

Knowledge Representation and Semantic Technologies

Exercises on OWL

Riccardo Rosati

Corso di Laurea Magistrale in Ingegneria Informatica

Sapienza Università di Roma

2015/2016

Exercise 1

Write an OWL ontology representing the following statements:

- URI1 and URI2 are classes
- URI3 is a property
- URI4 is an instance of class URI1, and URI5 and URI6 are instances of class URI2
- URI3 has domain URI1 and range URI2
- (URI6,URI4) is an instance of property URI3

Exercise 1: Solution

`Declaration (Class (mys : URI1))`

`Declaration (Class (mys : URI2))`

`Declaration (ObjectProperty (mys : URI3))`

`ClassAssertion (mys : URI1 mys : URI4)`

`ClassAssertion (mys : URI2 mys : URI5)`

`ClassAssertion (mys : URI2 mys : URI6)`

`SubClassOf (`

`ObjectSomeValuesFrom (mys : URI3 owl : Thing)`

`mys : URI1)`

Exercise 1: Solution (continued)

```
SubClassOf (  
  ObjectSomeValuesFrom (  
    ObjectInverseOf (myns:URI3)  
    owl:Thing)  
  myns:URI2)
```

```
ObjectPropertyAssertion (myns:URI3 myns:URI6  
  myns:URI4)
```

Exercise 2

Write an OWL ontology that formalizes knowledge about the domain of people, in particular the classes `person`, `man`, `woman`, and the properties `parent`, `mother`, `father`.

Try to express all the knowledge you have about such classes and properties (e.g.: every man is a person, every woman is a person, every mother is a woman, etc.).

Exercise 2: Solution

SubClassOf (myns:man myns:person) (every man is a person)

SubClassOf (myns:woman myns:person) (every woman is a person)

SubObjectPropertyOf (myns:hasMother myns:hasParent)
(hasMother is a subproperty of hasParent)

SubObjectPropertyOf (myns:hasFather myns:hasParent)
(hasFather is a subproperty of hasParent)

SubClassOf (
 ObjectSomeValuesFrom (
 ObjectInverseOf (myns:hasMother)
 owl:Thing)
 myns:woman) (every mother is a woman)

Exercise 2: Solution (continued)

SubClassOf (

ObjectSomeValuesFrom (

ObjectInverseOf (mys : hasFather)

owl:Thing)

mys : man) (every father is a man)

ClassAssertion (mys : man mys : Joe) (Joe is a man)

ObjectPropertyAssertion (mys : hasMother mys : Joe

mys : Ann) (Ann is the mother of Joe)

Exercise 3

Add to the ontology of Exercise 2 the following information:

- Man and woman are disjoint classes
- Every person has a mother
- Every person has a father
- Every person has exactly two parents
- Every person has a father, who is a man
- Every person has a mother, who is a woman
- Every person has a father and a mother

Exercise 3: Solution

- 1) **DisjointClasses (myns:man myns:woman)** (man and woman are disjoint classes)

- 2) **SubClassOf (**
 myns:person
 ObjectSomeValuesFrom (myns:hasMother owl:Thing))
(every person has a mother)

- 3) **SubClassOf (**
 myns:person
 ObjectSomeValuesFrom (myns:hasFather owl:Thing))
(every person has a father)

Exercise 3: Solution (continued)

4) **SubClassOf (**

mys : person

ObjectExactCardinality (2 mys : hasParent)

(every person has exactly two parents)

5) **SubClassOf (**

mys : person

ObjectSomeValuesFrom (mys : hasFather mys : man)

(every person has a father who is a man)

6) **SubClassOf (**

mys : person

ObjectSomeValuesFrom (mys : hasMother mys : woman)

(every person has a mother who is a woman)

Exercise 3: Solution (continued)

7) **SubClassOf (**

mys : person

ObjectIntersectionOf (

ObjectSomeValuesFrom (mys : hasMother owl : Thing)

ObjectSomeValuesFrom (mys : hasFather owl : Thing)))

(every person has a mother and a father)

Notice that axiom 7) is equivalent to the above pair of axioms 2) and 3)