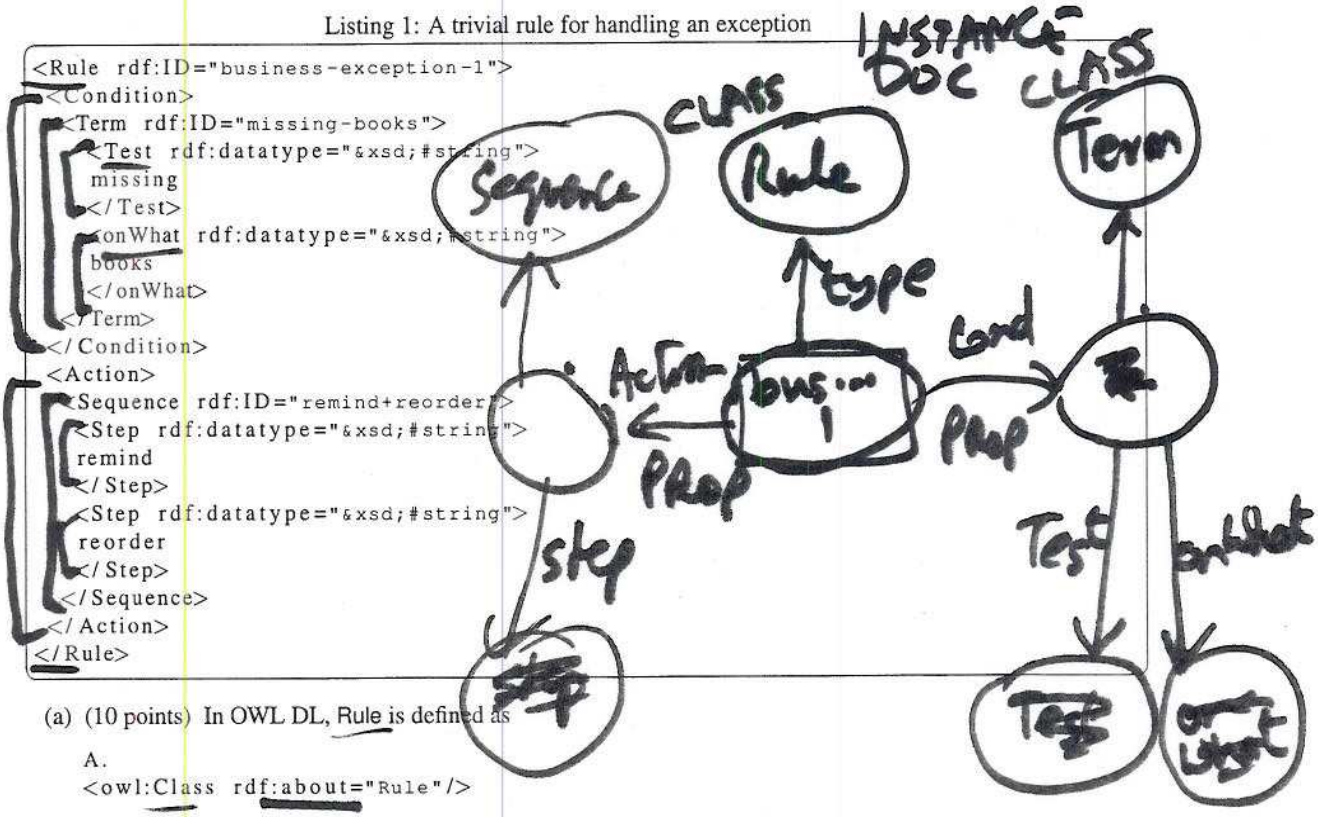
2. Let us model business exception handlers as rules of the form if condition then action. Rules are modeled in OWL. The following trivial rule captures the knowledge that *if the books are missing, then remind and reorder.* This listing presupposes suitable OWL definitions as below.

Listing 1: A trivial rule for handling an exception

```
<Rule rdf:ID="business-exception-1">
 <Condition>
  <Term rdf:ID="missing-books">
   <Test rdf:datatype="&xsd;#string">
    missing
   </Test>
   <onWhat rdf:datatype="&xsd;#string">
    books
   </onWhat>
  </Term>
 </Condition>
 <Action>
  <Sequence rdf:ID="remind+reorder">
   <Step rdf:datatype="&xsd;#string">
    remind
   </Step>
   <Step rdf:datatype="&xsd;#string">
    reorder
   </Step>
  </Sequence>
 </Action>
</Rule>
```



(a) (10 points) In OWL DL, Rule is defined as

A.
```
<owl:Class rdf:about="Rule"/>
```

B.
```
<owl:Class rdf:ID="Rule">
  <rdfs:subClassOf rdf:about='#Condition'/>
</owl:Class>
```

C.
```
<owl:Class rdf:ID="Rule"/>
```

D.
```
<owl:Class rdf:ID="Rule">
  <rdfs:subClassOf rdf:about='#Condition'/>
  <rdfs:subClassOf rdf:about='#Action'/>
</owl:Class>
```

(b) (10 points) In OWL DL, Action is defined as

E.
```
<owl:DatatypeProperty rdf:ID="Action">
  <rdfs:domain rdf:resource="#Rule"/>
  <rdfs:range rdf:resource="#Sequence"/>
</owl:DatatypeProperty>
```

F.
```
<owl:ObjectProperty rdf:ID="Action">
  <rdfs:domain rdf:resource="#Rule"/>
  <rdfs:range rdf:resource="#Sequence"/>
</owl:ObjectProperty>
```

G.
```
<owl:ObjectProperty rdf:ID="Action">
  <rdfs:domain rdf:resource="#Rule"/>
  <rdfs:range rdf:resource="#Step"/>
</owl:ObjectProperty>
```

H.
```
<owl:DatatypeProperty rdf:ID="Action">
  <rdfs:domain rdf:resource="#Rule"/>
  <rdfs:range rdf:resource="&xsd;#string"/>
</owl:DatatypeProperty>
```

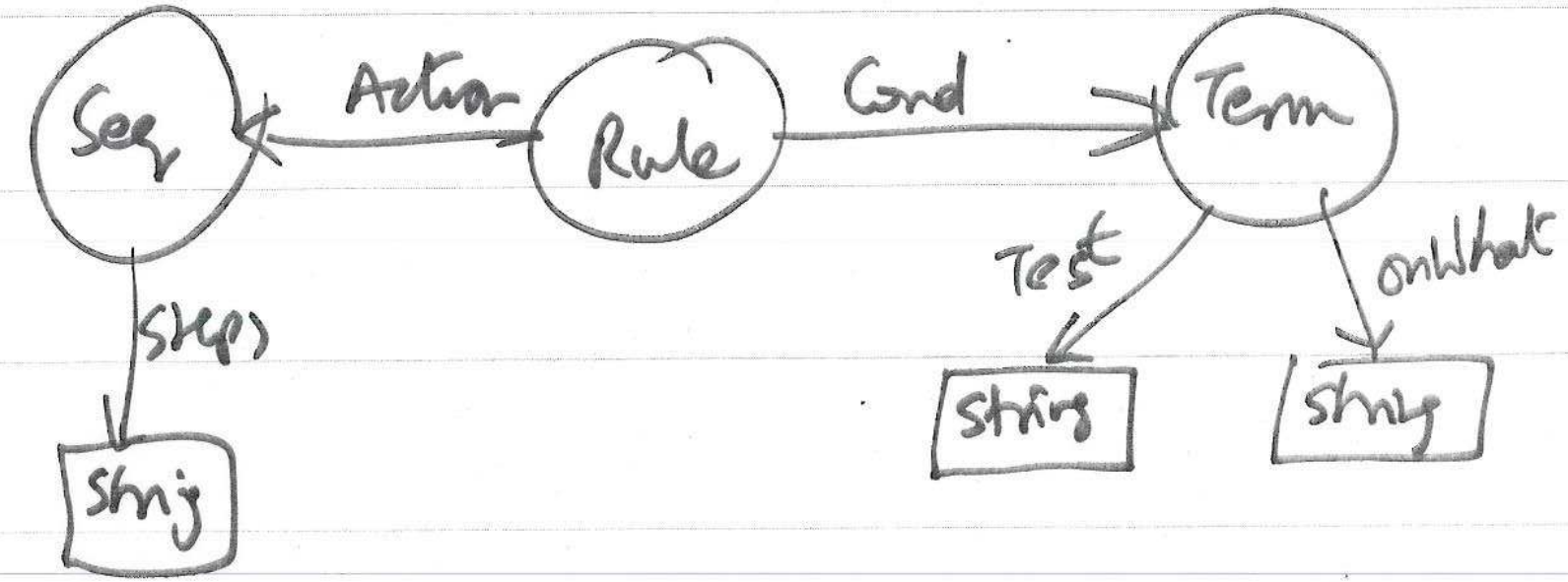
(c) (10 points) In OWL DL, onWhat is defined as

I.
```
<owl:ObjectProperty rdf:ID="onWhat">
  <rdfs:domain rdf:resource="#Term"/>
  <rdfs:range rdf:resource="&xsd;#string"/>
</owl:ObjectProperty>
```

J.
```
<owl:ObjectProperty rdf:ID="onWhat">
  <rdfs:domain rdf:resource="#Term"/>
  <rdfs:range rdf:resource="#Term"/>
</owl:ObjectProperty>
```

K.
```
<owl:Class rdf:ID="onWhat">
  <rdfs:subClassOf rdf:resource="#Term"/>
</owl:Class>
```

L.
```
<owl:DatatypeProperty rdf:ID="onWhat">
  <rdfs:domain rdf:resource="#Term"/>
  <rdfs:range rdf:resource="&xsd;#string"/>
</owl:DatatypeProperty>
```



3. Consider the ontology of Listing 2, which uses the smallerThan and greaterThan properties to enable comparing service providers in terms of some unstated qualities. Based on the usual definition of maximality, we would like to define MaximalProvider as the class of providers that no provider is greater than, and NonMaximalProvider as the class of providers that are not maximal.

Listing 2: A trivial ontology that compares service providers

```
<owl:Class rdf:ID="Comparable"/>
<owl:Class rdf:ID="Provider">
  <rdfs:subClassOf rdf:resource="#Comparable"/>
</owl:Class>

<owl:TransitiveProperty rdf:ID="smallerThan">
  <rdfs:domain rdf:resource="#Comparable"/>
  <rdfs:range rdf:resource="#Comparable"/>
</owl:TransitiveProperty>

<owl:TransitiveProperty rdf:ID="greaterThan">
  <owl:inverseOf rdf:resource="#smallerThan"/>
</owl:TransitiveProperty>
```

(a) (20 points) We can define NonMaximalProvider in OWL DL as

A.

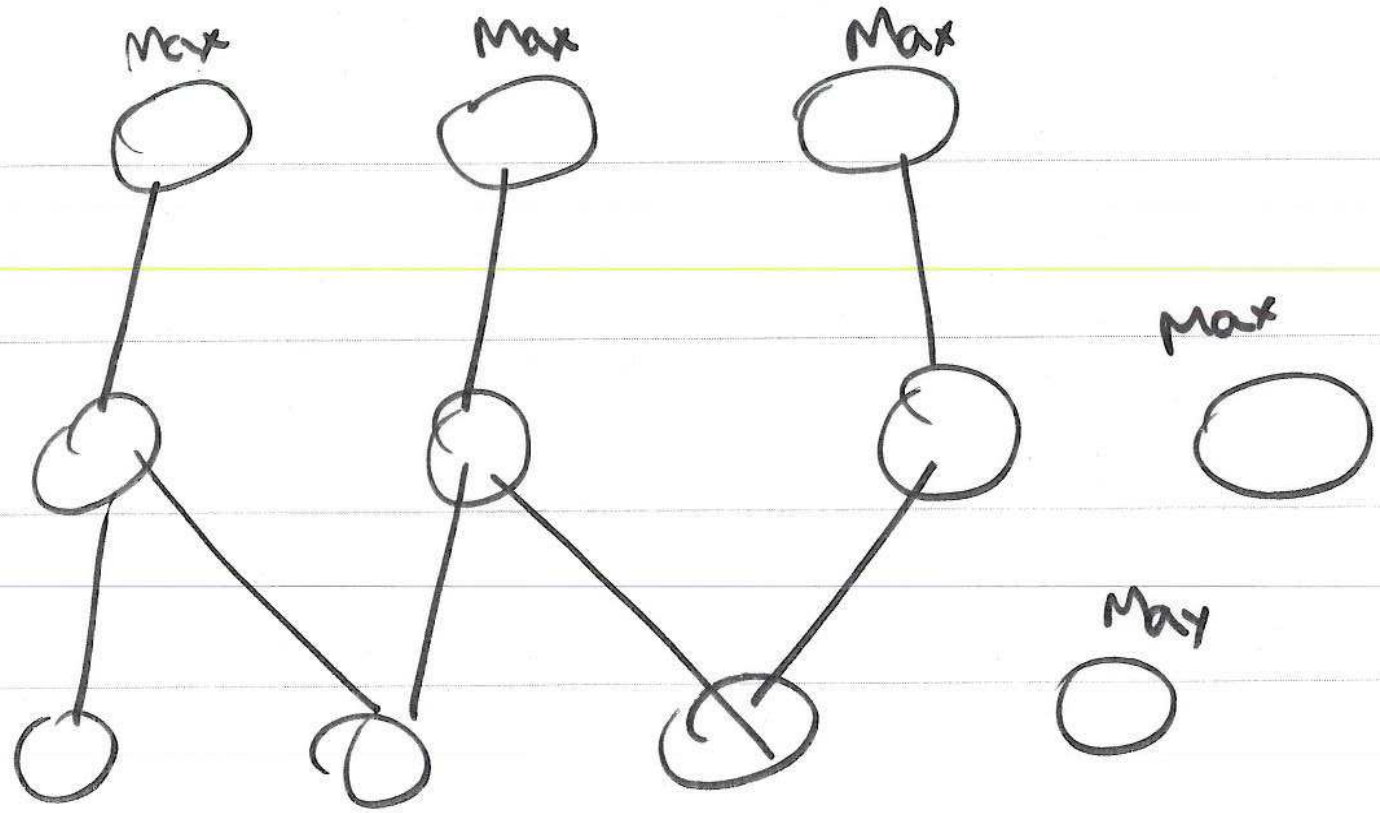
```
<owl:Class rdf:ID="NonMaximalProvider">
  <rdfs:subClassOf rdf:resource="#Provider"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#smallerThan"/>
      <owl:someValuesFrom rdf:resource="#Provider"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

B.

```
<owl:Class rdf:ID="NonMaximalProvider">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Provider"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#smallerThan"/>
      <owl:someValuesFrom rdf:resource="#Provider"/>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

C.

```
<owl:Class rdf:ID="NonMaximalProvider">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Provider"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#greaterThan"/>
      <owl:someValuesFrom rdf:resource="#Provider"/>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```



D.

```
<owl:Class rdf:ID="NonMaximalProvider">
  <rdfs:subClassOf rdf:resource="#Provider"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#smallerThan"/>
      <owl:allValuesFrom rdf:resource="#Provider"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

E.

```
<owl:Class rdf:ID="NonMaximalProvider">
  <rdfs:subClassOf rdf:resource="#Provider"/>
  <owl:disjointWith>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#greaterThan"/>
      <owl:someValuesFrom rdf:resource="#Provider"/>
    </owl:Restriction>
  </owl:disjointWith>
</owl:Class>
```

F.

```
<owl:Class rdf:ID="NonMaximalProvider">
  <rdfs:subClassOf rdf:resource="#Provider"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#smallerThan"/>
      <owl:allValuesFrom rdf:resource="#Provider"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#greaterThan"/>
      <owl:someValuesFrom rdf:resource="#Provider"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

(b) (20 points) We can define MaximalProvider in OWL DL as

G.

```
<owl:Class rdf:ID="MaximalProvider">
  <owl:disjointWith rdf:resource="#NonMaximalProvider"/>
</owl:Class>
```

H.

```
<owl:Class rdf:ID="MaximalProvider">
  <rdfs:subClassOf rdf:resource="#Provider"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#greaterThan"/>
      <owl:allValuesFrom rdf:resource="#Provider"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

I.

```
<owl:Class rdf:about="Provider">
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#MaximalProvider"/>
    <owl:Class rdf:about="#NonMaximalProvider"/>
  </owl:unionOf>
</owl:Class>
```

J.

```
<owl:Class rdf:ID="MaximalProvider">
  <owl:disjointWith rdf:resource="#NonMaximalProvider"/>
</owl:Class>

<owl:Class rdf:about="Provider">
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#MaximalProvider"/>
    <owl:Class rdf:about="#NonMaximalProvider"/>
  </owl:unionOf>
</owl:Class>
```

K.

```
<owl:Class rdf:ID="MaximalProvider">
  <rdfs:subClassOf rdf:resource="#Provider"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#greaterThan"/>
      <owl:someValuesFrom rdf:resource="#Provider"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

L.

```
<owl:Class rdf:ID="MaximalProvider">
  <rdfs:subClassOf rdf:resource="#Provider"/>
  <not>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#smallerThan"/>
      <owl:allValuesFrom rdf:resource="#Provider"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  </not>
</owl:Class>

<owl:Class rdf:about="Provider">
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#MaximalProvider"/>
    <owl:Class rdf:about="#NonMaximalProvider"/>
  </owl:unionOf>
</owl:Class>
```