Übung zur Lehrveranstaltung

Semantic Web Technologies 1

Sebastian Rudolph und Elena Simperl

Wintersemester 2011/12

http://semantic-web-grundlagen.de

Lösung der Übung 4:OWL, Teil 1

Lösung der Aufgabe 4.1

```
<?xml version="1.0"?> <!DOCTYPE rdf:RDF [
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
]>
<rdf:RDF
   xmlns ="http://example.org/"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
```

- Die Klasse Gemüse ist eine Unterklasse von PizzaBelag.

```
<owl:Class rdf:about="Gemüse">
   <rdfs:subClassOf rdf:resource="PizzaBelag"/>
</owl:Class>
```

- Die Klasse Pizza hat keine gemeinsamen Elemente mit der Klasse Pizza.

```
<owl:Class rdf:about="PizzaBelag">
   <owl:disjointWith rdf:resource="Pizza" />
</owl:Class>
```

- Die Individuum Aubergine ist ein Element der Klasse Gemüse.

```
<owl:NamedIndividual rdf:about="Aubergine">
   <rdf:type rdf:resource="Gemüse"/>
</owl:NamedIndividual>
```

- Die abstrackte Rolle hatBelag besteht ausschließlich zwischen Elementen der Klasse Pizza and der Klasse PizzaBelag.

```
<owl:ObjectProperty rdf:about="hatBelag">
   <rdfs:domain rdf:resource="Pizza"/>
   <rdfs:range rdf:resource="PizzaBelag"/>
</owl:ObjectProperty>
```

- Pizzen haben immer mindestens zwei Beläge.

```
<owl:Class rdf:about="Pizza">
   <rdfs:subClassOf>
      <owl:Restriction>
         <owl:onProperty rdf:resource="hatBelag"/>
         <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">
             2
         </owl:cardinality>
      </owl:Restriction>
   </rdfs:subClassOf>
   <owl:disjointWith rdf:resource="PizzaBelag"/>
</owl:Class>
```

- Jede Pizza der Klasse PizzaMargarita hat Tomate als Belag.

```
<owl:Class rdf:about = "PizzaMargaritta">
    <rdfs:subClassOf>
       <owl:Restriction>
          <owl:onProperty rdf:resource="hatBelag"/>
          <owl:someValuesFrom rdf:resource="Tomate"/>
       </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

- Die Klasse VegetarischePizza besteht aus den Elementen, die sowohl in der Klasse PizzaOhneFleisch als auch in der Klasse PizzaOhneFisch sind.

```
 <owl:Class rdf:about="VegetarischePizza">
     <rdfs:subClassOf rdf:resource="PizzaOhneFisch"/>
     <rdfs:subClassOf rdf:resource="PizzaOhneFleisch"/>
  </owl:Class>
oder
   <owl:Class rdf:about="VegetarischePizza">
      <owl:intersectionOf rdf:parseType="Collection">
         <owl:Class rdf:resource="PizzaOhneFisch"/>
         <owl:Class rdf:resource="PizzaOhneFleisch"/>
      </owl:intersectionOf>
   </owl:Class>
```

- Keine Pizza der Klasse PizzaMargarita hat Belag aus der Klasse Fleisch.

```
<owl:Class rdf:about = "PizzaMargaritta">
    <rdfs:subClassOf>
       <owl:Class>
          <owl:complementOf>
             <owl:Restriction>
                <owl:onProperty rdf:resource="hatBelag"/>
                <owl:someValuesFrom rdf:resource="Fleisch"/>
             </owl:Restriction>
          </owl:complementOf>
       </owl:Class>
    </rdfs:subClassOf>
</owl:Class>
```

- Jede Pizza der Klass PizzaSalami hat mindestens ein Belag aus der Klass Salami.

```
<owl:Class rdf:about = "PizzaSalami">
      <rdfs:subClassOf>
         <owl:Class>
            <owl:Restriction>
               <owl:onProperty rdf:resource="hatBelag"/>
               <owl:someValuesFrom rdf:resource="Salami"/>
            </owl:Restriction>
         </owl:Class>
      </rdfs:subClassOf>
</owl:Class>
```

- Eine Pizza enthält die Zutaten seinen Belag.

```
<owl:ObjectProperty rdf:about="enthält">
   <owl:propertyChainAxiom rdf:parseType="Collection">
      <owl:ObjectPropery rdf:resource="hatBelag"/>
      <owl:ObjectProperty rdf:resource="hatZutat"/>
   </owl:propertyChainAxiom>
</owl:ObjectProperty>
```

```
</rdf:RDF>
```

Lösung der Aufgabe 4.2

- Die Rolle hatZutat ist transitiv. ✔
- Die Rolle hatBelag ist funktional. ✗
- Die Rolle hatBelag ist invers funktional. ✔
- Die Rolle hatZutat ist asymmetrisch. ✔

Lösung der Aufgabe 4.3

We need two more role names (Rollennamen) than what is given in the exercise–we need the role `anzeigen` and `kennen`. Using these roles, the solution to each statement is as follows:

1. Jeder, der ehrlich ist und ein Verbrechen verübt hat, zeigt sich selbst an.

```
<owl:Class>
   <owl:intersectionOf rdf:parseType = "Collection">
      <owl:Class rdf:resource = "Ehrlich" />
      <owl:Restriction>
         <owl:onProperty rdf:resource="verübt"/>
         <owl:someValuesFrom rdf:resource="Verbrechen"/>
      </owl:Restriction>
   </owl:intersectionOf>
   <rdfs:subClassOf>
      <owl:Restriction>
         <owl:onProperty rdf:resource="anzeigen"/>
         <owl:hasSelf rdf:datatype="&xsd;boolean">true</owl:hasSelf>
      </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

2. Wer klug und ehrlich ist, verübt kein Verbrechen.

```
<owl:Class>
   <owl:intersectionOf rdf:parseType = "Collection">
      <owl:Class rdf:resource = "Ehrlich" />
      <owl:Class rdf:resource ="Klug" />
   </owl:intersectionOf>
   <rdfs:subClassOf>
      <owl:complementOf>
      <owl:Restriction>
         <owl:onProperty rdf:resource="verübt"/>
         <owl:someValuesFrom rdf:resource="Verbrechen"/>
      </owl:Restriction>
      </owl:complementOf>
    </rdfs:subClassOf>
</owl:Class>
```

3. Bonnie zeigt Clyde nicht an.

```
<owl:Class>
   <owl:oneOf rdf:parseType="Collection">
      <rdf:Description rdf:about="Clyde"/>
   </owl:oneOf>
   <rdfs:subClassOf>
      <owl:Class>
         <owl:complementOf>
            <owl:Restriction>
               <owl:onProperty rdf:resource="anzeigen"/>
                  <owl:someValuesFrom>
                  <owl:Class>
                    <owl:oneOf rdf:parseType="Collection">
                       <rdf:Description rdf:about="Clyde"/>
                    </owl:oneOf>
                  </owl:Class>
                 </owl:someValuesFrom>
             </owl:Restriction>
         </owl:complementOf>
      </owl:Class>
   </rdfs:subClassOf>
</owl:Class>
oder in OWL2

<owl:NegativePropertyAssertion>
   <owl:sourceIndividual rdf:about="Bonnie"/>
   <owl:assertionProperty rdf:about="anzeigen"/>
```

```
   <owl:targeIndividual rdf:about="Clyde"/>
</owl:NegativePropertyAssertion>
```

4. Niemand zeigt einen Menschen an, mit dem gemeinsam er ein Verbrechen verübt hat.
   Not possible as discussed in Tutorial.

5. Clyde hat mindestens 10 Verbrechen verübt.

```
<rdf:Description rdf:about="Clyde">
 <rdf:type>
  <owl:Restriction>
    <owl:onProperty rdf:resource="verübt"/>
    <owl:onClass rdf:resource="Verbrechen"/>
    <owl:maxQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">
              10  </owl:maxQualifiedCardinality>
  </owl:Restriction>
 </rdf:type>
</rdf:Description>
```

6. Bonnie und Clyde haben mindestens ein Verbrechen gemeinsam verübt.
   As partially discussed in tutorial that one has to use Inverse role construct to express such statments.

```
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
     <rdf:Description rdf:about="Bonnie"/>
  </owl:oneOf>
  <rdfs:subClassOf>
     <owl:Restriction>
        <owl:onProperty rdf:resource="verübt"/>
        <owl:someValuesFrom>
             <owl:Restriction>
                 <owl:onProperty>
                   <rdf:Description>
                      <owl:inverseOf rdf:resource="verübt"/>
                   </rdf:Description>
                 </owl:onProperty>
                 <owl:someValuesFrom>
                    <owl:Class>
                        <owl:oneOf rdf:parseType="Collection">
                           <rdf:Description rdf:about="Clyde"/>
                        </owl:oneOf>
```

```
                    </owl:Class>
                 </owl:someValuesFrom>
              </owl:Restriction>
           </owl:someValuesFrom>
        </owl:Restriction>
     </rdfs:subClassOf>
  </owl:Class>
```

What does it mean? We are stating that Bonnie is a person who has commited a crime which has been also commited by (see the inverseOf verübt) Clyde.

7. Wer gemeinsam mit seinem Ehepartner ein Verbrechen verübt hat, der ist nicht ehrlich. Not possible.

8. Jeder, der einen Verdächtigen kennt, ist selbst verdächtig.

```
<owl:ObjectProperty rdf:about="verdächtigt">
   <owl:propertyChainAxiom rdf:parseType="Collection">
      <owl:ObjectProperty rdf:resource="verdächtigt"/>
      <owl:ObjectProperty>
         <owl:inverseOf rdf:resource="kennen"/>
      </owl:ObjectProperty>
   </owl:propertyChainAxiom>
</owl:ObjectProperty>
```