:white+thing pp:van)))))
Class(pp:white+van+man partial
     restriction(pp:reads allValuesFrom pp:tabloid))

Can All This Really be Done?

- quite a bit is going on here
- reasoning in OWL is difficult
- next part of tutorial

What Follows in the Example Ontology

Individual(pp:Q123+ABC type(pp:white+thing) type(pp:van))
Individual(pp:Mick type(pp:male)
           value(pp:reads pp:Daily+Mirror)
           value(pp:drives pp:Q123+ABC))

Mick drives a white van, so he must be an adult (because all drivers are adults). As Mick is male, thus he is a white van man, so any paper he reads must be a tabloid, thus the Daily Mirror is a tabloid.