COMP318

Ontologies and Semantic Web

OWL - Part 4

Dr Valentina Tamma

V.Tamma@liverpool.ac.uk

Where were we

- OWL, a KR language for the web
  - OWL extends RDFS
- Relationship between OWL and DLs
- OWL ontology header and housekeeping information

Terminological knowledge: classes and subclasses

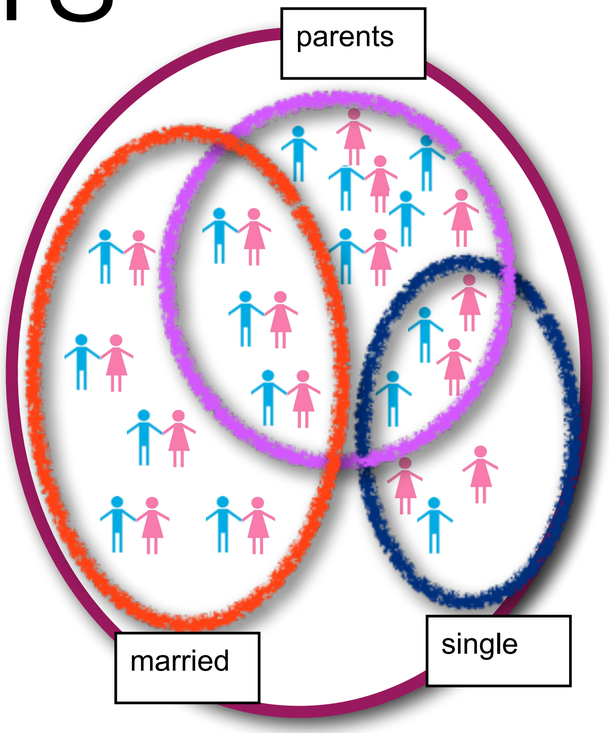
- Classes are defined using owl:class
  - subclass of **`rdfs:class`**

```
<owl:Class rdf:ID=”parents”>
  <rdfs:subClassOf rdf:resource=”#people”/>
</owl:Class>
```

```
<owl:Class rdf:about=”#children”>
  <owl:disjointWith rdf:resource=”#parents”/>
</owl:Class>
<owl:Class rdf:ID=”offspring”>
  <owl:equivalentClass rdf:resource=”#children”/>
</owl:Class>
```

OWL class constructors

- Boolean combinations are used to define classes
  - married and single are disjoint but parent and single are not!




```
Class: Married
  EquivalentTo: not Single
```

Manchester syntax

```
:married  owl:subClassOf [
  rdf:type         owl:Class ;
  owl:intersectionOf  (
                [ rdf:type        owl:Class ;
                  owl:complementOf  :single ] )
 ] .
```

turtle syntax

```
<owl:Class rdf:about=”#married”>
  <owl:subClassOf>
    <owl:Class>
      <owl:complementOf rdf:resource=”#single”/>
    </owl:Class>
  </owl:subClassOf>
  </owl:Class>
```

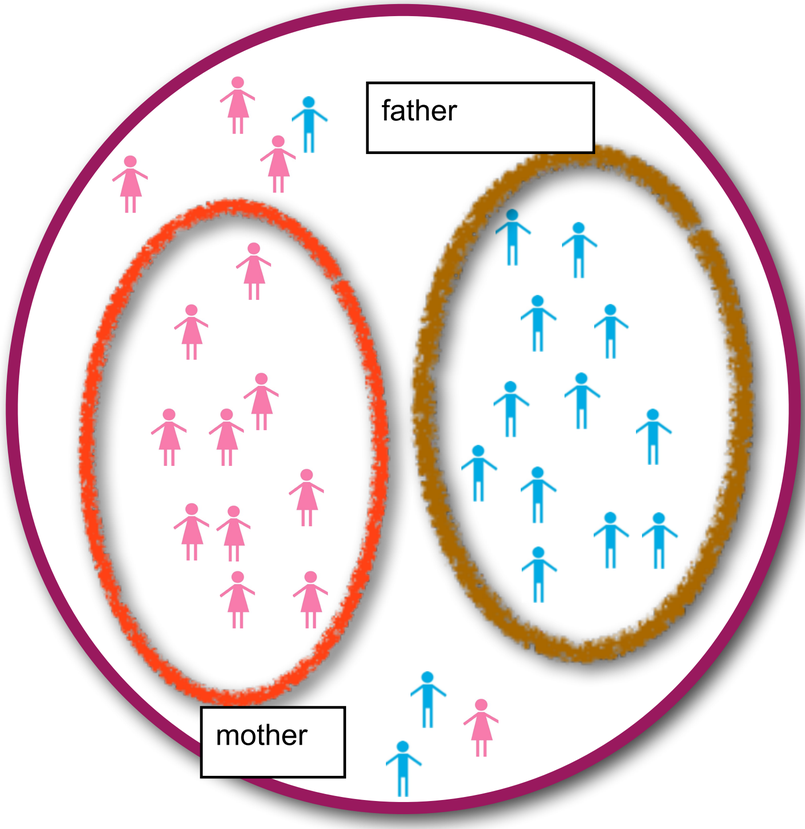
OWL class constructors

- parents are mothers and father

```
<owl:Class rdf:about="Parent">
  <owl:equivalentClass>
   <owl:Class>
    <owl:unionOf rdf:parseType="Collection">
     <owl:Class rdf:about="Mother"/>
     <owl:Class rdf:about="Father"/>
    </owl:unionOf>
   </owl:Class>
  </owl:equivalentClass>
 </owl:Class>
```

```
Class: Parent
  EquivalentTo: Mother or Father
```

```
:Parent owl:equivalentClass [
  rdf:type    owl:Class ;
  owl:unionOf ( :Mother :Father )
] .
```



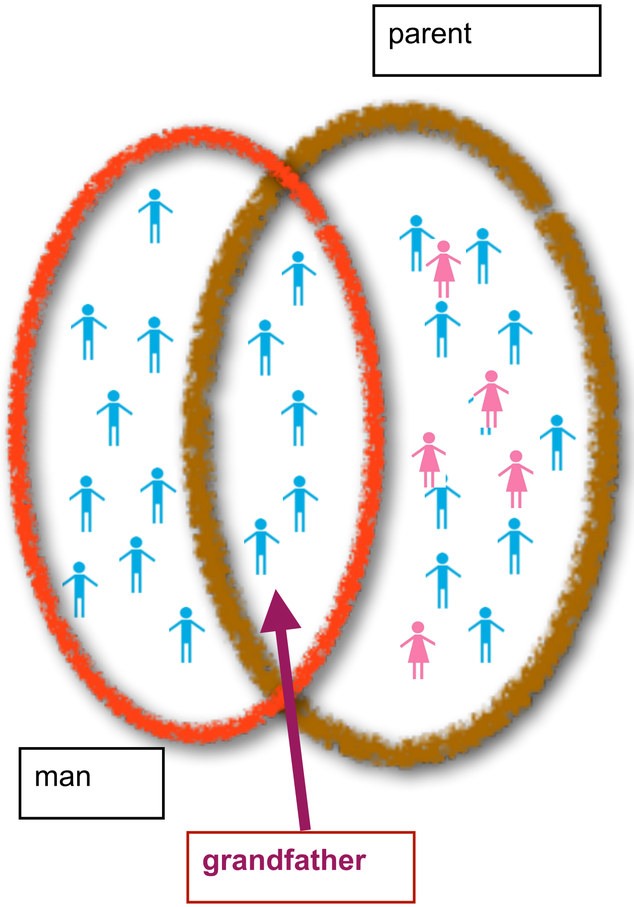
OWL class constructors

- Grandfather is both a man and a father

```
<owl:Class rdf:about="Grandfather">
  <rdfs:subClassOf>
   <owl:Class>
    <owl:intersectionOf rdf:parseType="Collection">
     <owl:Class rdf:about="Man"/>
     <owl:Class rdf:about="Parent"/>
    </owl:intersectionOf>
   </owl:Class>
  </rdfs:subClassOf>
 </owl:Class>
```

```
Class: Grandfather
  SubClassOf: Man and Parent
```

```
:Grandfather  rdfs:subClassOf  [
  rdf:type          owl:Class ;
  owl:intersectionOf  ( :Man  :Parent )
] .
```

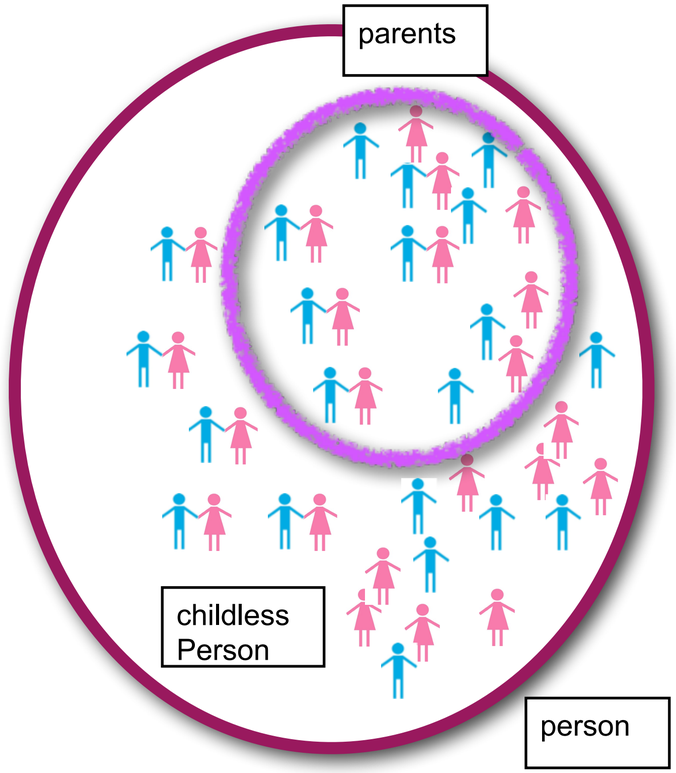


OWL class constructors

- a childless person is someone who is a person but not a parent

```
<owl:Class rdf:about="ChildlessPerson">
  <owl:equivalentClass>
   <owl:Class>
    <owl:intersectionOf rdf:parseType="Collection">
     <owl:Class rdf:about="Person"/>
     <owl:Class>
      <owl:complementOf rdf:resource="Parent"/>
     </owl:Class>
    </owl:intersectionOf>
   </owl:Class>
  </owl:equivalentClass>
 </owl:Class>
```

```
:ChildlessPerson  owl:equivalentClass  [
  rdf:type          owl:Class ;
  owl:intersectionOf  ( :Person
                [ rdf:type        owl:Class ;
                  owl:complementOf  :Parent ] )
 ] .
```




```
Class: ChildlessPerson
  EquivalentTo: Person and not Parent
```

OWL class constructors

- Classes can also be defined through enumeration using **`owl:oneOf`**
  - allows a class to be defined extensionally,
    - with exactly the enumerated individual

```
<owl:Class rdf:about=”#simpsonFamily”>
  <owl:oneOf rdf:parseType=”Collection”>
    <owl:Thing rdf:about=“#marge”/>
    <owl:Thing rdf:about=“#homer”/>
    <owl:Thing rdf:about=“#lisa”/>
    <owl:Thing rdf:about=“#maggie”/>
    <owl:Thing rdf:about=“#bart”/>
  </owl:oneOf>
</owl:Class>
```

```
:simpsonFamily  owl:equivalentClass [
  rdf:type   owl:Class ;
  owl:oneOf  ( :marge, :homer, :lisa, :maggie, ;bart)
 ] .
```

```
Class: simpsonFamily
  EquivalentTo: { marge, homer, lisa, maggie, bart }
```

Recap

- OWL preliminaries
- OWL class constructors

- `https://www.w3.org/TR/owl2-primer/`

COMP318

Ontologies and Semantic Web

End of OWL - Part 4

Dr Valentina Tamma

V.Tamma@liverpool.ac.uk