SENG 42273
Semantic web and Ontological Engineering

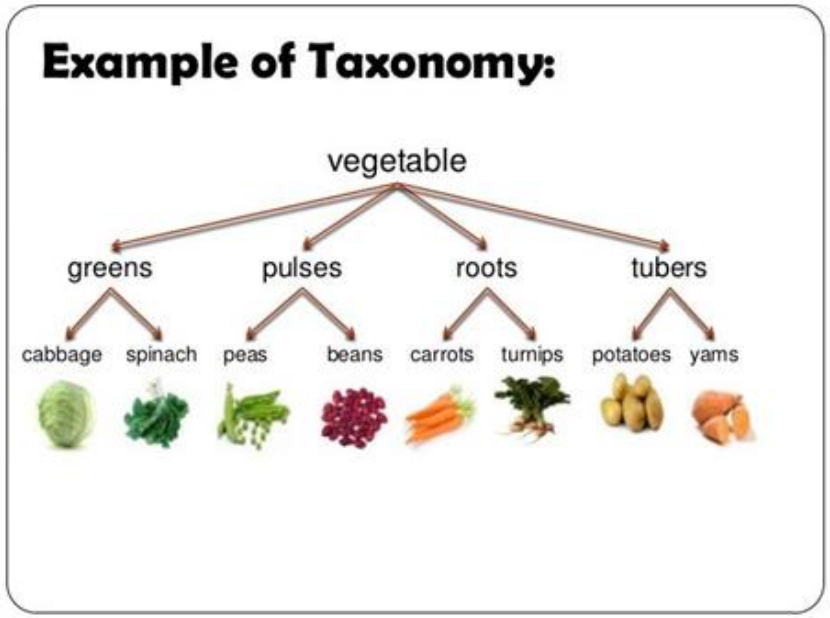
Lecture 6

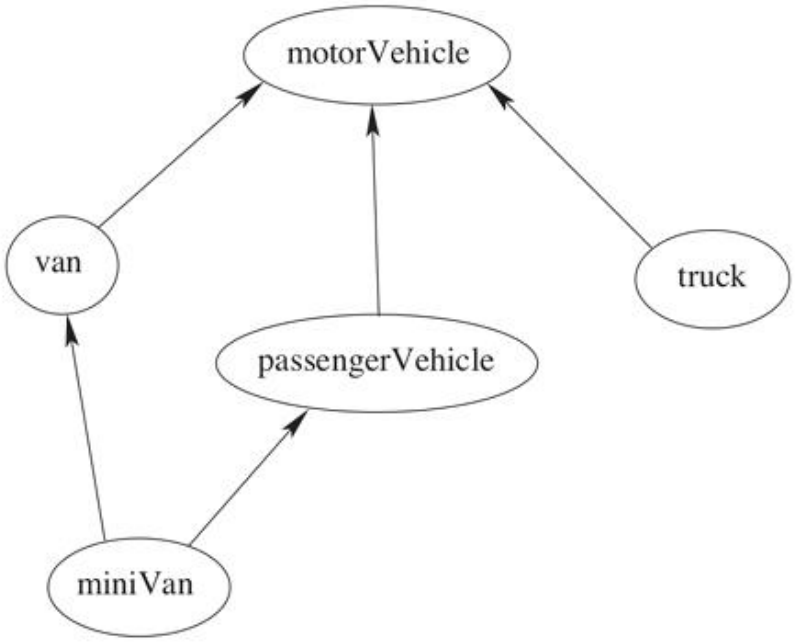
What is an Ontology?

- We can describe the things/concepts using RDF and RDFS.
  - Define the same concept in different documents is a redundancy
  - Cannot reuse the definition of a concept defined in a document
  - Is there a way to define those concepts in a reusable manner and using a common standard?
- OWL: Web Ontology Language
  - that allow us to express Ontologies (concepts) described using RDF and RDFS
  - defined concepts in reusable manner

What is an Ontology?

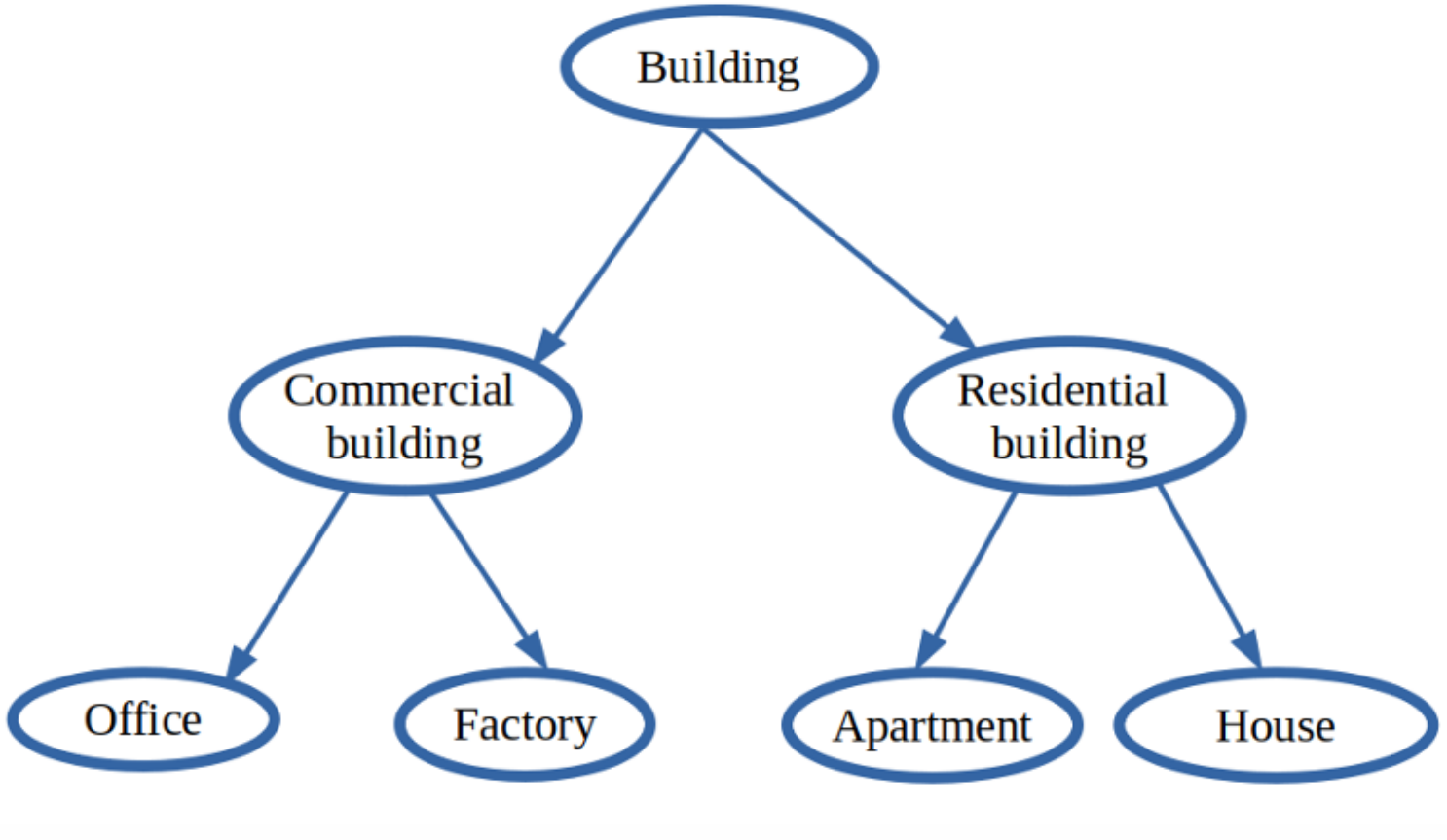
- **Taxonomy**
  - Classification of things(not the properties)
- **Ontology**
  - An ontology defines the terms used to describe and represent an area(domain) of knowledge (W3C's OWL Requirements Documents). Ontology defines not only the classes but also their properties. It further indicates the type of values these properties may have and what classes they may be associated with, thereby creating sophisticated relationships among the classes.
  - Explicit formal specification of the concepts in a domain is called an ontology.



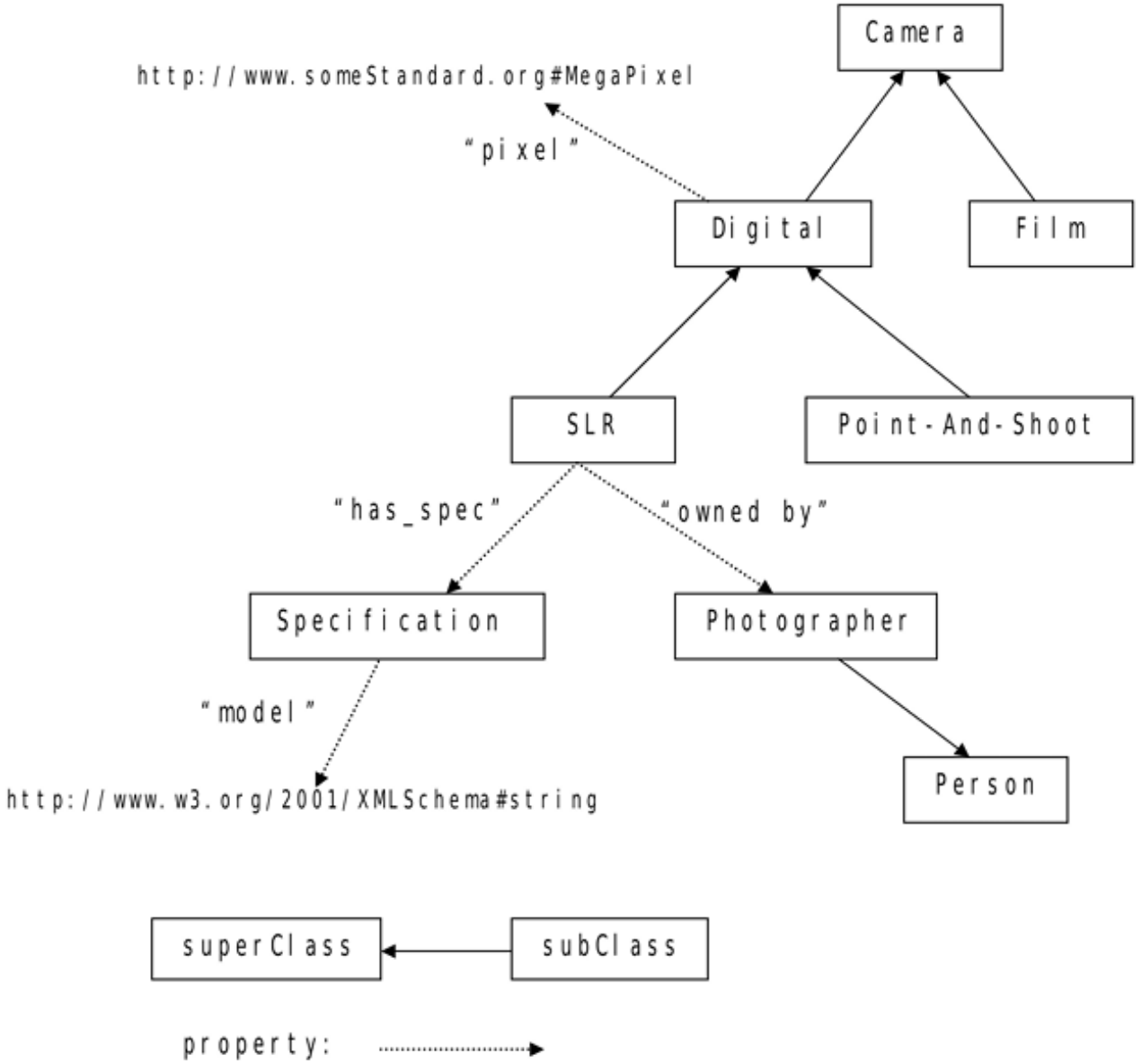


- **Ontology**
  - is domain specific
  - defines a group of terms in the given domain and the relationships among them

Example


Building
Commercial building
Residential building
Office
Factory
Apartment
House

Example


Camera
http://www.someStandard.org#MegaPixel
"pixel"
Digital
Film
SLR
Point-And-Shoot
"has_spec"
"owned by"
Specification
Photographer
"model"
http://www.w3.org/2001/XMLSchema#string
Person
superClass
subClass
property:

Why we need Ontologies?

- It provides a common and shared understanding/definition about certain key concepts in the domain.
- It provides a way to reuse domain knowledge.
- It makes the domain assumptions explicit.
- Together with ontology description languages (such as RDF schema), it provides a way to encode knowledge and semantics such that machines can understand.
- It makes automatic large-scale machine processing possible.

Web Ontology Language(OWL)

Most popular language for creating Ontologies

OWL = RDFS(all classes and properties) + new constructs for expressiveness

- A well-defined syntax, a formal semantics, sufficient expressive power, convenience of expression, and efficient reasoning support
  - Ex: A person can have one and only one birth date
- Built upon description logic(DL) and RDFS.
  - Reasoning power/ inference power

Web Ontology Language(OWL)

- The three sublanguages of OWL and their most important general properties.

OWL Full

- contains OWL DL and OWL Lite,
- is the only OWL sublanguage containing all of RDFS,
- very expressive,
- semantically difficult to understand and to work with,
- undecidable,
- supported by hardly any software tools.

OWL DL

- contains OWL Lite and is contained in OWL Full,
- decidable,
- fully supported by most software tools,
- worst-case computational complexity: NExpTime.

OWL Lite

- contained in OWL DL and OWL Full,
- decidable,
- less expressive,
- worst-case computational complexity: ExpTime.

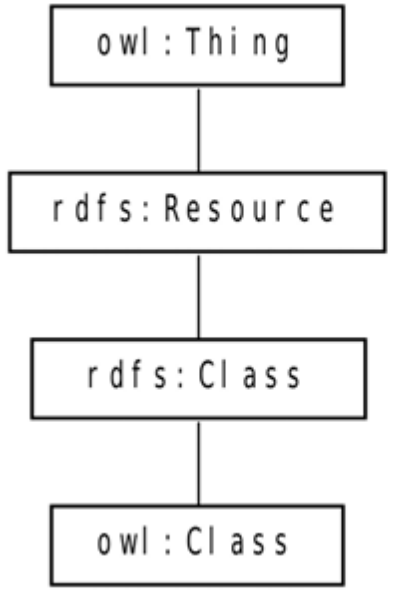
Web Ontology Language(OWL)

RDFS root class definition

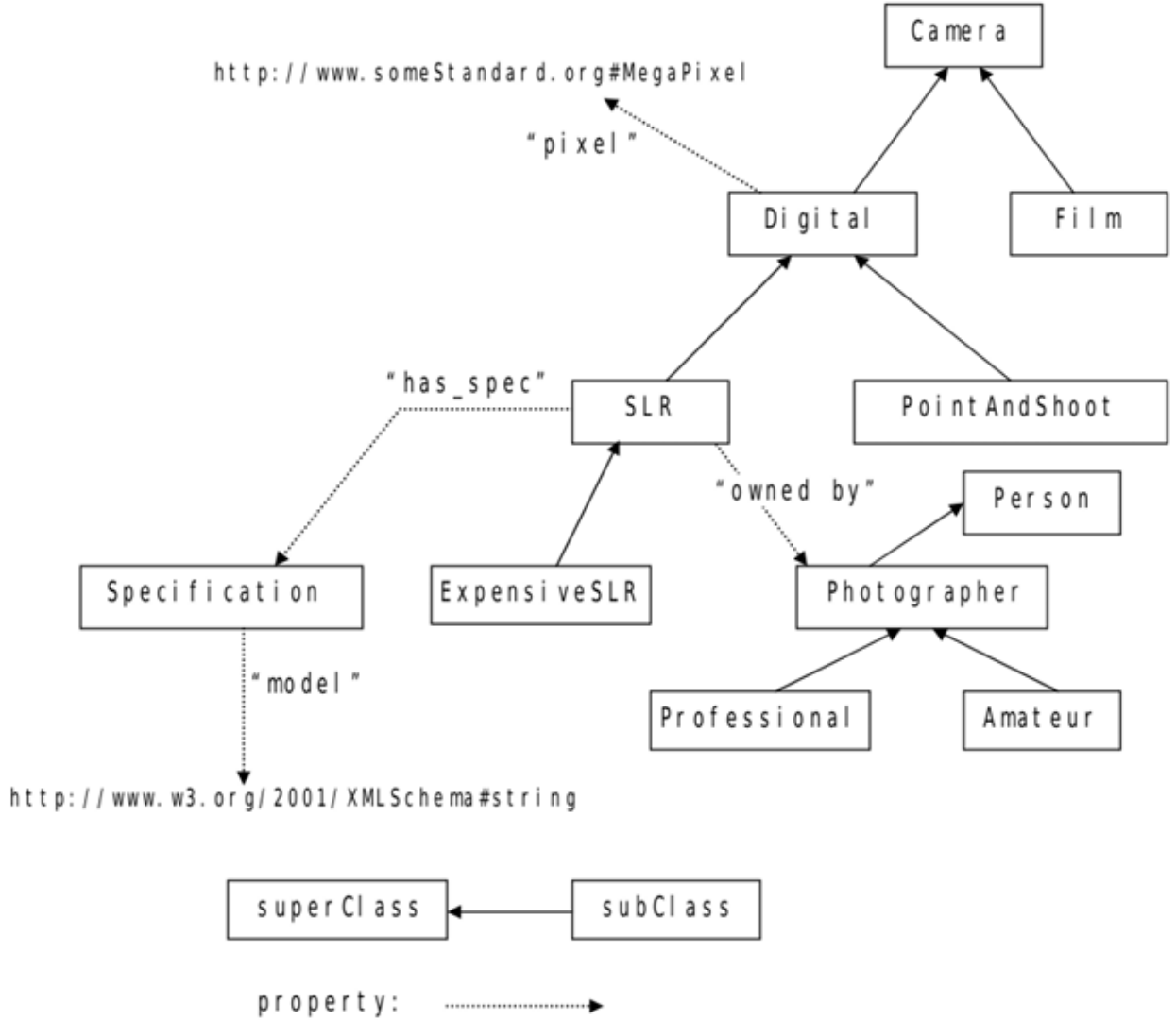
`rdfs:resource`
`http://www.w3.org/2001/01/rdf-schema#resource`

OWL root class definition

`owl:Thing`
`http://www.w3.org/2002/07/owl#Thing`



Example


Camera
http://www.someStandard.org#MegaPixel
"pixel"
Digital
Film
"has_spec"
SLR
PointAndShoot
"owned by"
Person
Specification
ExpensiveSLR
Photographer
"model"
Professional
Amateur
http://www.w3.org/2001/XMLSchema#string
superClass
subClass
property:

OWL namespace

```
http://www.w3.org/2002/07/owl#
```

Define camera ontology top classes,Camera

<owl:Class rdf:ID="Camera">
</owl:Class>
This tells that "Camera" is an OWL class

Define camera ontology top classes,Camera

<owl:Class rdf:ID="Camera">
<rdfs:subClassOf
rdfs:resource="http://www.w3.org/2002/07/owl#Thing"/>
</owl:Class>
Tells that "Camera is an OWL class and it is a subclass of owl:thing"

Example

```
//
// classes definitions
//
6:    <owl:Class rdf:ID="Camera">
7:    </owl:Class>

8:    <owl:Class rdf:ID="Person">
9:    </owl:Class>

10:   <owl:Class rdf:ID="Digital">
11:      <rdfs:subClassOf rdf:resource="#Camera"/>
12:   </owl:Class>

13:   <owl:Class rdf:ID="Film">
14:      <rdfs:subClassOf rdf:resource="#Camera"/>
15:   </owl:Class>

16:   <owl:Class rdf:ID="SLR">
17:      <rdfs:subClassOf rdf:resource="#Digital"/>
18:   </owl:Class>

19:   <owl:Class rdf:ID="PointAndShoot">
20:      <rdfs:subClassOf rdf:resource="#Digital"/>
21:   </owl:Class>

22:   <owl:Class rdf:ID="Photographer">
23:      <rdfs:subClassOf rdf:resource="#Person"/>
24:   </owl:Class>

25:   <owl:Class rdf:ID="Specifications">
26:   </owl:Class>

//
// property definitions: coming up ...
//
```

Enhanced Reasoning Power of OWL

Inheritance

- Subclasses inherit properties if its super class(es)
  - "ExpensiveSLR" subclass inherits "owned by" property from class SLR. So, `ExpensiveSLR owned by Amateur` is valid

New Vocabularies

- `owl:allValuesFrom`
- `owl:someValuesFrom`
- `owl:hasValue`
- `owl:cardinality`
- `owl:minCardinality`
- `owl:maxCardinality`

Enhanced Reasoning Power of OWL

Restrict inheritance using `owl:allValuesFrom`

How do we modify the definition of ExpensiveSLR to ensure that it can be owned only by Professional ?

LIST 5.2
`owl:allValuesFrom` Example

```
1: <owl:Class rdf:ID="ExpensiveSLR">
2:    <rdfs:subClassOf rdf:resource="#SLR"/>
3:      <rdfs:subClassOf>
4:        <owl:Restriction>
5:          <owl:onProperty rdf:resource="#owned_by"/>
6:          <owl:allValuesFrom rdf:resource="#Professional"/>
7:        </owl:Restriction>
8:    </rdfs:subClassOf>
9: </owl:Class>
```

We can interpret this definition as follows:

Here is a definition of class `ExpensiveSLR`; it is a subclass of `SLR` and has a property named `owned_by`, and only an instance of class `Professional` can be the value of this property.

Enhanced Reasoning Power of OWL

OWL inferencing power 1

```
<ExpensiveSLR rdf:ID="Nikon D200">
    <owned_by
rdf:resource="http://www.yuchen.net/people#Liyang"/>
    <owned_by
rdf:resource="http://www.yuchen.net/people#Jin"/>
    </ExpensiveSLR>
```

- The agent(semantic search engine) understands that Both Liyang and Jin are Professionals (not Photographers or Amateurs).

Enhanced Reasoning Power of OWL

How to express "at least one value of owned_by property is an instance of Professional"

LIST 5.3
owl:someValuesFrom Example

```
1: <owl:Class rdf:ID="ExpensiveSLR">
2:   <rdfs:subClassOf rdf:resource="#SLR"/>
3:     <rdfs:subClassOf>
4:       <owl:Restriction>
5:         <owl:onProperty rdf:resource="#owned_by"/>
6:         <owl:someValuesFrom rdf:resource="#Professional"/>
7:       </owl:Restriction>
8:     </rdfs:subClassOf>
9: </owl:Class>
```

This can be interpreted as follows:

A class called `ExpensiveSLR` is defined. It is a subclass of `SLR`, and it has a property called `owned_by`. Furthermore, at least one value of `owned_by` property is an instance of `Professional`.

Enhanced Reasoning Power of OWL

Define a specific "value" for a property

LIST 5.4
owl:hasValue Example

```
1: <owl:Class rdf:ID="ExpensiveSLR">
2:    <rdfs:subClassOf rdf:resource="#SLR"/>
3:    <rdfs:subClassOf>
4:       <owl:Restriction>
5:          <owl:onProperty rdf:resource="#expensiveOrNot"/>
6:          <owl:hasValue rdf:datatype="http://www.w3.org/2001/
                XMLSchema#string">expensive
            </owl:hasValue>
7:       </owl:Restriction>
8:    </rdfs:subClassOf>
9: </owl:Class>
```

Enhanced Reasoning Power of OWL

Cardinality constraints

LIST 5.7
Using `owl:maxCardinality` and `owl:minCardinality` to Specify a Range

```
1:  <owl:Class rdf:ID="ExpensiveSLR">
2:     <rdfs:subClassOf rdf:resource="#SLR"/>
3:     <rdfs:subClassOf>
4:        <owl:Restriction>
5:           <owl:onProperty rdf:resource="#owned_by"/>
6:           <owl:minCardinality
       rdf:datatype="http://www.w3.org/2001/XMLSchema
           #nonNegativeInteger">
7:           1
8:           </owl:minCardinality>
9:           <owl:maxCardinality
       rdf:datatype="http://www.w3.org/2001/XMLSchema
           #nonNegativeInteger">
10:          2
11:          </owl:maxCardinality>
12:       </owl:Restriction>
10:    </rdfs:subClassOf>
11: </owl:Class>
```

Enhanced Reasoning Power of OWL

`owl:property` characteristics

Property can be

- Symmetric
- Transitive
- Functional
- Inverse functional
- The inverse of another property
- ...

Enhanced Reasoning Power of OWL

Symmetric

If resource `R1` is connected to resource `R2` by property `P` , then resource `R2` is also connected to resource `R1` by the same property. If person `A` is friend_with person `B` , then person `B` is certainly friend_with person `A`

Example of Symmetric Property

```
1: <owl:ObjectProperty rdf:ID="friend_with">
2:    <rdf:type
          rdf:resource="http://www.w3.org/2002/07/owl
              #SymmetricProperty"/>
3:    <rdfs:domain rdf:resource="#Person"/>
4:    <rdfs:range rdf:resource="#Person"/>
5: </owl:ObjectProperty>
```

Enhanced Reasoning Power of OWL

Transitive

If a resource R1 is connected to resource R2 by property P , and resource R2 is connected to resource R3 by the same property, then resource R1 is also connected to resource R3 by property P

Example of Transitive Property

```
1: <owl:ObjectProperty rdf:ID="betterQualityPriceRatio">
2:    <rdf:type
          rdf:resource="http://www.w3.org/2002/07/owl
              #TransitiveProperty"/>
3:    <rdfs:domain rdf:resource="#Camera"/>
4:    <rdfs:range rdf:resource="#Camera"/>
5: </owl:ObjectProperty>
```

Example

Validation Results

Your RDF document validated successfully.

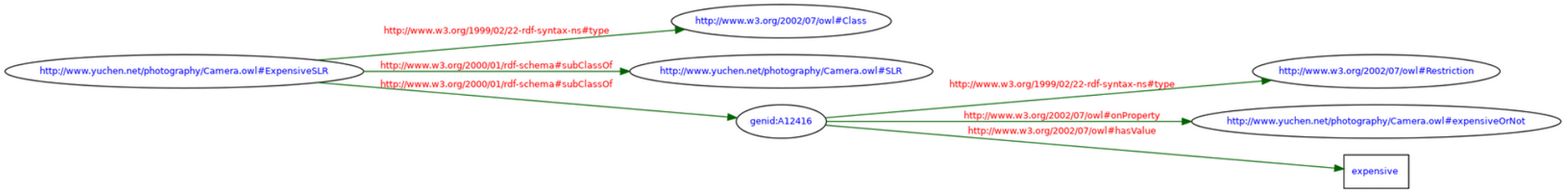
Triples of the Data Model

Number	Subject	Predicate	Object
1	http://www.yuchen.net/photography/Camera.owl#ExpensiveSLR	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#Class
2	http://www.yuchen.net/photography/Camera.owl#ExpensiveSLR	http://www.w3.org/2000/01/rdf-schema#subClassOf	http://www.yuchen.net/photography/Camera.owl#SLR
3	genid:A12416	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#Restriction
4	http://www.yuchen.net/photography/Camera.owl#ExpensiveSLR	http://www.w3.org/2000/01/rdf-schema#subClassOf	genid:A12416
5	genid:A12416	http://www.w3.org/2002/07/owl#onProperty	http://www.yuchen.net/photography/Camera.owl#expensiveOrNot
6	genid:A12416	http://www.w3.org/2002/07/owl#hasValue	"expensive"^^http://www.w3.org/2001/ XMLSchema#string

The original RDF/XML document

```
1: <?xml version="1.0"?>
2: <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3: xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
4: xmlns:owl="http://www.w3.org/2002/07/owl#"
5: xml:base="http://www.yuchen.net/photography/Camera.owl">
6: <owl:Class rdf:ID="ExpensiveSLR">
7: <rdfs:subClassOf rdf:resource="#SLR"/>
8: <rdfs:subClassOf>
9: <owl:Restriction>
10: <owl:onProperty rdf:resource="#expensiveOrNot"/>
11: <owl:hasValue rdf:datatype="http://www.w3.org/2001/
12: XMLSchema#string">expensive
13: </owl:hasValue>
14: </owl:Restriction>
15: </rdfs:subClassOf>
16: </owl:Class>
17: </rdf:RDF>
```

Graph of the data model



End