The OWL2 Web Ontology Language

G. Falquet

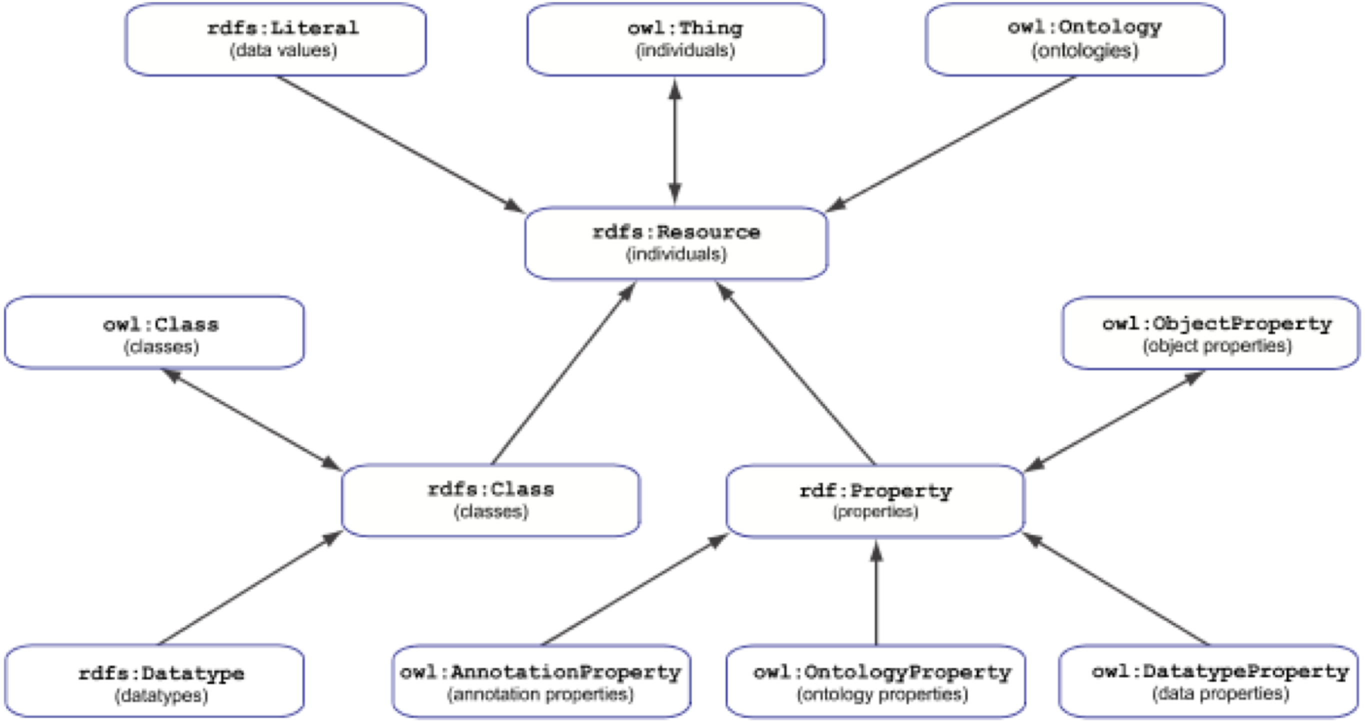
OWL2

- A language in the Description Logics family
- with
  - A rich set of class constructors and property constructors
  - Several relations to define axioms
    - on classes (subclass, equivalent, disjoint, ...)
    - on properties (transitive, functional, ...)
    - on individuals (same, different)
- Can be expressed in RDF (https://www.w3.org/TR/owl2-mapping-to-rdf/)

Knowledge representation primitives in OWL

- individuals
- classes
- datatypes
- properties
  - object properties
  - datatype properties

- axioms


rdfs:Literal
(data values)
owl:Thing
(individuals)
owl:Ontology
(ontologies)
rdfs:Resource
(individuals)
owl:Class
(classes)
owl:ObjectProperty
(object properties)
rdfs:Class
(classes)
rdf:Property
(properties)
rdfs:Datatype
(datatypes)
owl:AnnotationProperty
(annotation properties)
owl:OntologyProperty
(ontology properties)
owl:DatatypeProperty
(data properties)

Class Constructors

Class Constructor	in RDF/Turtle	
◦ the top class:	owl:Thing	
◦ the bottom (impossible) class:	owl:Nothing	(always empty)
◦ a class name	C a owl:Class	
◦ class conjunction	C a owl:Class ; owl:intersectionOf (C1 C2)	
◦ a disjunction:	C a owl:Class ; owl:unionOf (C1 C2)	
◦ a complement:	C a owl:Class ; owl:complementOf C1	

Examples

```
:BachelorStudent a owl:Class .:MasterStudent a owl:Class . :PhDStudent a owl:Class .

:Student a owl:Class;
        owl:unionOf (:BachelorStudent :MasterStudent PhDStudent) .

:BandMStudent a owl:Class;
        owl:intersectionOf (:BachelorStudent :MasterStudent) .

:PhDOnlyStudent a owl:Class ;
        owl:intersectionOf (
                :PhDStudent
                [owl:complementOf
                        [owl:unionOf (BachelorStudent MasterStudent)]])
```

Semantics (simplified)*

If a graph E contains C owl:intersectionOf (C1 C2)

An interpretation of E must satisfy

$$I(C) = I(C1) \cap I(C2)$$

Consequence:

E + x rdf:type C, C owl:intersectionOf (C1 C2)

entails E + x rdf:type C1 and x rdf:type C2

and vice versa

* the formal definition is in https://www.w3.org/TR/owl2-rdf-based-semantics/

Semantics

If a graph E contains C owl:unionOf (C1 C2)

An interpretation of E must satisfy

$$I(C) = I(C1) \cup I(C2)$$

Consequence:

E + x rdf:type C1 or x rdf:type C2 (or both)

entails E + x rdf:type C, C owl:unionOf (C1 C2)

but not vice-versa

Semantics

If a graph E contains C owl:complementOf D

An interpretation must satisfy

$$I(C) = \text{Universe} \setminus I(D)$$

Consequence

x rdf:type C. x rdf:type D .

is inconsistent.

More Class Constructors

existential restriction:

C a ow:Restriction; C owl:onProperty P; owl:someValuesFrom D

universal restriction

C a ow:Restriction; C owl:onProperty P; owl:allValuesFrom D

Semantics (simplified)

If a graph E contains

C a owl:Restriction; owl:onProperty P; owl:someValuesFrom D

An interpretation of E must satisfy

for all x in I(C) there exists y in I(D) such that (x, y) is in I(R)

Consequence:

- E + C a owl:Restriction; owl:onProperty P; owl:someValuesFrom D
- *y* rdf:type D. x P y.

  entails
- y rdf:type C

Semantics (simplified)

If a graph E contains

C a owl:Restriction; owl:onProperty P; owl:allValuesFrom D

An interpretation of E must satisfy

for all x in I(C) if x P y then y is in I(D)

Consequence:

- E + C a owl:Restriction; owl:onProperty P; owl:allValuesFrom D
- *x* rdf:type C. x P y.
  entails
- y rdf:type D

Class Axioms

C rdfs:subClassOf D	x in I(C) $\rightarrow$ x in I(D)
C owl:disjointWith D	x in I(C) $\rightarrow$ not x in I(D)
C owl:equivalentClass D	x in I(C) $\leftrightarrow$ x in I(D)

Example

```
:Human rdfs:subClassOf
        [a owl:Restriction; owl:onProperty :hasOffspring; owl:allValuesFrom :Human] .

 [a owl:Restriction; owl:onProperty :hasOffspring; owl:allSomeValuesFrom :Cat]
        rdfs:subClassOf :Cat .

:Bob a :Human . :Bob :hasOffspring :Carl .
:Miki a :Cat . :Felix :hasOffspring :Miki .
```

entails

```
:Carl a :Human
:Felix a :Cat
```

but ...

```
:Human rdfs:subClassOf
        [a owl:Restriction; owl:onProperty :hasOffspring; owl:allValuesFrom :Human] .

[a owl:Restriction; owl:onProperty :hasOffspring; owl:allSomeValuesFrom :Cat]
        rdfs:subClassOf :Cat .

:Carl a :Human . :Bob :hasOffspring :Carl .
:Felix a :Cat . :Felix :hasOffspring :Miki .
```

does not entail

```
:Bob a :Human
:Miki a :Cat
```

Property axioms

- P rdfs:subPropertyOf Q
- P owl:propertyDisjointWith Q
- P owl:equivalentProperty Q
- P owl:inverseOf Q
- P owl:propertyChainAxiom (Q1 Q2 ... Qn)

Property Characteristics

- owl:FunctionalProperty — x P y1, x P y2 $\rightarrow$ y1 = y2
- owl:InverseFunctionalProperty — x1 P y, x2 P y $\rightarrow$ x1 = x2
- owl:ReflexiveProperty
- owl:IrreflexiveProperty
- owl:SymmetricProperty
- owl:AsymmetricProperty
- owl:TransitiveProperty

Individual Axioms

- owl:sameAs
- owl:differentFrom

There is no unique name assumption
-> different IRIs may be interpreted as the same thing

Examples

:livesIn a owl:FunctionalProperty .

:Bob :livesIn :Geneva .

:Bob :livesIn :Zurich .

is consistent.

It entails

:Geneva :sameAs :Zurich