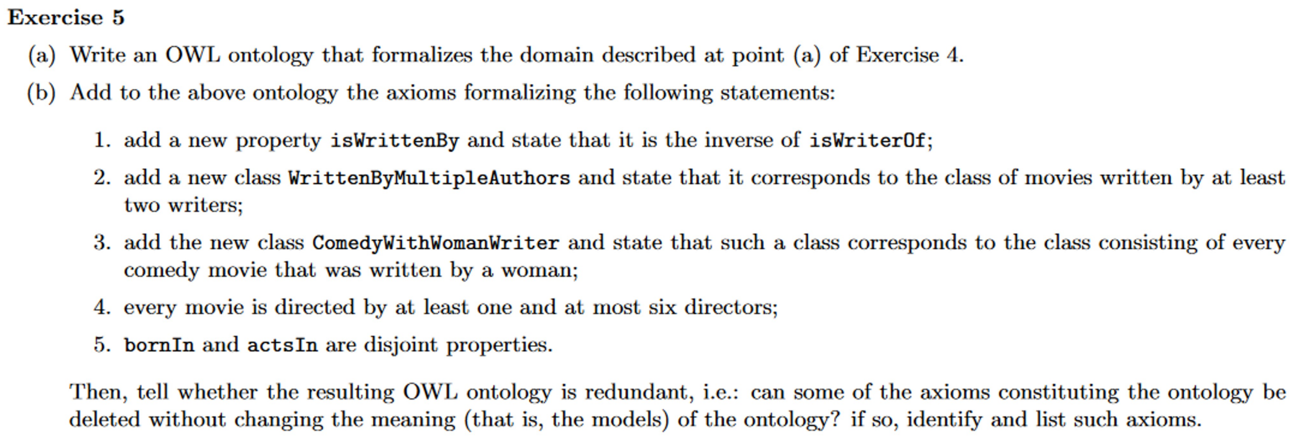
14/06/2016 EX 5

martedì 5 gennaio 2021 16:18



**Exercise 5**

(a) Write an OWL ontology that formalizes the domain described at point (a) of Exercise 4.

(b) Add to the above ontology the axioms formalizing the following statements:

1. add a new property `isWrittenBy` and state that it is the inverse of `isWriterOf`;
2. add a new class `WrittenByMultipleAuthors` and state that it corresponds to the class of movies written by at least two writers;
3. add the new class `ComedyWithWomanWriter` and state that such a class corresponds to the class consisting of every comedy movie that was written by a woman;
4. every movie is directed by at least one and at most six directors;
5. `bornIn` and `actsIn` are disjoint properties.

Then, tell whether the resulting OWL ontology is redundant, i.e.: can some of the axioms constituting the ontology be deleted without changing the meaning (that is, the models) of the ontology? if so, identify and list such axioms.

a)

1) Declaration(Class(myns:Person))
   Declaration(Class(myns:Director))
   Declaration(Class(myns:Writer))
   Declaration(Class(myns:Actor))
   Declaration(Class(myns:Country))
   Declaration(Class(myns:Movie))
   Declaration(Class(myns:Comedy))
   Declaration(Class(myns:Drama))
   Declaration(Class(myns:Man))
   Declaration(Class(myns:Woman))
2) subClassOf(myns:Man myns:Person)
   subClassOf(myns:Woman myns:Person)
3) subClassOf(myns:Comedy myns:Movie)
   subClassOf(myns:Drama myns:Movie)
4) Declaration(ObjectProperty(myns:actsIn))
   Declaration(ObjectProperty(myns:bornIn))
   Declaration(ObjectProperty(myns:filmedIn))
   Declaration(ObjectProperty(myns:isDirectorOf))
   Declaration(ObjectProperty(myns:isWriterOf))
5) subClassOf(ObjectSomeValuesFrom(myns:isDirectorOf owl:Thing) myns:Director)

subClassOf(ObjectSomeValuesFrom(ObjectInverseOf(myns:isDirectorOf) owl:Thing) myns:Movie)

6) subClassOf(ObjectSomeValuesFrom(myns:filmedIn owl:Thing) myns:Movie)
subClassOf(ObjectSomeValuesFrom(ObjectInverseOf(myns:filmedIn) owl:Thing) myns:Country)

7) subClassOf(ObjectSomeValuesFrom(myns:bornIn owl:Thing) myns:Person)
subClassOf(ObjectSomeValuesFrom(ObjectInverseOf(myns:bornIn) owl:Thing) myns:Country)

8) subClassOf(ObjectSomeValuesFrom(myns:actsIn owl:Thing) myns:Actor)
subClassOf(ObjectSomeValuesFrom(ObjectInverseOf(myns:actsIn) owl:Thing) myns:Movie)

9) ObjectPropertyAssertion(myns:isDirectorOf myns:Ann myns:XYZ)
ObjectPropertyAssertion(myns:isWriterOf myns:Ann myns:XYZ)

10) ObjectPropertyAssertion(myns:actsIn myns:Joe myns:ABC)
ObjectPropertyAssertion(myns:actsIn myns:Paul myns:ABC)

11) ObjectPropertyAssertion(myns:filmedIn myns:ABC myns:France)

12) ClassAssertion(myns:Woman myns:Ann)

13) ClassAssertion(myns:Man myns:Paul)

b)

1) InverseObjectProperty(myns:isWrittenBy myns:isWriterOf)

2) Declaration(Class(myns:WrittenByMultipleAuthors))
EquivalentClasses(myns:WrittenByMultipleAuthors ObjectIntersectionOf(myns: Movie ObjectMinCardinality(2 myns:isWrittenBy myns:Writers))

3) Declaration(Class(myns:ComedyWithWomanWriter))
EquivalentClasses(myns:ComedyWithWomanWriter ObjectIntersectOf(myns:Comedy ObjectSomeValuesFrom(myns:isWrittenBy myns:Woman)))

4) subClassOf(myns:Movie ObjectIntersectOf(ObjectMinCardinality(1 ObjectInverseOf(myns:isDirectorOf) myns:Director) ObjectMaxCardinality(6 ObjectInverseOf(myns:isDirectorOf) myns:Director)))
5) DisjointObjectProperties(myns:bornIn myns:actsIn)