OWL (sémantika)

ISKM89 Organizace dat - sémantický web | podzim 2023
Zuzana Nevěřilová | Centrum zpracování přirozeného jazyka

Definice třídy (z deskripční logiky)

Koncept (unární predikát, třída, např. Rodič)
Role (binární predikát, vlastnost, např. máDítě)

Třída - agregovaný popis generického objektu skrze vlastnosti a relace s jinými třídami.

Třída Rodič:

- podtřída Člověk
- spojení se třídou Dítě pomocí vlastnosti máDítě
- ...

Instance je členem jedné nebo více tříd (typicky odpovídají individuovým rolím).

RDFS - doména, obor hodnot (class, domain, range)

Doména a obor hodnot (domain, range):

```
ex:Člověk rdf:type rdfs:Class .
ex:Povolání rdf:type rdfs:Class .
ex:máPovolání rdf:type rdf:Property .
ex:máPovolání rdfs:domain ex:Člověk .
ex:máPovolání rdfs:range ex:Povolání .
```

Inference pro individua:

```
ex:Pavel ex:máPovolání ex:Řezník .
⇒
ex:Pavel rdf:type ex:Člověk.
ex:Řezník rdf:type ex:Povolání.
```

Definice třídy v OWL

Na rozdíl od rdfs:Class, owl:Class je množina. Proto lze aplikovat množinové operace.

1. a class identifier (a URI reference)
2. an exhaustive enumeration of individuals that together form the instances of a class
3. a property restriction
4. the intersection of two or more class descriptions
5. the union of two or more class descriptions
6. the complement of a class description

1 Definice třídy referencí

```
<owl:Class rdf:ID="Human"/>
```
⇔
```
ex:Human rdf:type owl:Class .
```

Na rozdíl od RDF(S), nelze mít trojici Human Human Human.

```
<owl:Class rdf:ID="Opera">
  <rdfs:subClassOf rdf:resource="#MusicalWork" />
</owl:Class>
```
⇔
```
ex:Opera rdf:type owl:Class .
ex:MusicalWork rdf:type owl:Class .
ex:Opera rdfs:subClassOf owl:MusicalWork .
```

2 Definice třídy výčtem

Použití vlastnosti owl:oneOf a vyjmenování individuí.

```
<owl:Class>
    <owl:oneOf rdf:parseType="Collection">
       <owl:Thing rdf:about="#Eurasia"/>
       <owl:Thing rdf:about="#Africa"/>
       <owl:Thing rdf:about="#NorthAmerica"/>
       <owl:Thing rdf:about="#SouthAmerica"/>
       <owl:Thing rdf:about="#Australia"/>
       <owl:Thing rdf:about="#Antarctica"/>
    </owl:oneOf>
</owl:Class>
```

Collection je uzavřená struktura, což znamená, že třída není definovaná ničím jiným.

3 Definice třídy restrikcí na vlastnost

Použití třídy owl:Restriction a vlastnosti onProperty:

- omezení hodnoty (value constraint):
  owl:hasValue, owl:someValuesFrom, owl:allValuesFrom
- omezení kardinality (cardinality constraint):
  owl:maxCardinality, owl:minCardinality, owl:cardinality

máDítě rdfs:domain Člověk .
máDítě rdfs:range Člověk .

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#hasParent" />
  <owl:allValuesFrom rdf:resource="#Human"  />
</owl:Restriction>

<owl:Restriction>
    <owl:onProperty rdf:resource="numberOfPlayers"/>
    <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">11</owl:cardinality>
</owl:Restriction>
```

Omezení je vždy v kontextu jedné restrikce (Restriction). Třída je definovaná jako výčet individuí, které splňují podmínky.

4, 5 Definice třídy množinovými operacemi

Množinové operace předpokládají, že třídy jsou rozdílné.

```
<owl:Class rdf:ID="LivingBeing">
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Plant"/>
    <owl:Class rdf:about="#Animal"/>
  </owl:unionOf>
</owl:Class>

<owl:Class rdf:ID="WhiteWine">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Wine" />
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasColor" />
      <owl:hasValue rdf:resource="#White" />
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

6 Definice třídy množinovými operacemi: doplněk (complement)

Doplněk je něco jiného, než explicitní deklarace rozdílnosti (`owl:disjointWith`).

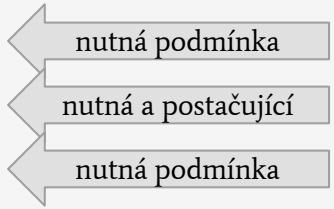
```
<owl:Class>
  <owl:complementOf>
    <owl:Class rdf:about="#Meat"/>
  </owl:complementOf>
</owl:Class>
```

Axiomy tříd (Class Axioms)

```
<owl:Class rdf:ID="Human"/>
```

Správná, avšak ne příliš užitečná definice. Hodí se vypsat nutné a dostačující podmínky.

```
<owl:Class rdf:ID="Human">
  <rdfs:subClassOf rdf:ID="Animal"/>
  <owl:equivalentClass rdf:ID="SmartChimp"/>
  <owl:disjointWith rdf:ID="Chimp"/>
</owl:Class>
```



`owl:equivalentClass` znamená, že třídy znamenají totéž. Dokud nepovažujeme třídy za individua, nelze o nich říct, že `owl:Class1 owl:sameAs owl:Class2.`

Typy vlastností (Property)

- hierarchie `rdf:subPropertyOf`, `rdfs:domain`, `rdfs:range`
- vlastnosti individuí: Object properties - Datatype properties
- ekvivalence vlastností: `owl:equivalentProperty`, `owl:inverseOf`
- (globální) omezení kardinality (cardinality constraints): `owl:FunctionalProperty`
- logické charakteristiky: `owl:SymmetricProperty`, `owl:TransitiveProperty`

Typy vlastností (Property) - ve vztahu k individuím

- **Object properties** - instance `owl:ObjectProperty`
  hodnoty jsou individua (instance tříd)
- **Datatype properties** - instance `owl:DatatypeProperty`
  hodnoty jsou datové hodnoty (data values, literály)
- Annotation properties - anotace (pro lidi) mohou být o individuích, třídách, vlastnostech, ontologiích: label, comment, seeAlso, isDefinedBy
  - žádný vliv na inferenci
- Ontology properties - versionInfo, imports, priorVersion, backwardCompatibleWith, incompatibleWith, ...

Typy vlastností (Property) - rovnost

`owl:equivalentProperty`

Kdy použít?

```
<dc:title> <owl:equivalentProperty> <ns:title>
ns:title rdfs:isDefinedBy
<http://ex.com/vocab>.
```

```
:book ns:title "My book" .
:book dc:title "My book" .
```

Při modelování ontologií využijeme více jmenných prostorů, jejichž sémantika se překrývá.

`owl:sameAs`

Kdy použít?

```
<dc:title> <owl:sameAs> <ns:title>
ns:title rdfs:isDefinedBy
<http://ex.com/vocab>.
```

```
:book ns:title "My book" .
:book dc:title "My book" .
dc:title rdfs:isDefinedBy
<http://ex.com/vocab>.
```

Takto owl:sameAs funguje jen mezi individui, takže tato inference bude fungovat jen při použití OWL Full.

Typy vlastností (Property) - rovnost s obrácenou šipkou

`owl:inverseOf` se používá u dvojic symetrických vlastností.

```
<owl:ObjectProperty rdf:ID="hasChild">
  <owl:inverseOf rdf:resource="#hasParent"/>
</owl:ObjectProperty>
```

`owl:inverseOf` je symetrická, tj. používáme-li inference, není třeba deklarovat

```
:hasParent owl:inverseOf :hasChild
```

Využívá se v dotazování (můžeme se ptát, kdo je něčí rodič, můžeme se ale i ptát, kdo je něčí dítě).

Typy vlastností (Property) - omezení kardinality

`owl:FunctionalProperty` a `owl:InverseFunctionalProperty`

Binární relace R={(x,y)} je funkce právě tehdy, když každé x má maximálně jedno y. Nelze mít trojice x1 :funcProperty :y1 a x1 :funcProperty :y2.

```
<owl:ObjectProperty rdf:ID="husband">
  <rdfs:domain rdf:resource="#Woman" />
  <rdfs:range  rdf:resource="#Man" />
</owl:ObjectProperty>
```

```
<owl:FunctionalProperty rdf:about="#husband" />
```

Je to něco jiného než deklarace třídy pomocí `owl:Restriction` na kardinalitu.

Typy vlastností (Property) - logické charakteristiky

`owl:TransitiveProperty` - tranzitivní relace (nejde dohromady s omezením kardinality)

`owl:SymmetricProperty` - symetrie

```
<owl:SymmetricProperty rdf:ID="friendOf">
  <rdfs:domain rdf:resource="#Human"/>
  <rdfs:range  rdf:resource="#Human"/>
</owl:SymmetricProperty>
```

Individua = instance tříd

O individuích (instancích) se dá říci, zda jsou stejná či jiná:

- owl:sameAs
- owl:differentFrom - pro dvojici
- owl:AllDifferent - pro seznam

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

V OWL Full jsou třídy (také) individua.

Instance vlastností

Sue věří, že kočky mají čtyři nohy.

Reifikace

```
:statement1 a rdf:Statement ;
    rdf:subject :kočka;
    rdf:predicate :has_part;
    rdf:object    [
            a :noha;
            :cardinality "4"
    ].
:sue :believes :statement1.
```

Sue věří, že kočky mají čtyři nohy.

Vlastnost singleton

```
:kočka :has_part1 [
            a :noha;
            :cardinality "4"
        ].
:has_part1 rdf:singletonPropertyOf :has_part.
:sue :believes :has_part1.
```

Reifikace a singleton vlastnosti

Obojí je W3C standard, později uvidíme možná rozšíření.

Vhodné pro modelování:

- původu tvrzení (provenance, kdo si to myslí?)
- metadat tvrzení (např. kdy si to myslí?)
- spolehlivost tvrzení, důvěra ve zdroj (reliability and trust)

Reifikace musí být modelována obezřetně, problém se může projevit nekonečnými cykly a nekonečnou rekurzí.

Abychom mohli definovat, co je to rekurze, musíme definovat rekurzi.