CEID

MSc on DATA DRIVEN COMPUTING AND DECISION MAKING (DDCDM)

Ontology Web Language: OWL

I. HATZILYGEROUDIS, PROFESSOR EMERITUS

Web Ontology Languages

- They allow writing explicit, rigorous conceptualizations for domain models
- Requirements
  - Well defined syntax
  - Efficient reasoning support
  - Formal semantics
  - Adequate expressiveness
  - Ease of expession

Web Ontology Languages-Reasoning

- Participation in classes
  -If x is an instance of C and C is a subclass of D, then x is an instance of D
- Class equivalence
  -If class A is equivalent to B and B to C, then A is equivalent to C
- Consistency
  -If x is an instance of A, A is a subclass of $B \cap D$, A is a subclass of D, and B and D are disjoint to each other, then we have an inconsistency (which must be detected)
- Classification
  -If particular property-value pairs are a sufficient condition for membership in a class A, then if an individual element x satisfies them it is an instance of A

Web Ontology Languages-Reasoning

- Reasoning support is important because it allows

  -Ontology consistency check

  -Checking for unwanted relationships between classes

  -The automatic classification of instances into classes
- Rigorous semantics and reasoning support is usually ensured by mapping to known rigorous formalisms (eg FOL) and using corresponding automated reasoning methods/tools.
- In the case of OWL, the corresponding rigorous formalism is a description logic (DL) and the corresponding reasoning mechanism one of existing DL reasoners (e.g. Pellet, FaCT, RACER etc.).
- Description logics are subsets of the full first-order logic (FOL) that ensure efficient reasoning support.

OWL-Syntax

- Uses RDF syntax based on XML (RDF/XML)
- There are other syntax formats for OWL:

  -XML based, RDF/XML independent

  -An abstract syntax more compact and readable than its XML and RDF/XML counterparts

  -A graphical syntax based on UML
- An OWL document is an RDF document and is commonly called an *OWL ontology*.

owl:imports has transitive property

OWL-Header

- Root element

```
<rdf:RDF
    xmlns:owl = "http://www.w3.org/2002/07/owl#"
    xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs = "http://www.w3.org/2001/01/rdf-shema#"
    xmlns:xsd = "http://www.w3.org/2001/02/XMLSchema#">
```

- Assertions

```
<owl:Ontology rdf:about = "">
  <rdfs:comment>παράδειγμα οντολογίας OWL</rdfs:comment>
  <owl:priorVersion rdf:resource=http://www.mydomain.org/
                                                uni-ns-old"/>
  <owl:imports rdf:resource=http://www.mydomain.org/persons"/>
  <rdfs:label>University Ontology</rdfs:label>
</owl:Ontology>
```

> owl:imports has implications for the logical meaning of the ontology: it imports other ontologies.

> Το owl:imports έχει μεταβατική ιδιότητα.

OWL-Classes

- They are defined using the owl:Class element.

```
<owl:Class rdf:ID="associateProfessor">
    <rdf:subClassOf rdf:resource="#academicStaffMember"/>
</owl:Class>
```

Superclass of all classes
owl:Thing

```
<owl:Class rdf:about="#associateProfessor">
    <owl:disjointWith rdf:resource="#professor"/>
    <owl:disjointWith rdf:resource="#assistantProfessor"/>
</owl:Class>
```

Subclass of all classes-
empty class
owl:Nothing

```
<owl:Class rdf:ID="faculty">
    <owl:equivalentClass rdf:resource="# academicStaffMember"/>
</owl:Class>
```

OWL-Properties

- Object properties

  -Associate objects with each other (eg isTaughtBy, supervises)

- Data type properties

  -They associate objects with values of a data type (eg phone, title, age).

  -OWL has no predefined data types

  -Allows the use of XML Schema data types.

OWL-Properties

- Examples

Object property

```
<owl:ObjectProperty rdf:ID="isTaughtBy">
    <rdf:domain rdf:resource="#course"/>
    <rdf:range rdf:resource="#academicStaffMember"/>
    <rdfs:subPropertyOf rdf:resource="#involves"/>
</owl:ObjectProperty>
```

More than one domain and range can be declared, in which dcase we take their intersection.

Data type property

```
<owl:DataTypeProperty rdf:ID="age">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema
    #nonNegativeInteger"/>
</owl:ObjectProperty>
```

User data types are collected in an XML schema and then used in an OWL ontology.

OWL-Properties

Reverse properties association

```
<owl:ObjectProperty rdf:ID="teaches">
      <rdf:domain rdf:resource="#academicStaffMember"/>
      <rdf:range rdf:resource="#course"/>
      <owl:inverseOf rdf:resource="#isTaughtBy"/>
</owl:ObjectProperty>
```

Property equivalence

```
<owl:ObjectProperty rdf:ID="lecturesIn">
   <owl:equivalentProperty rdf:resource="#teaches"/>
</owl:ObjectProperty>
```

OWL-Properties

- Property restrictions

```
<owl:Class rdf:about="#firstYearCourse">
    <rdfs:subClassOf>
      <owl:Restriction>
         <owl:onProperty rdf:resource="#isTaughtBy"/>
         <owl:allValuesFrom rdf:resource="#Professor"/>
      </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

The "owl:allValuesFrom" element declares the possible values that the "isTaughtBy" property can take, as instances of the class 'Professor'.

Anonymous superclass

(First year courses are taught only by professors.)

We declare that the class 'firstYearCourse' is a subclass of an anonymous class, which gathers all objects that satisfy some constraints.

OWL-Properties

```
<owl:Class rdf:about="#mathCourse">
    <rdfs:subClassOf>
      <owl:Restriction>
         <owl:onProperty rdf:resource="#isTaughtBy"/>
         <owl:hasValue rdf:resource="#949318"/>
      </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

(All mathematics courses are taught by the teacher with code 949318-e.g. John Hatzis)

The “owl:hasValue” element specifies a specific value that the “isTaughtBy” property should take.

OWL-Properties

```
<owl:Class rdf:about="#academicStaffMember">
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#teaches"/>
        <owl:someValuesFrom rdf:resource="#undergradCourse"/>
      </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

(All members of academic staff must teach at least one undergraduate course)

owl:allValuesFrom → universal quantification

owl:someValuesFrom → existential quantification

OWL-ιδιότητες

```
<owl:Class rdf:about="#department">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasMember"/>
      <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger"/>
        10
      </owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasMember"/>
      <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger"/>
        30
      </owl:maxCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

An "owl:Restriction" element contains an "owl:onProperty" element and one or more restriction declarations.

(A department must have a minimum of 10 and a maximum of 30 members)

OWL-Properties

- owl:Restriction defines an anonymous class, which has no ID and has local scope.
- There are generally two kinds of classes, regular ones defined through owl:Class and having an ID, and local anonymous classes (as collections of objects that satisfy constraints or combinations of classes), as above, also called class expressions.
- An owl:Restriction element contains an owl:onProperty element and one or more constraint declarations (owl:allValuesFrom, owl:someValuesFrom, owl:hasValue, owl:minCardinality, owl:maxCardinality)

OWL-Properties

- Special properties
  - owl:TransitiveProperty (defines a transitive property: "is taller than", "is ancestor of")
  - owl:SymmetricProperty (defines a symetric property : "has same grade as", "is sibling of")
  - owl:FunctionalProperty (defines a property with at most one value for each object: "age", "height")
  - owl:InverseFunctionalProperty (defines a property for which two different objects cannot have the same value: "isTheSocialSecurityNumber")

OWL-Properties

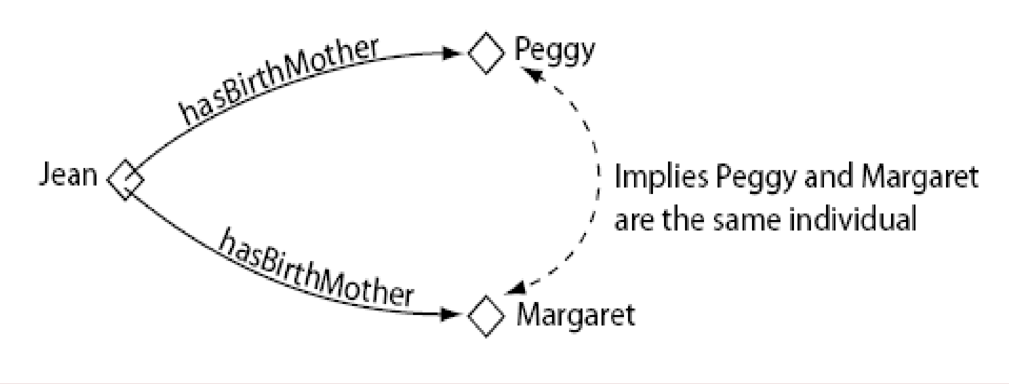
Example

```
<owl:ObjectProperty rdf:ID=“hasSameGradeAs”>
      <rdf:type rdf:resource=“&owl;TransitiveProperty”/>
      <rdf:type rdf:resource=“&owl;SymmetricProperty”/>
      <rdfs:domain rdf:resource=”#student”/>
      <rdfs:range rdf:resource=”#student”/>
</owl:ObjectProperty>
```

OWL-Properties

Functional properties

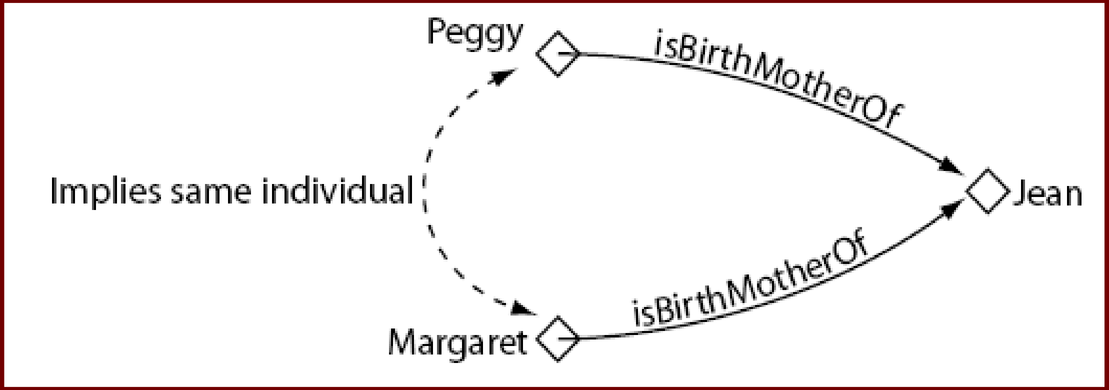
- If a property is 'functional', for a given entity, there can be at most one entity associated with that property.
  - For a given domain, range should be unique
- Functional properties are also known as single value properties.



OWL-Properties

Inverse functional properties

- If a property is 'inverse functional', then its inverse is functional.
  – For a given domain, range should be unique



OWL-Properties

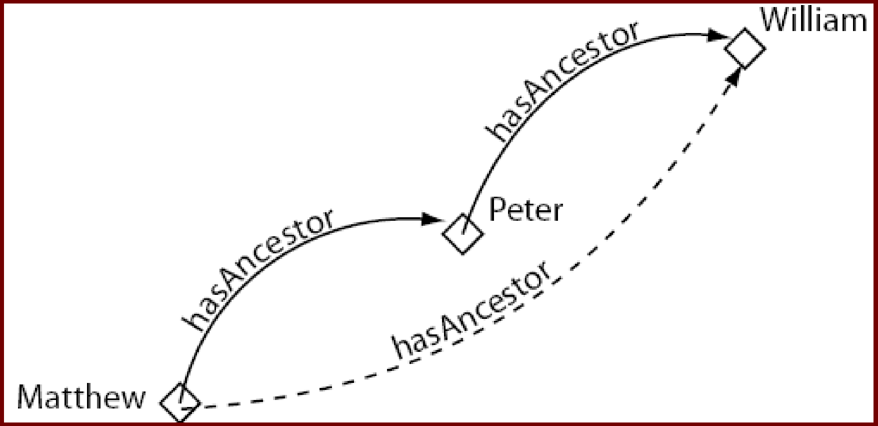
Functional vs Inverse functional properties

	domain	range	example
Functional Property	For a given domain	Range is unique	hasFather: A hasFather B, A hasFather C →B=C
InverseFunctional Property	Domain is unique	For a given range	hasID: A hasID B, C hasID B →A=C

OWL-Properties

Transitive properties

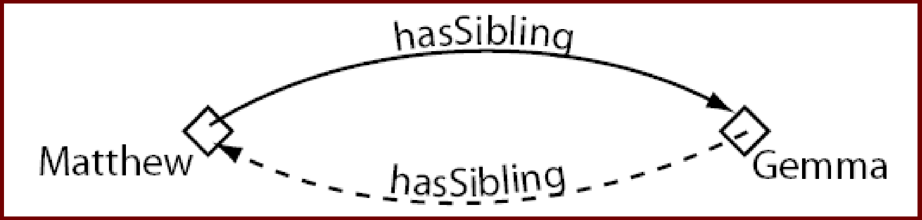
- If a property is transitive and relates entity A to entity B and entity B to entity C, then it is inferred that it also relates entity A to C.



OWL-Properties

Symmetric properties

- If a property is 'symmetric' and relates entity A to entity B, then it is inferred that it also relates entity B to A.



OWL-logical combinations

Logical combinations of classes (union, intersection, complement)

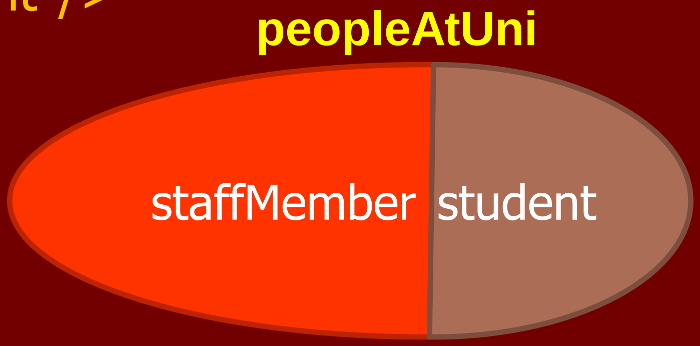
```
<owl:Class rdf:about="#course">
    <rdfs:subClassOf>
      <owl:Class>
        <owl:complementOf rdf:resource="#staffMember"/>
      </owl:Class>
    </rdfs:subClassOf>
</owl:Class>
```

(Each course is an instance of the complement of staff members, i.e. no class is a staff member, i.e. class 'course' and class 'staff member' are foreign to each other.)

(Alternatively, the element owl:disjointWith could be used.)

OWL-logical combinations

```
<owl:Class rdf:ID=“peopleAtUni”>
   <owl:unionOf rdf:parseType = “Collection”>
      <owl:Class rdf:about = “#staffMember”/>
      <owl:Class rdf:about=“#student”/>
   </owl:unionOf>
</owl:Class>
```

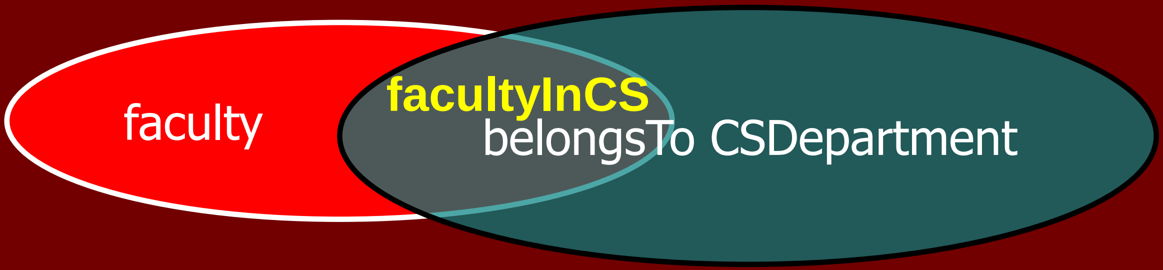


(The new class is not declared to be a subclass of the union, but equal to the union of two classes: case of class equivalence. It also doesn't state that the two classes must be foreign to each other, so a member of ‘staffMember’ can also be member of ‘student’).

OWL-logical combinations

```
<owl:Class rdf:ID="facultyInCS">
   <owl:intersectionOf rdf:parseType = "Collection">
     <owl:Class rdf:about = "#faculty"/>
     <owl:Restriction>
         <owl:onProperty rdf:about = "#belongsTo"/>
         <owl:hasValue rdf:about = "#CSDepartment"/>
     </owl:Restriction>
   </owl:interSectionOf>
</owl:Class>
```

Anonymous class



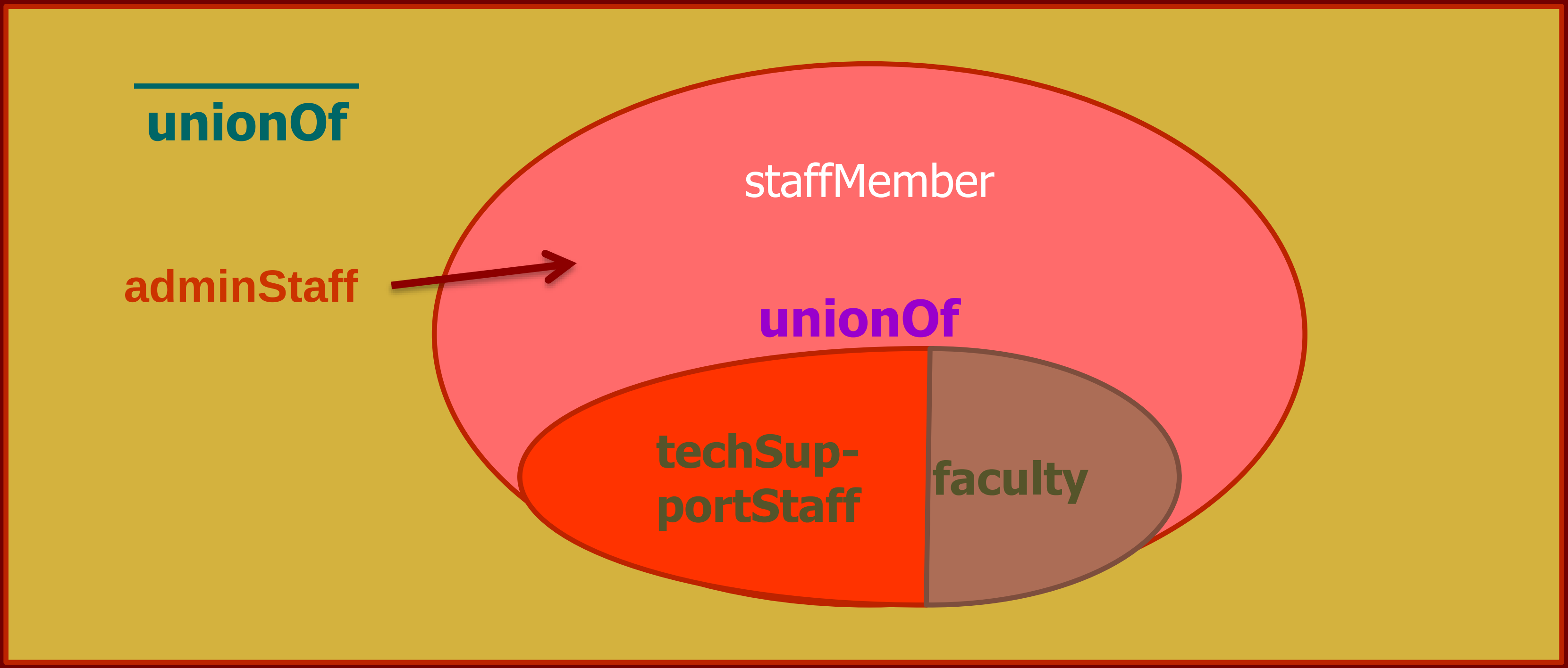
(The intersection of two classes is created, one of which is anonymous - objects belonging to the Department of Computers - and the other the 'faculty', so finally the teaching staff of the Department of Computers is resulted.)

OWL- nested logical operators

```
<owl:Class rdf:ID="adminStaff">
   <owl:intersectionOf rdf:parseType="Collection">
      <owl:Class rdf:about="#staffMember"/>
      <owl:Class>
         <owl:complementOf>
            <owl:Class>
               <owl:unionOf rdf:parseType="Collection">
                  <owl:Class rdf:about="#faculty"/>
                  <owl:Class rdf:about="#techSupportStaff"/>
               </owl:unionOf>
            </owl:Class>
         </owl:complementOf>
      </owl:Class>
   </owl:intersectionOf>
</owl:Class>
```

Administrative staff (adminStaff) are the staff members (staffMember) who are neither teaching (faculty) nor technical staff (techSupportStaff).

OWL- nested logical operators



OWL-enumeration

```
<owl:Class rdf:ID="weekdays">
   <owl:oneOf rdf:parseType = "Collection">
      <owl:Thing rdf:about = "#Monday"/>
     <owl:Thing rdf:about = "#Tuesday"/>
     <owl:Thing rdf:about = "#Wednesday"/>
     <owl:Thing rdf:about = "#Thursday"/>
     <owl:Thing rdf:about = "#Friday"/>
     <owl:Thing rdf:about = "#Saturday"/>
     <owl:Thing rdf:about = "#Sunday"/>
   <owl:oneOf>
</owl:Class>
```

OWL- instances

As in RDF

```
<rdf:Description rdf:ID="949352">
   <rdf:type rdf:resource = "#academicStaffMember"/>
</rdf:Description>
```

or equivalently

```
<academicStaffMember rdf:ID = "949352"/>
```

or with more details

```
<academicStaffMember rdf:ID = "949352"/>
  <uni:age rdf:datatype = "&xsd;integer">39</uni:age>
</academicStaffMember>
```

OWL- instances

- Owl does not adopt the assumption of unique names
  Π.χ.

```
<owl:ObjectProperty rdf:ID="isTaughtBy">
   <rdf:type rdf:resource="&owl;FunctionalProperty"/>
</owl:ObjectProperty>
```

(Each course is taught by at most one member of staff)

```
<course rdf:ID="CS4553">
   <isTaughtBy rdf:resource="#949318"/>
   <isTaughtBy rdf:resource="#949352"/>
</course>
```

(Course CS4553 is taught by 949318 and 949352)

Owl does not create a reasoning error. 949318 and 949352 are considered non-different.

OWL- instances

- To ensure diversity we must declare it:

```
<lecturer rdf:ID="949318">
   <owl:differentFrom rdf:resource="#949352"/>
</lecturer>
```

or in a group mode:

```
<owl:AllDifferent>
   <owl:distinctMembers rdf:parseType="Collection"/>
      <lecturer rdf:about = "#949318"/>
      <lecturer rdf:about = "#949352"/>
      <lecturer rdf:about = "#949311"/>
   </owl:distinctMembers>
</owl:AllDifferent>
```

OWL Sub-languages

- OWL Full
  - It uses all the fundamental elements ("constructors") of OWL and allows combining them in any arbitrary way, via RDF and RDFS.
  - Capability to change meaning of fundamental elements of RDF and OWL.
  - Fully compatible with RDF.
    - Every valid RDF document is also a valid OWL Full document.
    - Any valid inference in RDF is also a valid inference in OWL full.
  - Reasoning efficiency problems.

OWL Sub-languages

- OWL DL ($\mathcal{SHOIN}^{(\mathcal{D})}$)
  - Rectrictions to OWL DL ontology
    - Vocabulary partitioning. Each resource is just: class, data type, data type property, object property, individual element, data value, part of built-in vocabulary. E.g. a class cannot be an individual element at the same time, or a property cannot be both a type property and an object property.
    - Ρητή τυποποίηση (explicit typing). Partitioning must be explicitly declared. E.g. Even if the following is declared

      <owl:Class rdf:ID="C1"> <rdf:subClassOf rdf:about="#C2"/> </owl:Class>

      the following must be explicitly declared too:

      <owl:Class rdf:ID="C2"/>
    - Separation of properties. The sets "object properties" and "type properties" are foreign to each other. So the following cannot be defined as data type properties: owl:inverseOf, owl:FunctionalProperty, owl:InverseFunctionalProperty, owl:SymmetricProperty

OWL Sub-languages

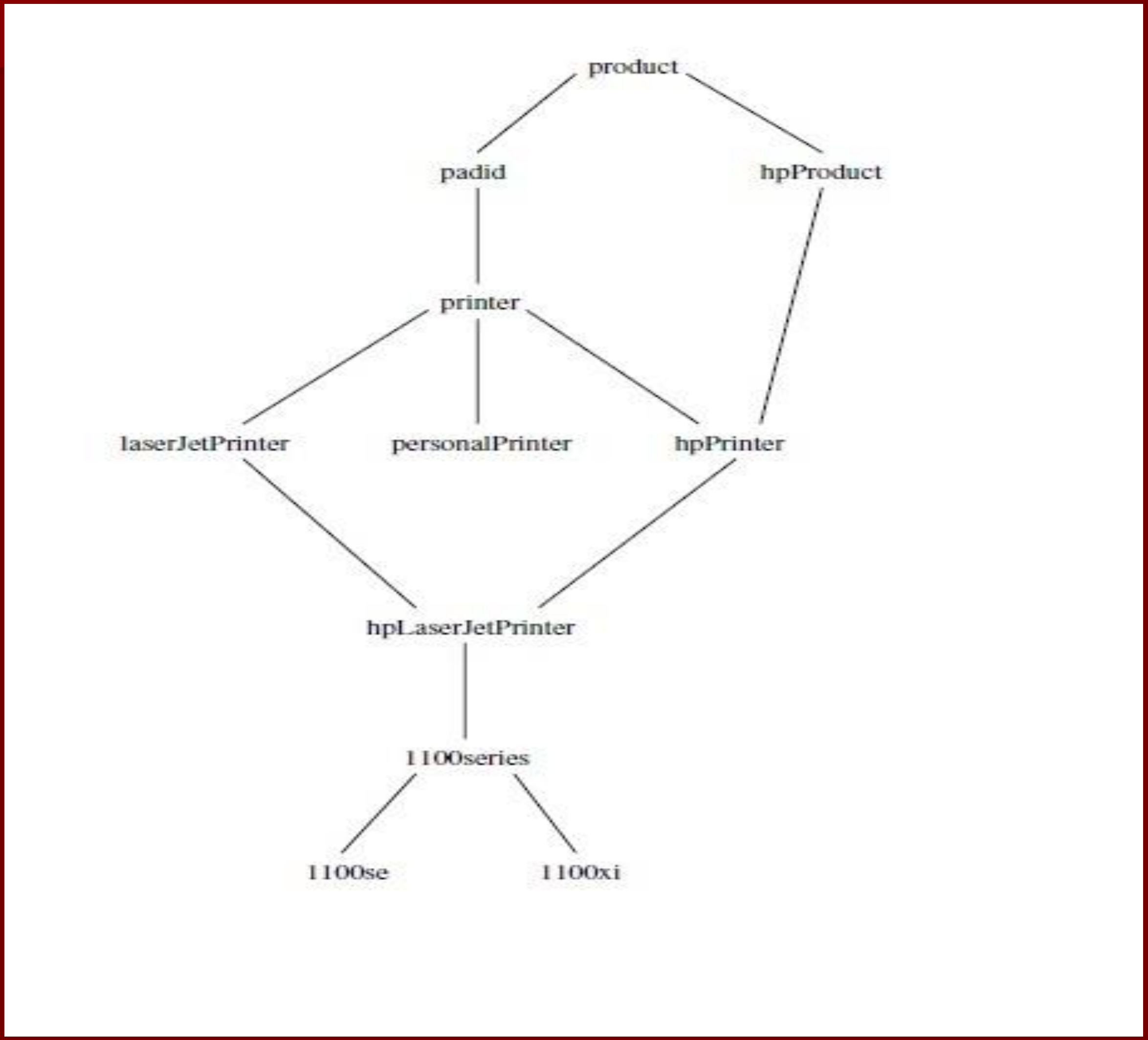
- Restrictions to OWL DL ontology (cont.)
  - Absence of transitive cardinality constraints. Cardinality constraints cannot be applied to transitive properties (or their superproperties that are also transitive)
  - Restricted anonymous classes. Anonymous classes may only appear as the domain and values set of either the owl:equivalentClass or owl:disjointWith element, as well as the values set (but not the domain) of the rdfs:subClassOf property.
- Full compatibility with RDF is lost.
  - An RDF document should be extended in some ways and restricted in others to be considered an OWL DL document.
  - Every valid OWL DL document is a valid RDF document.

OWL Sub-languages

- OWL Lite ($\mathcal{SHIF}^{(\mathcal{D})}$)
  - Rectrictions to OWL Lite ontology (on top of those for OWL DL)
    - The elements ("constructors") owl:oneOf, owl:disjointWith, owl:unionOf, owl:complementOf and owl:hasValue are not allowed.
    - Cardinality statements (min, max, and exact numbers) can only be made for the values 0 or 1, not for arbitrary non-negative integers.
    - Statements owl:equivalentClass can no longer be made between anonymous classes, but only between class identifiers.
    - Every valid OWL Lite ontology is a valid OWL DL ontology.
    - Any valid inference in OWL Lite is a valid inference in OWL DL

Printers Ontology

From "A Semantic Web Primer", by G. Antoniou and F.V. Harmelen



Printers Ontology

```
<!DOCTYPE owl [
    <!ENTITY xsd  "http://www.w3.org/2001/XMLSchema#" >
]>

<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:owl ="http://www.w3.org/2002/07/owl#"
  xmlns="http://www.cs.vu.nl/~frankh/spool/printer.owl#">

<owl:Ontology rdf:about="">
 <owl:versionInfo>
  My example version 1.2, 17 October 2002
 </owl:versionInfo>
</owl:Ontology>

<owl:Class rdf:ID="product">
 <rdfs:comment>Products form a class.</rdfs:comment>
</owl:Class>
```

Printers Ontology

```
<owl:Class rdf:ID="padid">
 <rdfs:comment>
  Printing and digital imaging devices
  form a subclass of products.
 </rdfs:comment>
 <rdfs:label>Device</rdfs:label>
 <rdfs:subClassOf rdf:resource="#product"/>
</owl:Class>

<owl:Class rdf:ID="hpProduct">
 <rdfs:comment>
  HP products are exactly those products
  that are manufactured by Hewlett Packard.
 </rdfs:comment>
 <owl:intersectionOf rdf:parseType="Collection">
  <owl:Class rdf:about="#product"/>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#manufactured_by"/>
```

Printers Ontology

```
    <owl:hasValue rdf:datatype="&xsd;string">
      Hewlett Packard
    </owl:hasValue>
   </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>

<owl:Class rdf:ID="printer">
 <rdfs:comment>
  Printers are printing and digital imaging devices.
 </rdfs:comment>
 <rdfs:subClassOf rdf:resource="#padid"/>
</owl:Class>

<owl:Class rdf:ID="personalPrinter">
 <rdfs:comment>
  Printers for personal use form a subclass of printers.
 </rdfs:comment>
 <rdfs:subClassOf rdf:resource="#printer"/>
</owl:Class>
```

Printers Ontology

```
<owl:Class rdf:ID="hpPrinter">
 <rdfs:comment>
  HP printers are  HP products and printers.
 </rdfs:comment>
 <rdfs:subClassOf rdf:resource="#printer"/>
 <rdfs:subClassOf rdf:resource="#hpProduct"/>
</owl:Class>

<owl:Class rdf:ID="laserJetPrinter">
 <rdfs:comment>
  Laser jet printers are exactly those
  printers that use laser jet printing technology.
 </rdfs:comment>
 <owl:intersectionOf rdf:parseType="Collection">
  <owl:Class rdf:about="#printer"/>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#printingTechnology"/>
   <owl:hasValue rdf:datatype="&xsd;string">
    laser jet
   </owl:hasValue>
  </owl:Restriction>
```

Printers Ontology

```
<owl:Class rdf:ID="hpLaserJetPrinter">
 <rdfs:comment>
  HP laser jet printers are HP products
  and laser jet printers.
 </rdfs:comment>
 <rdfs:subClassOf rdf:resource="#laserJetPrinter"/>
 <rdfs:subClassOf rdf:resource="#hpPrinter"/>
</owl:Class>

<owl:Class rdf:ID="1100series">
 <rdfs:comment>
  1100series printers are HP laser jet printers with
  8ppm printing speed and 600dpi printing resolution.
 </rdfs:comment>
 <rdfs:subClassOf rdf:resource="#hpLaserJetPrinter"/>
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#printingSpeed"/>
   <owl:hasValue rdf:datatype="&xsd;string">
    8ppm
   </owl:hasValue>
  </owl:Restriction>
```

Printers Ontology

```
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:onProperty rdf:resource="#printingResolution"/>
     <owl:hasValue rdf:datatype="&xsd;string">
      600dpi
     </owl:hasValue>
    </owl:Restriction>
   </rdfs:subClassOf>
  </owl:Class>

  <owl:Class rdf:ID="1100se">
   <rdfs:comment>
     1100se printers belong to the 1100 series and cost $450.
   </rdfs:comment>
   <rdfs:subClassOf rdf:resource="#1100series"/>
   <rdfs:subClassOf>
```

Printers Ontology

```
  <owl:Restriction>
   <owl:onProperty rdf:resource="#price"/>
   <owl:hasValue rdf:datatype="&xsd;integer">
    450
   </owl:hasValue>
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:ID="1100xi">
 <rdfs:comment>
   1100xi printers belong to the 1100 series and cost $350.
 </rdfs:comment>
 <rdfs:subClassOf rdf:resource="#1100series"/>
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#price"/>
   <owl:hasValue rdf:datatype="&xsd;integer">
    350
   </owl:hasValue>
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
```

Printers Ontology

```
<owl:DatatypeProperty rdf:ID="manufactured_by">
 <rdfs:domain rdf:resource="#product"/>
 <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:ID="price">
 <rdfs:domain rdf:resource="#product"/>
 <rdfs:range rdf:resource="&xsd;nonNegativeInteger"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:ID="printingTechnology">
 <rdfs:domain rdf:resource="#printer"/>
 <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:ID="printingResolution">
 <rdfs:domain rdf:resource="#printer"/>
 <rdfs:range rdf:resource="&xsd;string"/>
```

Printers Ontology

```
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:ID="printingSpeed">
 <rdfs:domain rdf:resource="#printer"/>
 <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

</rdf:RDF>
```