KDI : Knowledge and Data Integration

OWL

W8.L15.M5.T15.1.2

Contents

Requirements for Ontology Languages

Ontology languages allow users to write explicit, formal conceptualizations of domain models (i.e. formal ontologies). The main requirements are:-

- A well-defined formal syntax
- Sufficient expressive power, and convenience of expression
- Formal semantics, and support for efficient reasoning
- A good tread-off between expressivity and efficiency

OWL (Web Ontology Language) has been designed to meet these requirements for the specification of ontologies and to reason about them and their instances

Reasoning capabilities required

- **Class membership:** If x is an instance of a class C, and C is a subclass of D, then we can infer that x is an instance of D
- **Equivalence of classes:** If class A is equivalent to class B, and class B is equivalent to class C, then A is equivalent to C
- **Disjointness and Consistency:** Determine that if the classes A and B are disjoint there cannot be individuals x which are instances of both A and B. This is an indication of an error in the ontology.
- **Classification:** Certain property-value pairs are a sufficient conditions for membership in a class A; if an individual x satisfies such conditions, we can conclude that x must be an instance of A.

Limitations in the expressive power of RDF schema

- **Range restrictions:** We cannot declare range restrictions that apply to some classes only (e.g. cows eat only plants, while other animals may eat meat too).
- **Disjointness of classes:** We cannot declare that two classes are disjoint (e.g. male and female).
- **Combinations of classes:** We cannot define new classes as union, intersection, and complement of other classes (e.g. person is the disjoint union of the classes male and female).
- **Cardinality restrictions:** We cannot express restrictions in the number of relations (e.g. a person has exactly two parents, a course is taught by at least one lecturer)
- **Meta-properties:** Transitive property (e.g. “greater than”), Unique property (e.g. “is mother of”), Inverse property (e.g. “eats” and “is eaten by”).

Contents

OWL RDF/XML Syntax

HEADER

```
<rdf:RDF
        xmlns:owl ="http://www.w3.org/2002/07/owl#"
        xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
        xmlns:xsd ="http://www.w3.org/2001/XLMSchema#">
```

ONTOLOGY

```
<owl:Ontology rdf:about="">
        <rdfs:comment>An example OWL ontology
</rdfs:comment>
        <owl:priorVersion

rdf:resource="http://www.mydomain.org/uni-ns-old"/>
        <owl:imports
                rdf:resource="http://www.mydomain.org/persons"/>
        <rdfs:label>University Ontology</rdfs:label>
</owl:Ontology>
```

Classes

Defined using owl:Class that is a subclass of rdfs:Class
owl:Thing is the most general class, which contains everything
owl:Nothing is the empty class

DISJOINT CLASSES *owl:disjointWith*

```
<owl:Class rdf:about="#associateProfessor">
        <owl:disjointWith rdf:resource="#professor"/>
        <owl:disjointWith rdf:resource="#assistantProfessor"/>
</owl:Class>
```

EQUIVALENT CLASSES *equivalentClass*

```
<owl:Class rdf:ID="faculty">
        <owl:equivalentClass rdf:resource= "#academicStaffMember"/>
</owl:Class>
```

Properties

Data type properties relate objects to datatype values (ATTRIBUTES)

```
<owl:DatatypeProperty rdf:ID="age">
        <rdfs:range rdf:resource= "http://www.w3.org/2001/XLMSchema
                #nonNegativeInteger"/>
</owl:DatatypeProperty>
```

Object properties relate objects to other objects (RELATIONS)

```
<owl:ObjectProperty rdf:ID="isTaughtBy">
        <owl:domain rdf:resource="#course"/>
        <owl:range rdf:resource= "#academicStaffMember"/>
        <rdfs:subPropertyOf rdf:resource="#involves"/>
```

Property restrictions: a kind of class description (I)

VALUE CONSTRAINT *owl:allValuesFrom*

A value constraint puts constraints on the range of the property. It corresponds to universal quantification.

```
<owl:Class rdf:about="#firstYearCourse">
        <rdfs:subClassOf>
          <owl:Restriction>
                <owl:onProperty rdf:resource="#isTaughtBy"/>
                <owl:allValuesFrom rdf:resource="#Professor"/>
          </owl:Restriction>
        </rdfs:subClassOf>
</owl:Class>
```

Property restrictions: a kind of class description (I)

CARDINALITY CONSTRAINT *someValuesFrom / owl:hasValue*

A cardinality constraint puts constraints on the number of values. It corresponds to the existential quantification or can indicate a specific value.

```
<owl:Class rdf:about="#firstYearCourse">
        <rdfs:subClassOf>
          <owl:Restriction>
                <owl:onProperty rdf:resource="#teaches"/>
                <owl:someValuesFrom rdf:resource="#undergraduateCourse"/>
                (or)
                <owl:onProperty rdf:resource= "#isTaughtBy"/>
                <owl:hasValue rdf:resource= "#949352"/>
    </owl:Restriction>
        </rdfs:subClassOf>
</owl:Class>
```

Cardinality restrictions (I)

owl:maxCardinality: It describes a class of all individuals that have at most N semantically distinct values (individuals or data values) for the property.

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#hasParent" />
  <owl:maxCardinality
rdf:datatype="&xsd;nonNegativeInteger">2</owl:maxCardinality>
</owl:Restriction>
```

Cardinality restrictions (I)

owl:minCardinality: It describes a class of all individuals that have at least N semantically distinct values (individuals or data values) for the property.

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#hasParent" />
  <owl:minCardinality
rdf:datatype="&xsd;nonNegativeInteger">2</owl:minCardinality>
</owl:Restriction>
```

Cardinality restrictions (I)

owl:cardinality: It describes a class of all individuals that have exactly N semantically distinct values (individuals or data values) for the property concerned, where N is the value of the cardinality constraint. This construct is redundant in that it can be replaced by a pair of matching owl:minCardinality and owl:maxCardinality constraints with the same value.

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#hasParent" />
  <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">2</owl:cardinality>
</owl:Restriction>
```

Meta-properties

EQUIVALENCE ***owl:equivalentProperty***

x P y implies x Q y

```
<owl:equivalentProperty
        <owl:ObjectProperty rdf:ID="lecturesIn">
        <owl:equivalentProperty rdf:resource="#teaches"/>
</owl:ObjectProperty>
```

NOTE: in RDF we need P rdfs:subPropertyOf Q and Q rdfs:subPropertyOf P

Meta-properties

INVERSE *owl:inverseOf*

x P y implies y Q x

```
<owl:ObjectProperty rdf:ID="teaches">
        <rdfs:range rdf:resource="#course"/>
        <rdfs:domain rdf:resource= "#academicStaffMember"/>
        <owl:inverseOf rdf:resource="#isTaughtBy"/>
</owl:ObjectProperty>
```

Meta-properties

SYMMETRIC *owl:SymmetricProperty*

x P y implies y P x

```
<owl:ObjectProperty rdf:ID="married">
        <rdf:type rdf:resource="&owl;SymmetricProperty"/>
        <rdfs:range rdf:resource="#person"/>
        <rdfs:domain rdf:resource= "#person"/>
</owl:ObjectProperty>
```

Meta-properties

TRANSITIVE *owl:TransitiveProperty*

x P y and y P z implies x P z

```
<owl:ObjectProperty rdf:ID="ancestor">
        <rdf:type rdf:resource="&owl;TransitiveProperty"/>
        <rdfs:range rdf:resource="#person"/>
        <rdfs:domain rdf:resource= "#person"/>
</owl:ObjectProperty>
```

Functional and inverse functional properties

- **FUNCTIONAL PROPERTY (owl:FunctionalProperty):** A functional property is a property that can have only one value as range for any given individual (e.g., hasMother , hasPresident).
- **INVERSE FUNCTIONAL PROPERTY (owl:InverseFunctionalProperty):** It defines a property that cannot have the same value as range for any given individual (e.g., MotherOf , PresidentOf).

Enumerations

It allows a class to be defined by exhaustively enumerating its instances. The class extension of a class described with owl:oneOf contains exactly the enumerated individuals, no more, no less.

```
<owl:Class rdf:ID="weekdays">
        <owl:oneOf rdf:parseType="Collection">
                <owl:Thing rdf:about="#Monday"/>
                <owl:Thing rdf:about="#Tuesday"/>
                <owl:Thing rdf:about="#Wednesday"/>
                <owl:Thing rdf:about="#Thursday"/>
                <owl:Thing rdf:about="#Friday"/>
                <owl:Thing rdf:about="#Saturday"/>
                <owl:Thing rdf:about="#Sunday"/>
        </owl:oneOf>
</owl:Class>
```

Intersection

```
<owl:Class>
 <owl:intersectionOf rdf:parseType="Collection">
  <owl:Class>
   <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#Tosca" />
    <owl:Thing rdf:about="#Salome" />
   </owl:oneOf>
  </owl:Class>
  <owl:Class>
   <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#Turandot" />
    <owl:Thing rdf:about="#Tosca" />
   </owl:oneOf>
  </owl:Class>
 </owl:intersectionOf>
</owl:Class>
```

Union

```
<owl:Class>
 <owl:unionOf rdf:parseType="Collection">
  <owl:Class>
   <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#Tosca" />
    <owl:Thing rdf:about="#Salome" />
   </owl:oneOf>
  </owl:Class>
  <owl:Class>
   <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#Turandot" />
    <owl:Thing rdf:about="#Tosca" />
   </owl:oneOf>
  </owl:Class>
```

Complement

```
<owl:Class>
  <owl:complementOf>
    <owl:Class rdf:about="#Meat"/>
  </owl:complementOf>
</owl:Class>
```

Instances

Instances of classes are declared as in RDF:

```
<rdf:Description rdf:ID="949352">
        <rdf:type rdf:resource= "#academicStaffMember"/>
</rdf:Description>
<academicStaffMember rdf:ID="949352">
        <uni:age rdf:datatype="&xsd;integer">39<uni:age>
</academicStaffMember>
```

Same instances:

```
<rdf:Description rdf:about="#William_Jefferson_Clinton">
  <owl:sameAs rdf:resource="#BillClinton"/>
</rdf:Description>
```

Different instances:

```
<Opera rdf:ID="Nozze_di_Figaro">
  <owl:differentFrom rdf:resource="#Don_Giovanni"/>
</Opera>
```

References

The following references can be consulted:-

- G. Antoniou F. van Harmelen (2004). A Semantic Web Primer (Cooperative Information Systems). MIT Press, Cambridge MA, USA.
- F. Giunchiglia, F. Farazi, L. Tanca, and R. D. Virgilio. The semantic web languages. In Semantic Web Information management, a model based perspective. Roberto de Virgilio, Fausto Giunchiglia, Letizia Tanca (Eds.), Springer, 2009.
- https://www.w3.org/TR/owl2-primer

KDI : Knowledge and Data Integration

W8.L15.M5.T15.1.2

OWL