Advanced ontologies and reasoning

María Poveda Villalón, Ontology Engineering Group
Universidad Politécnica de Madrid, Spain

[mpoveda@fi.upm.es] @MariaPovedaV SSoLDAC24

Licence

- This work is licensed under Creative Commons Attribution – Non Commercial – Share Alike License
- *You are free:*
  - *to Share — to copy, distribute and transmit the work*
  - *to Remix — to adapt the work*

- Under the following conditions
  - *Attribution — You must attribute the work by inserting*
    - "[source http://www.oeg-upm.net/]" at every reused slide
    - A slide declaring: "This material is partially based on "Ontology Development" by María Poveda Villalón"
  - *Non-commercial*
  - *Share-Alike*

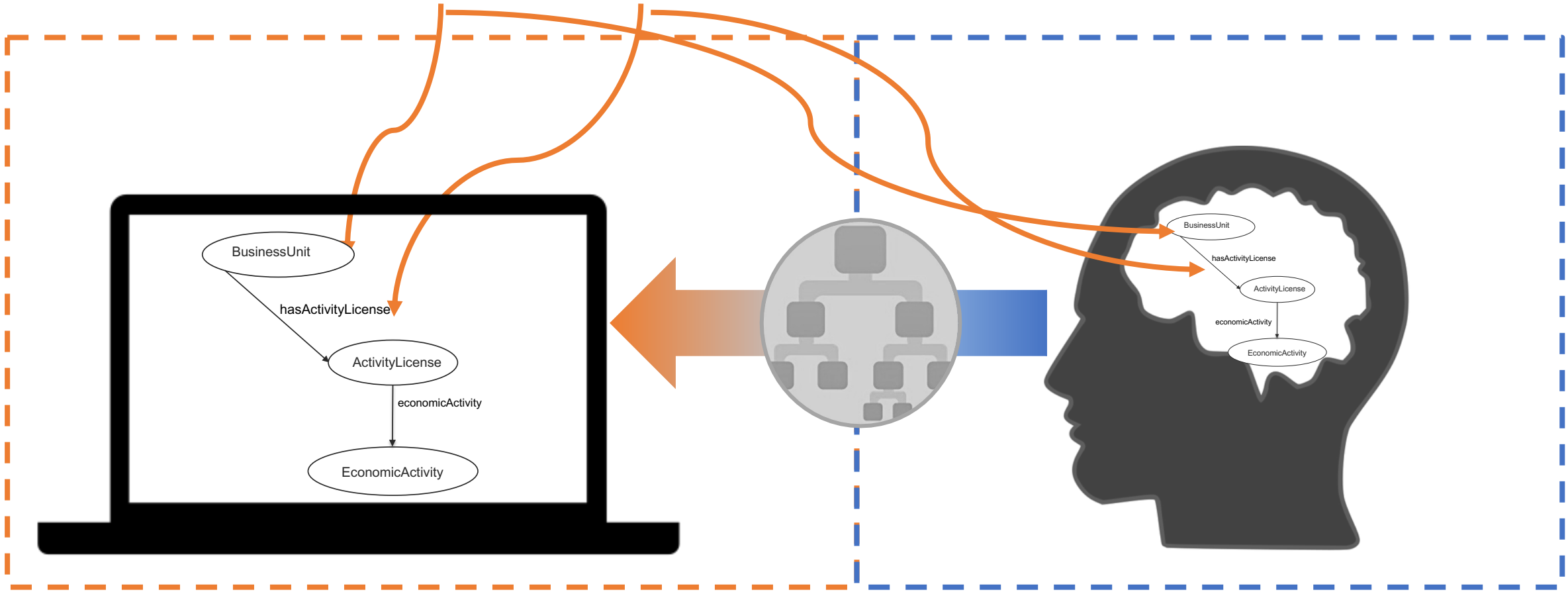
About this material

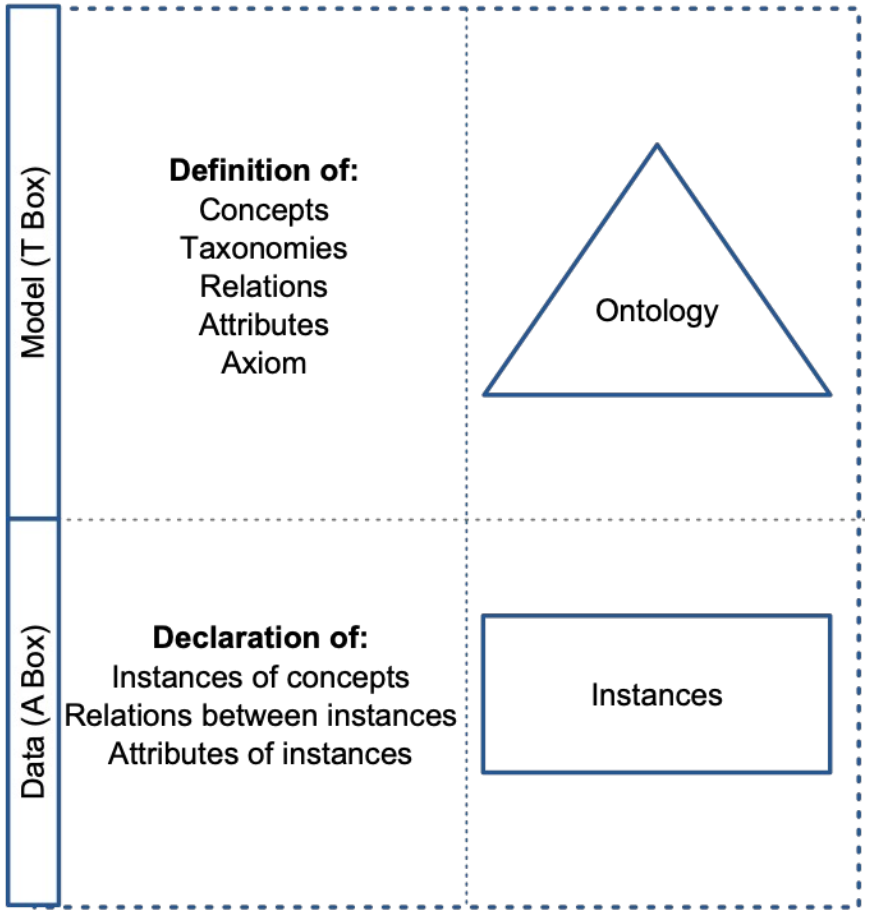
- The materials for this session has been elaborated by María Poveda Villalón reusing content generated by the following OEG colleagues:
  - Oscar corcho
  - Raúl García-Castro
  - Alba Fernández-Izquierdo
  - Etc.

Ask anytime

(https://www.w3.org/standards/semanticweb/ontology)

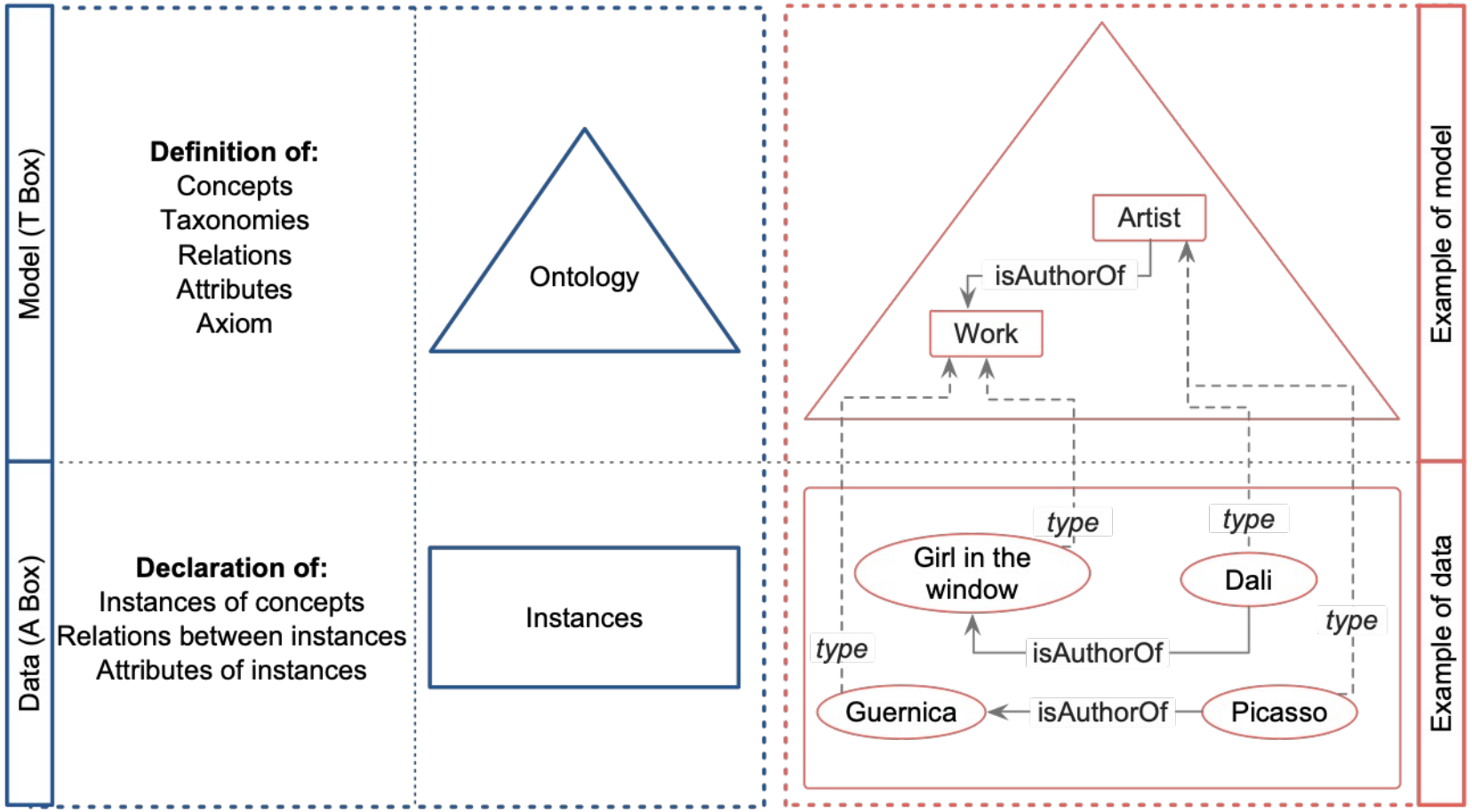
A vocabulary defines the **concepts** and **relations** used to describe and represent a **domain** of interest



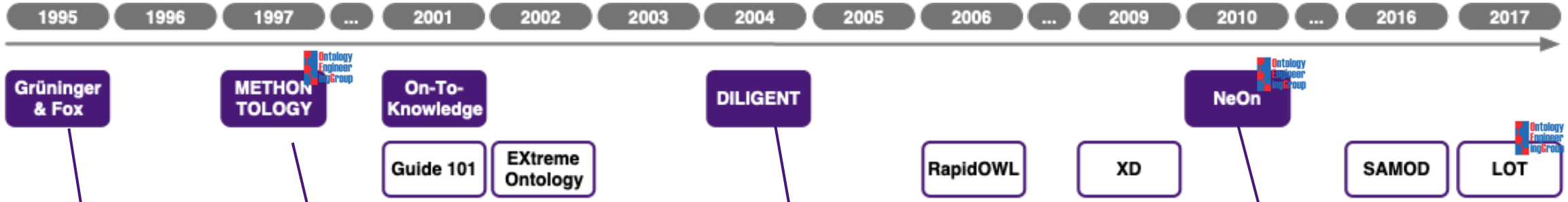


- A knowledge base is divided into:
  - **Tbox** (*Terminological KB*): set of axioms that define the domain:
    - Artist⊏ Person
    - Person ≡ Man ⊔ Woman
  - **Abox** (*Assertional KB*): set of axioms that descibe a situation:
    - Picasso: Man
    - isAuthorOf (Picasso, Guernica)

Structure of *DL* ontologies



Some Ontology Development Methodologies



- Competency Questions technique

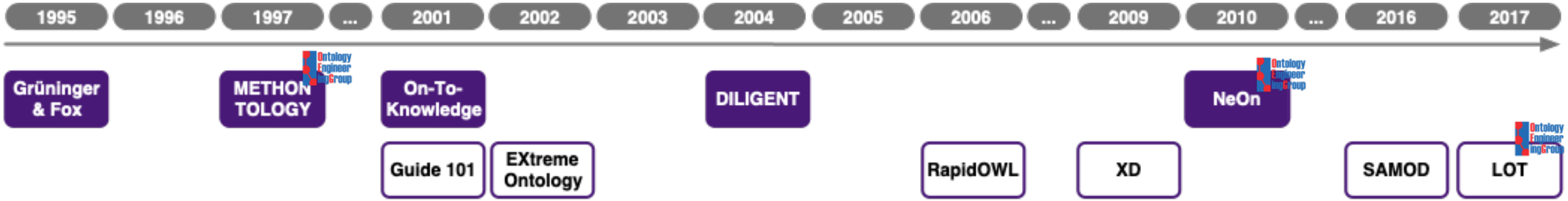
- Intermediate representations
- Activities identification and definition
- Activities organization

- Best practices
- Lightweight
- Common problems addressed
- Frames oriented… but still applicable

- Collaborative process
- Local adaptations

- 9 Scenarios
- Activity templates
- Guidelines for: specification, ontology reuse, NOR reuse, ODP reuse, scheduling, etc.

Ontology Development Methodologies

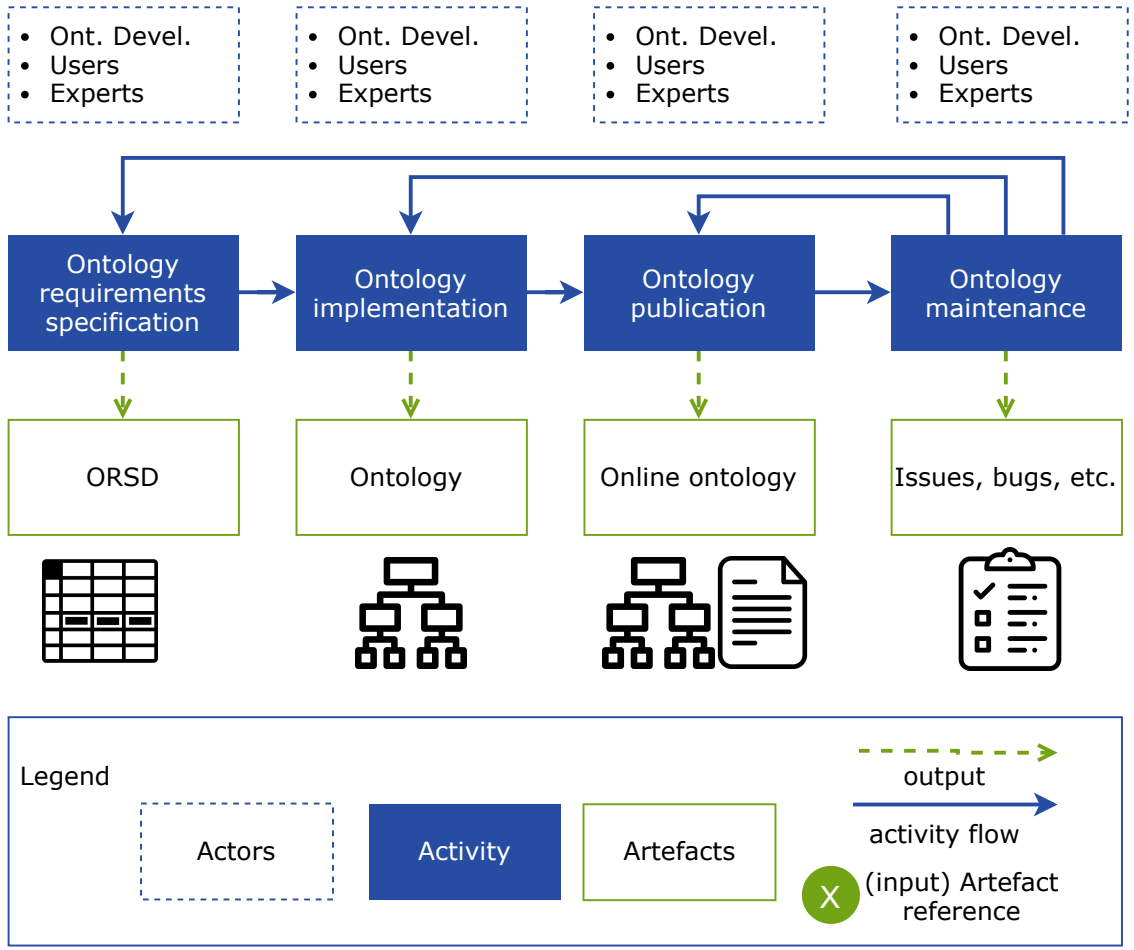


- Towards lightweight and agile processes
- Inspiration from software development practices
- Coupling Software and ontology development

O. Development Methodologies | **Ontology Development Lightweight Approaches**

Slide taken from “Trendy Practices and Tools in Ontological Engineering” by Dr. María Poveda Villalón

Ontology development process overview

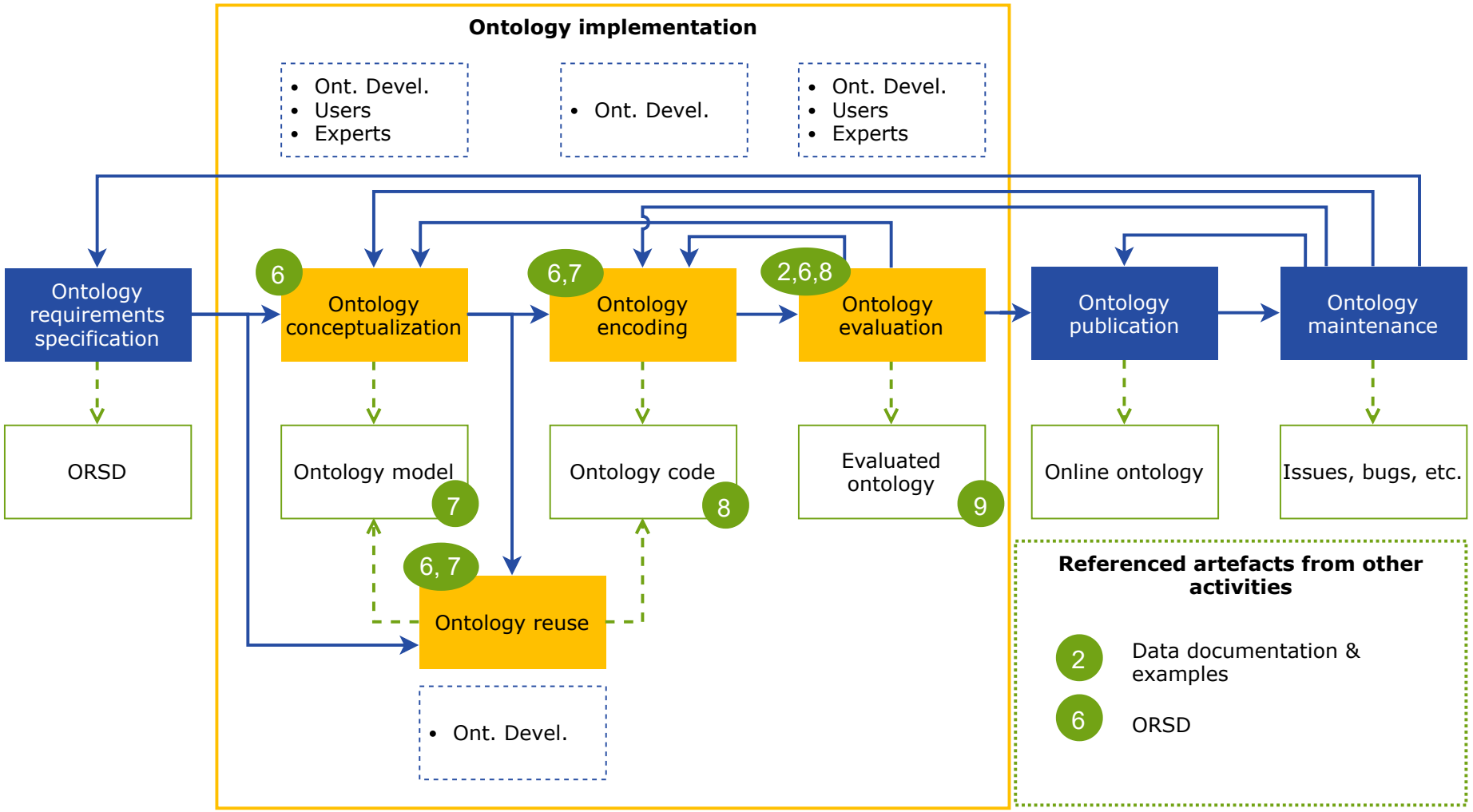


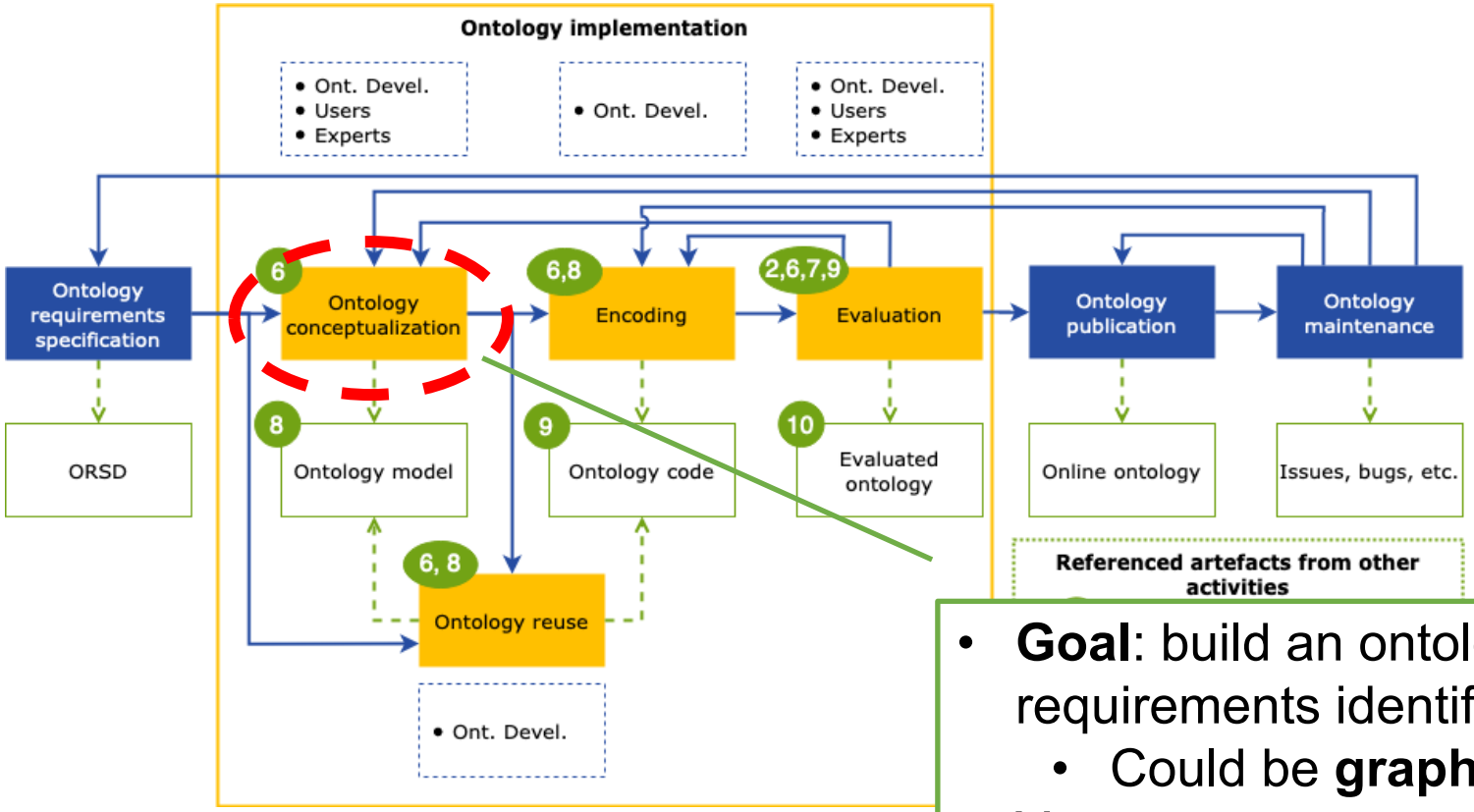
AURORAL

BIMERR RENOVATION 4.0

COGITO

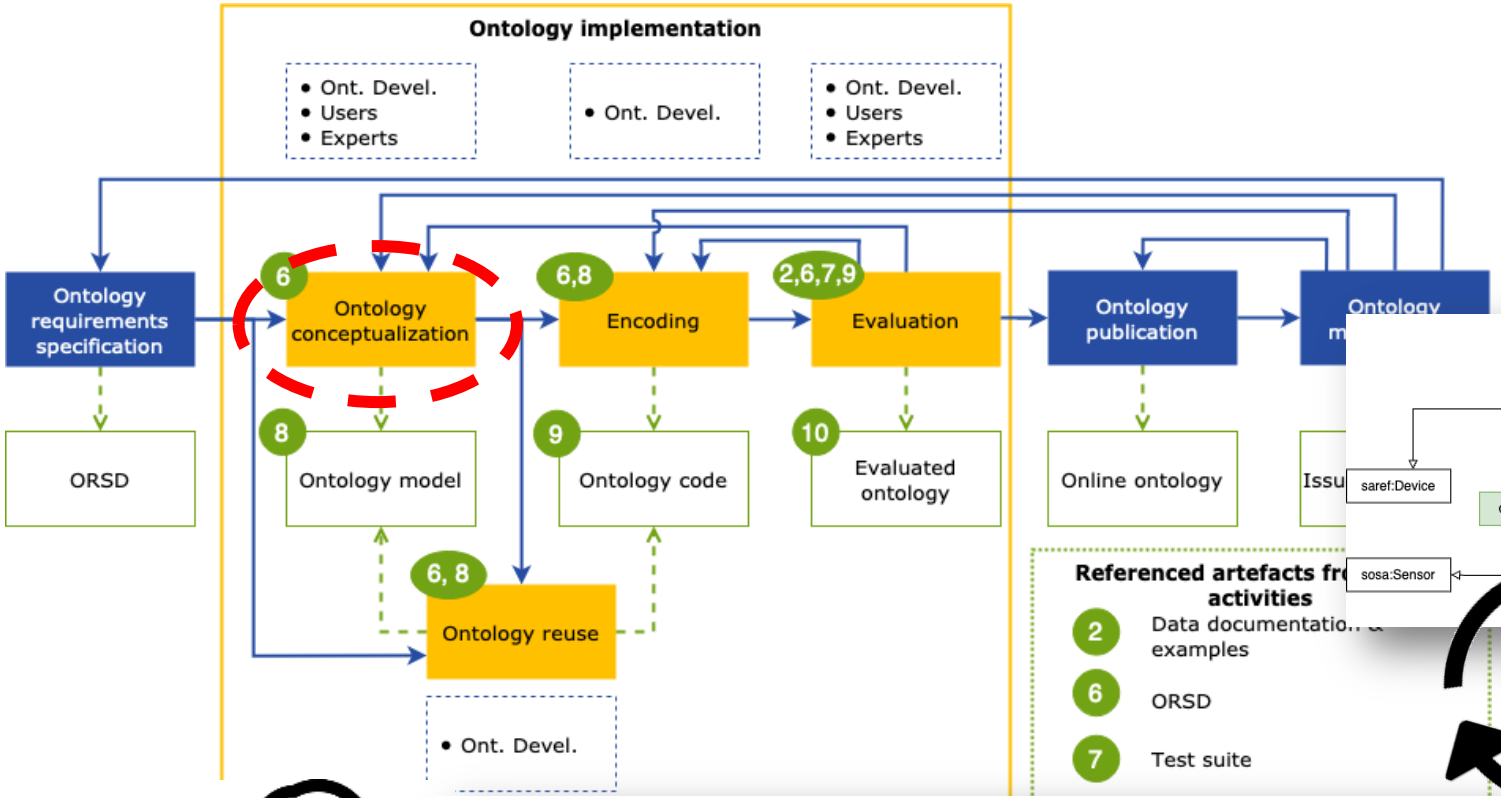
http://lot.linkeddata.es/ https://doi.org/10.1016/j.engappai.2022.104755

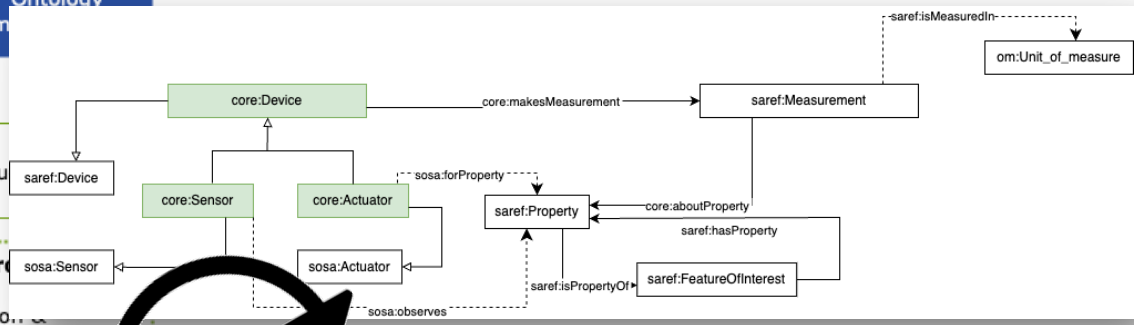

Ontology implementation
• Ont. Devel.
• Users
• Experts
• Ont. Devel.
• Ont. Devel.
• Users
• Experts
Ontology requirements specification
6
Ontology conceptualization
6,7
Ontology encoding
2,6,8
Ontology evaluation
Ontology publication
Ontology maintenance
ORSD
Ontology model
7
Ontology code
8
Evaluated ontology
9
Online ontology
Issues, bugs, etc.
6, 7
Ontology reuse
• Ont. Devel.
Referenced artefacts from other activities
2 Data documentation & examples
6 ORSD

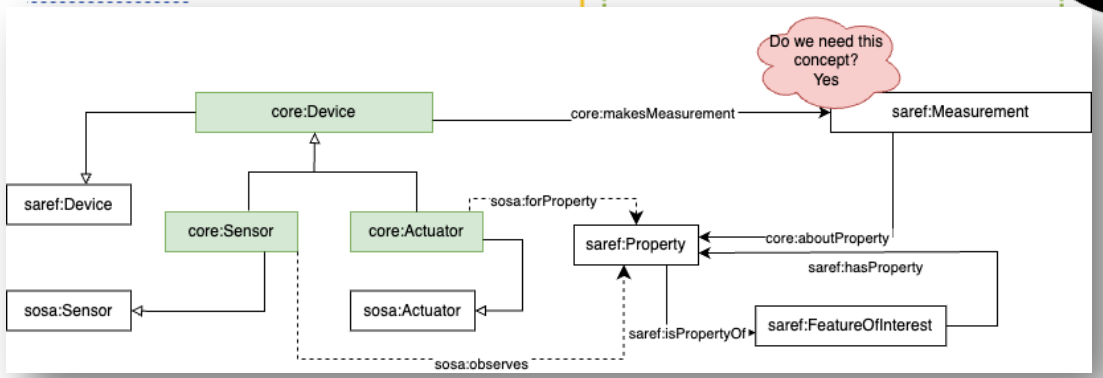


- **Goal**: build an ontology model from the ontological requirements identified.
  - Could be **graphical** or described in a formal system
- You can use
  - Blackboard
  - Pen & pencil
  - Drawing tools (diagrams.net, visio, yEd…)

Suárez-Figueroa, Mari Carmen, Asunción Gómez-Pérez, and Mariano Fernández-López. "The NeOn methodology for ontology engineering." *Ontology engineering in a networked world*. Springer, Berlin, Heidelberg, 2012. 9-34.



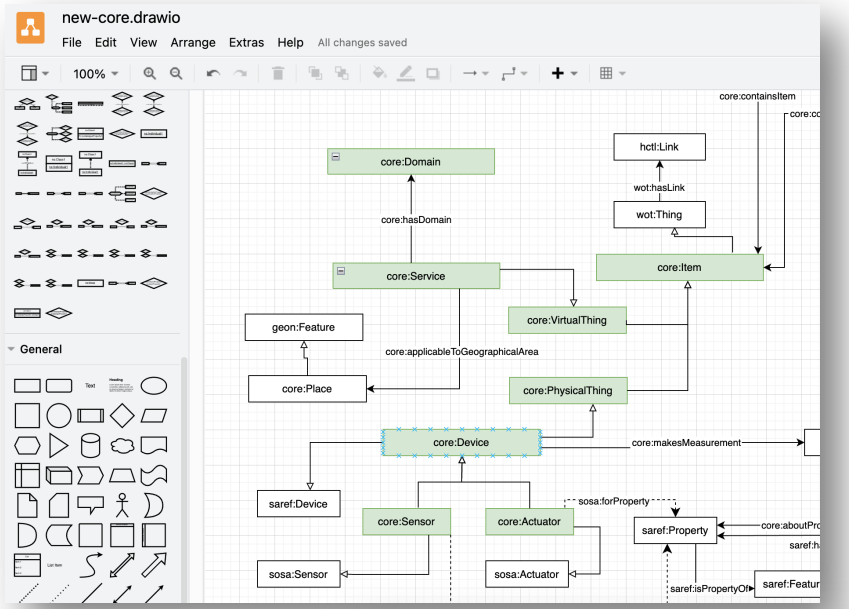




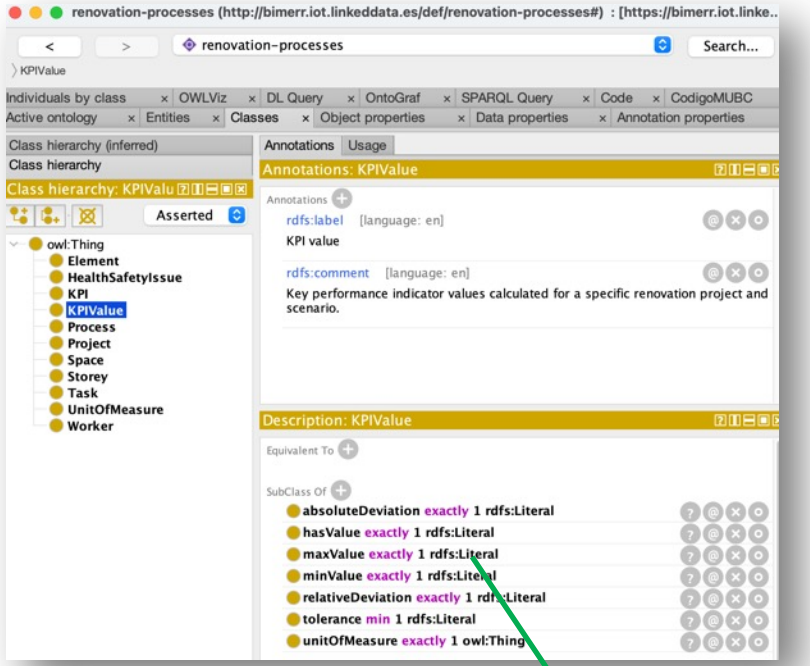
Domain Expert

Ontology Engineer

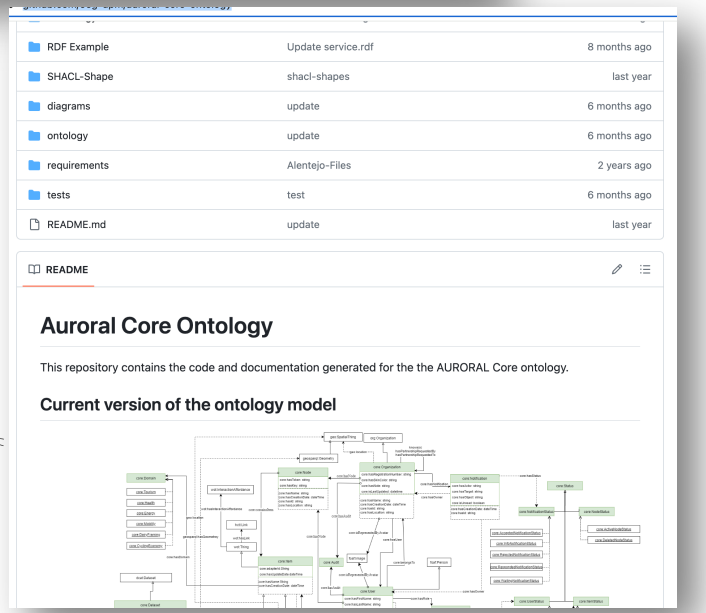
Implementation - Encoding



Chowlk

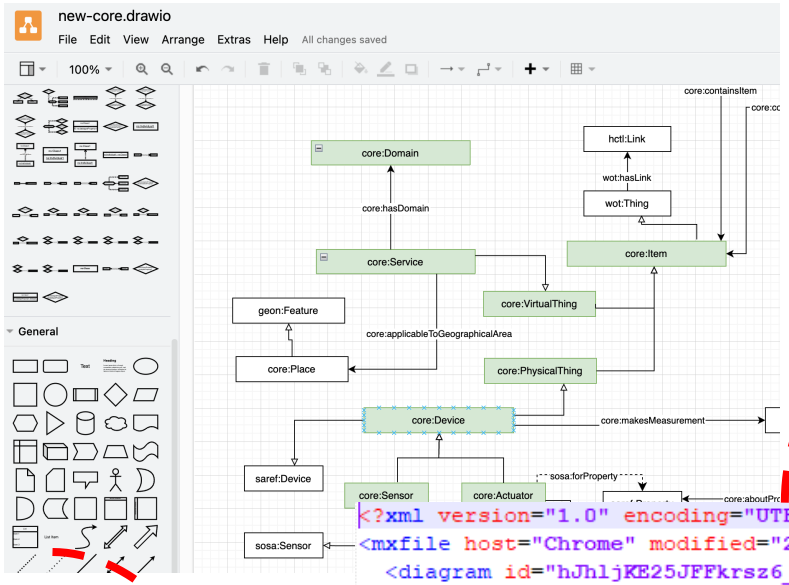


protégé

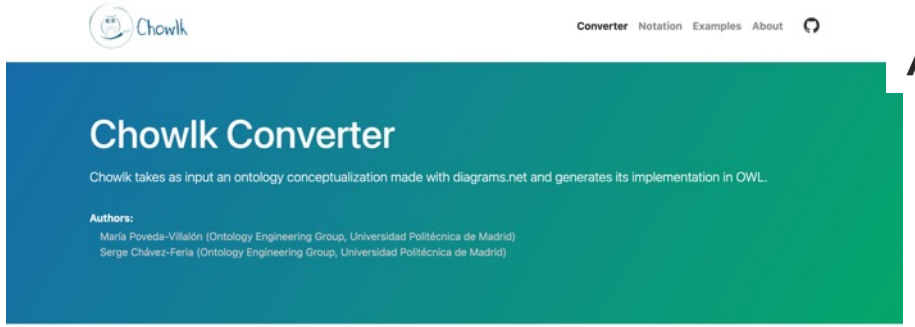


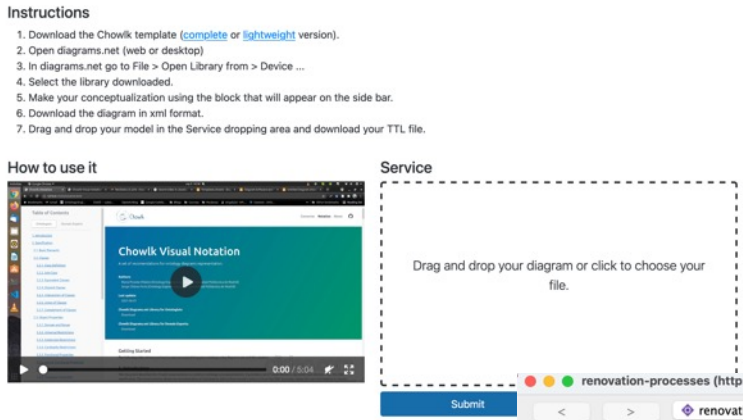
GitHub

```
@prefix : <http://bimerr.iot.linkeddata.es/def/renovation-processes#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@base <http://bimerr.iot.linkeddata.es/def/renovation-processes#> .

<http://bimerr.iot.linkeddata.es/def/renovation-processes#> rdf:type owl:Ontology ;
                                                             <http://purl.org/dc/elements/1.1/creator>
                                                             "María Poveda-Villalón" ,
                                                                    "Serge Chávez-Feria" ;
                                                             <http://purl.org/dc/elements/1.
                                                             1/description> "Ontology code created by
                                                             Chowlk" ;
                                                             <http://purl.org/dc/elements/1.1/license>
                                                             http://purl.org/NET/rdflicense/cc-by4.0" ;
                                                             <http://purl.org/dc/elements/1.1/publisher
                                                              "http://www.oeg-upm.net/" ;
                                                             <http://purl.org/dc/elements/1.1/title>
                                                             "Renovation Processes and Work Orders
                                                             Ontology" ;
                                                             <http://purl.
                                                             org/vocab/vann/preferredNamespacePrefix>
                                                             "renp" ;
                                                             rdfs:comment "Ontology to model building
                                                             renovation processes and work orders sent
                                                             to the workers on-site."@en ;
                                                             owl:versionInfo "0.0.3" .

#################################################################
#    Annotation properties
#################################################################

###  http://purl.org/dc/elements/1.1/creator
```



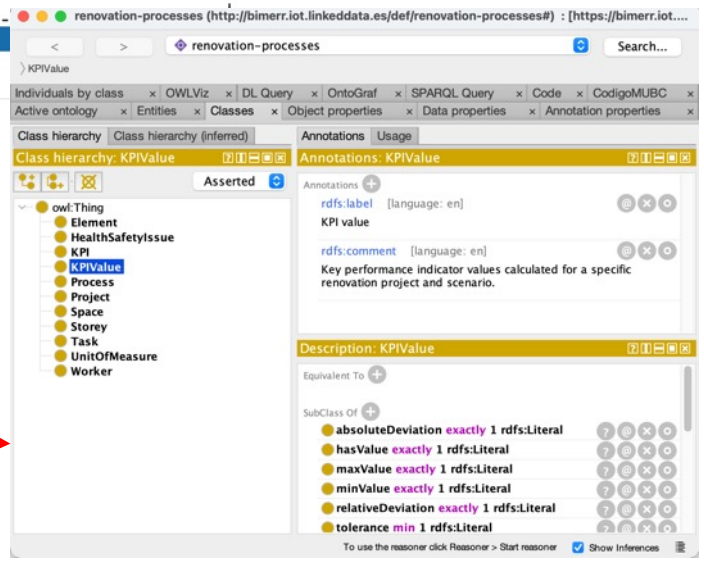
Convert into OWL with Chowlk





Export XML from diagrams.net

Continue in Protégé (if you want)



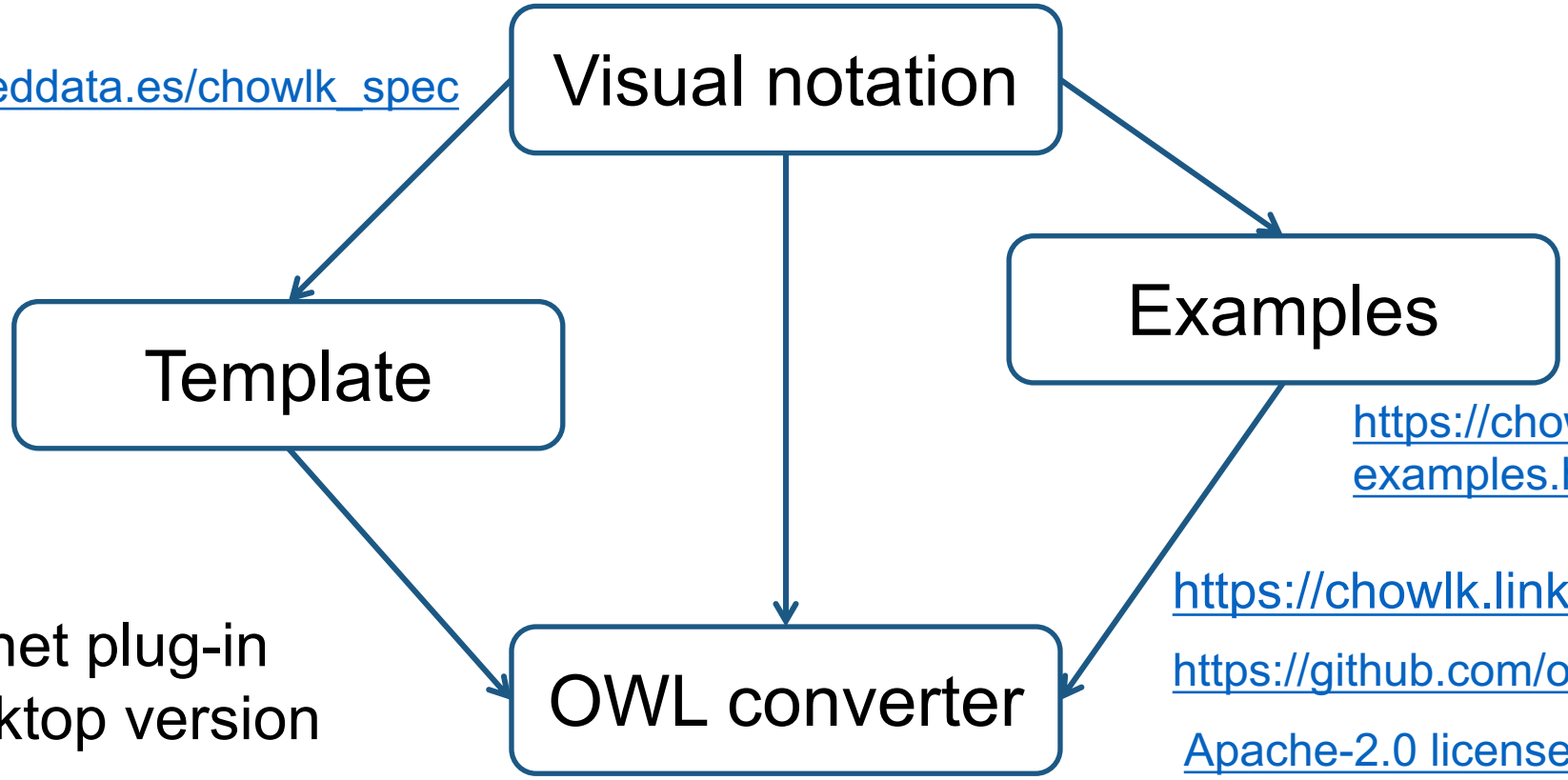
Chowlk

AURORAL

BIMERR

COGITO

https://chowlk.linkeddata.es/chowlk_spec



https://chowlk.linkeddata.es/examples.html

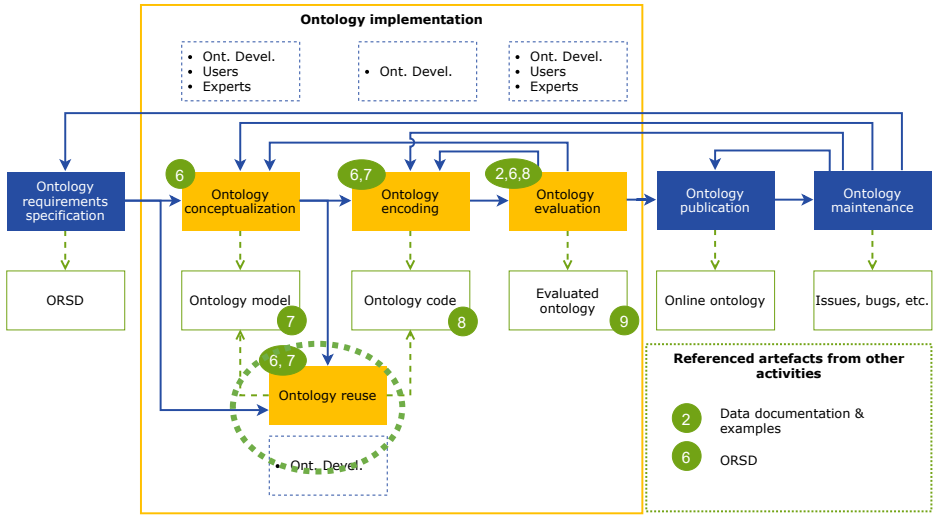
+ diagrams.net plug-in
Only for desktop version

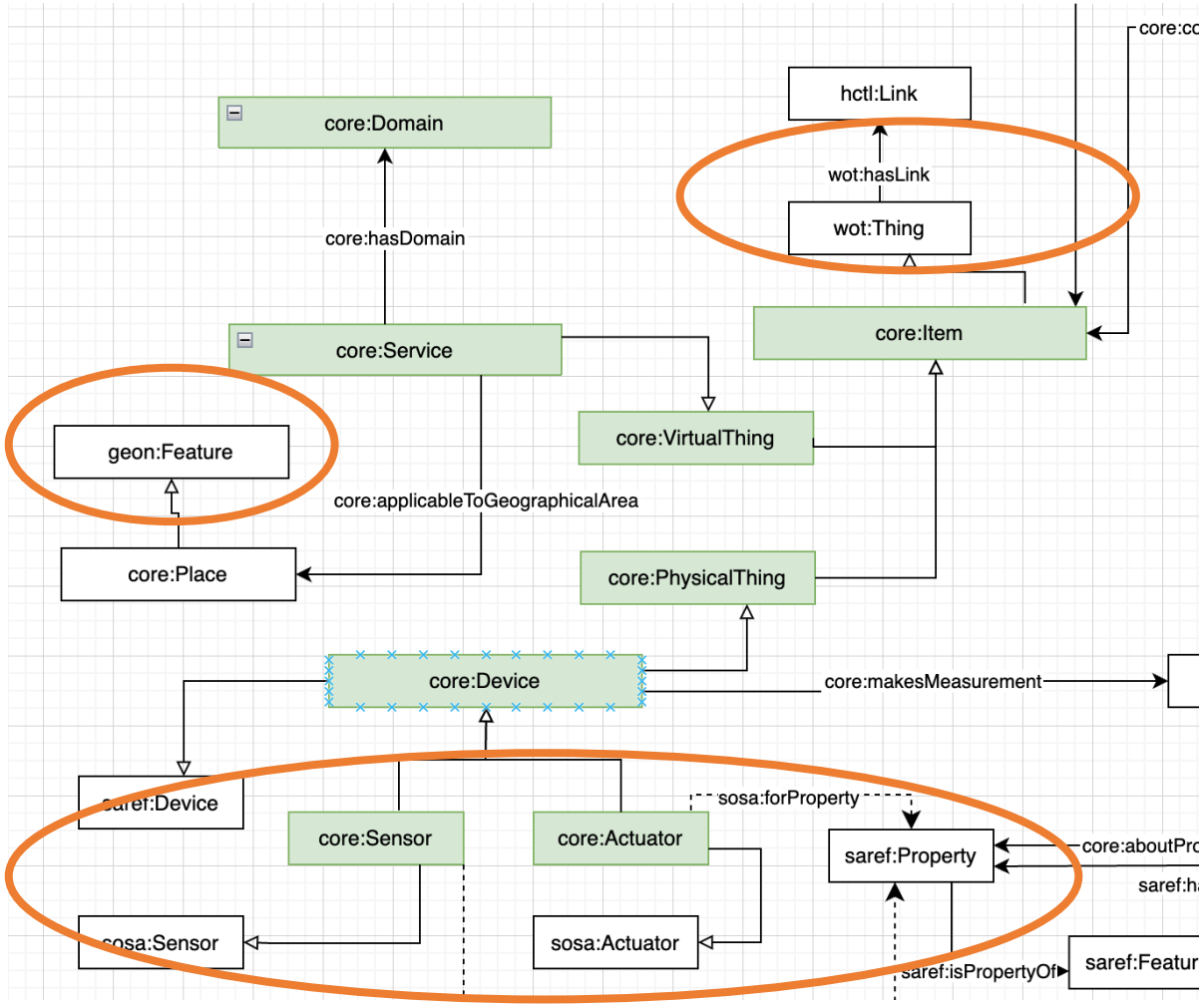
https://chowlk.linkeddata.es

https://github.com/oeg-upm/Chowlk

Apache-2.0 license

Implementation - Reuse

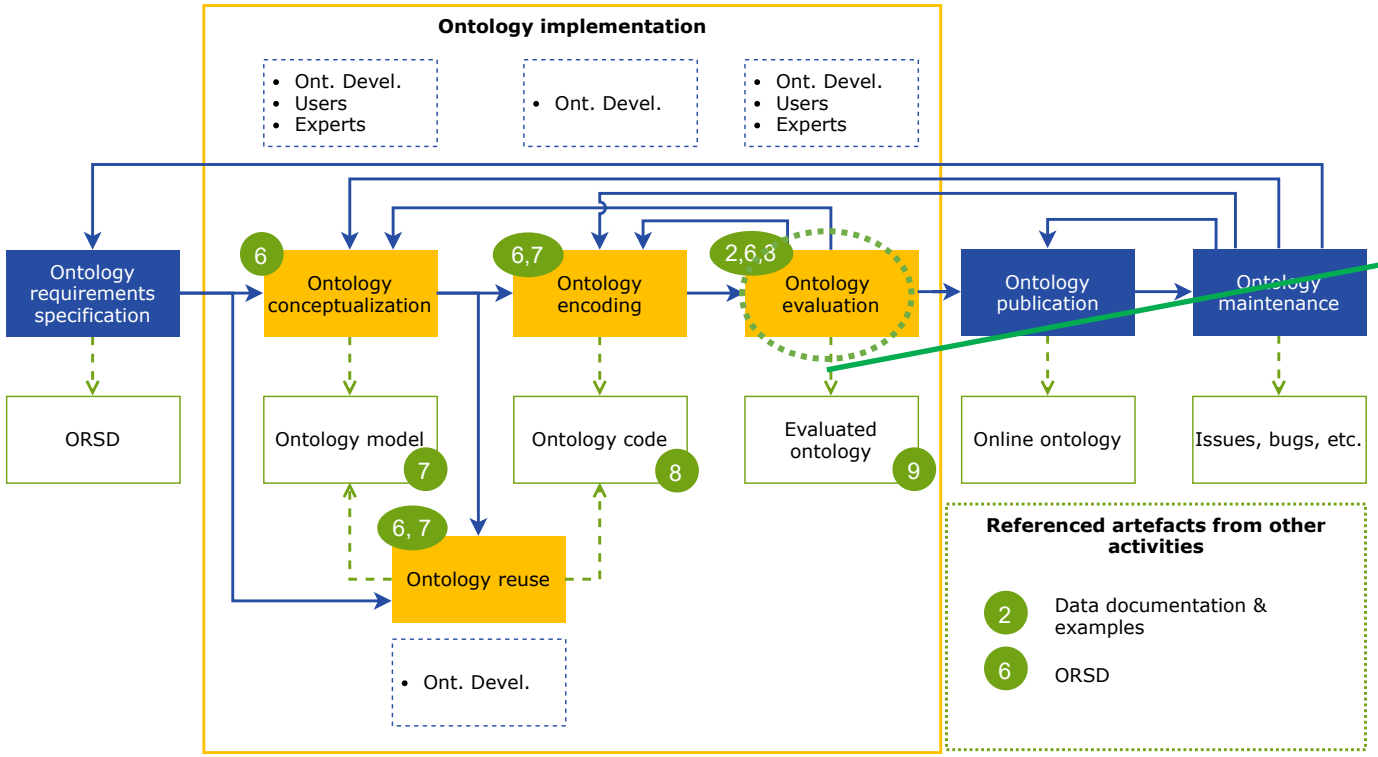


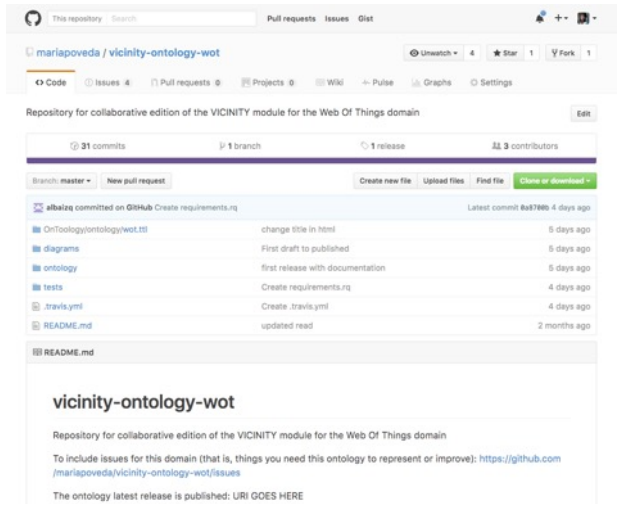


- **Reusing knowledge resources**

XML … Google refine

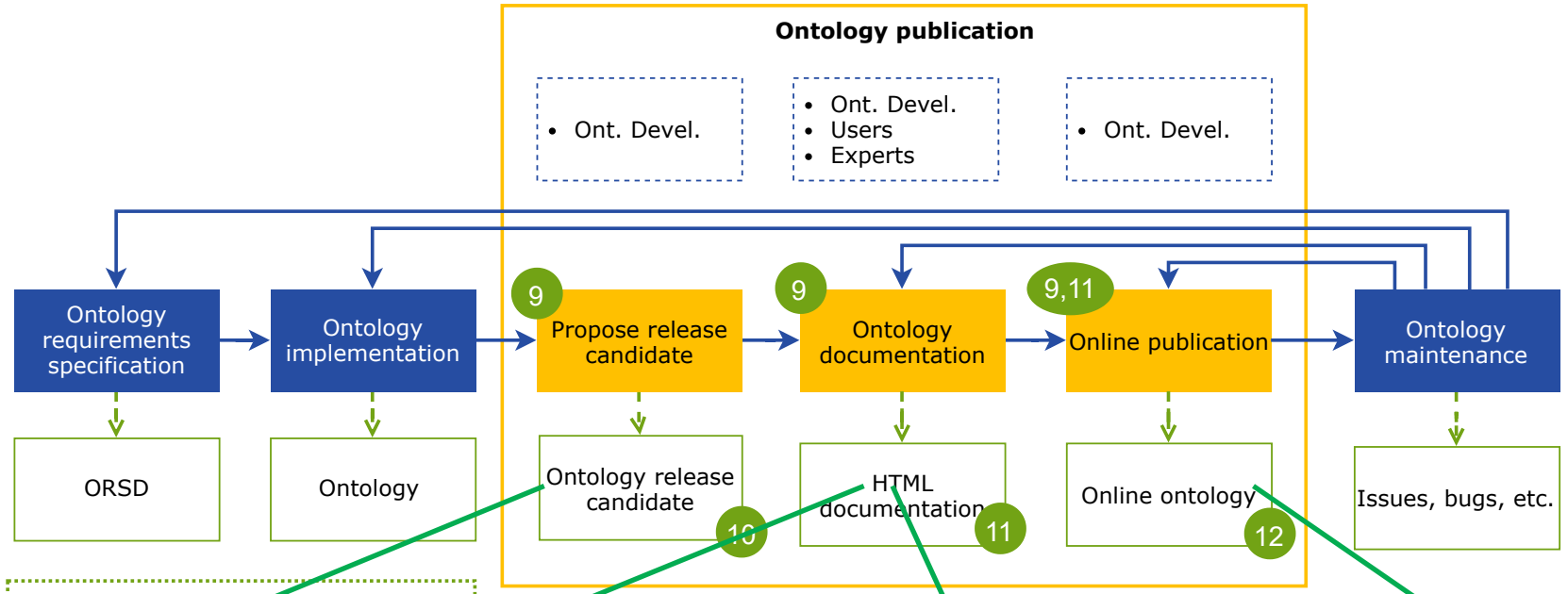
Look for existing ontologies:
https://lov.linkeddata.es
Etc.





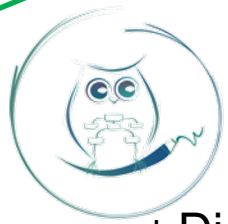
Online and **notifications** in **GitHub** repository
https://github.com/mariapoveda/vicinity-ontology-wot

- It refers to the activity of checking the technical quality of an ontology against a frame of reference. [NeOn]
  - Logical consistency checking
  - Domain coverage
  - Check common errors → **OOPS!** (http://oops.linkeddata.es/)
  - Check functional requirements → **Themis** (http://themis.linkeddata.es)



GitHub

- HTML generation from OWL code
- Multilingual
- Separated sections

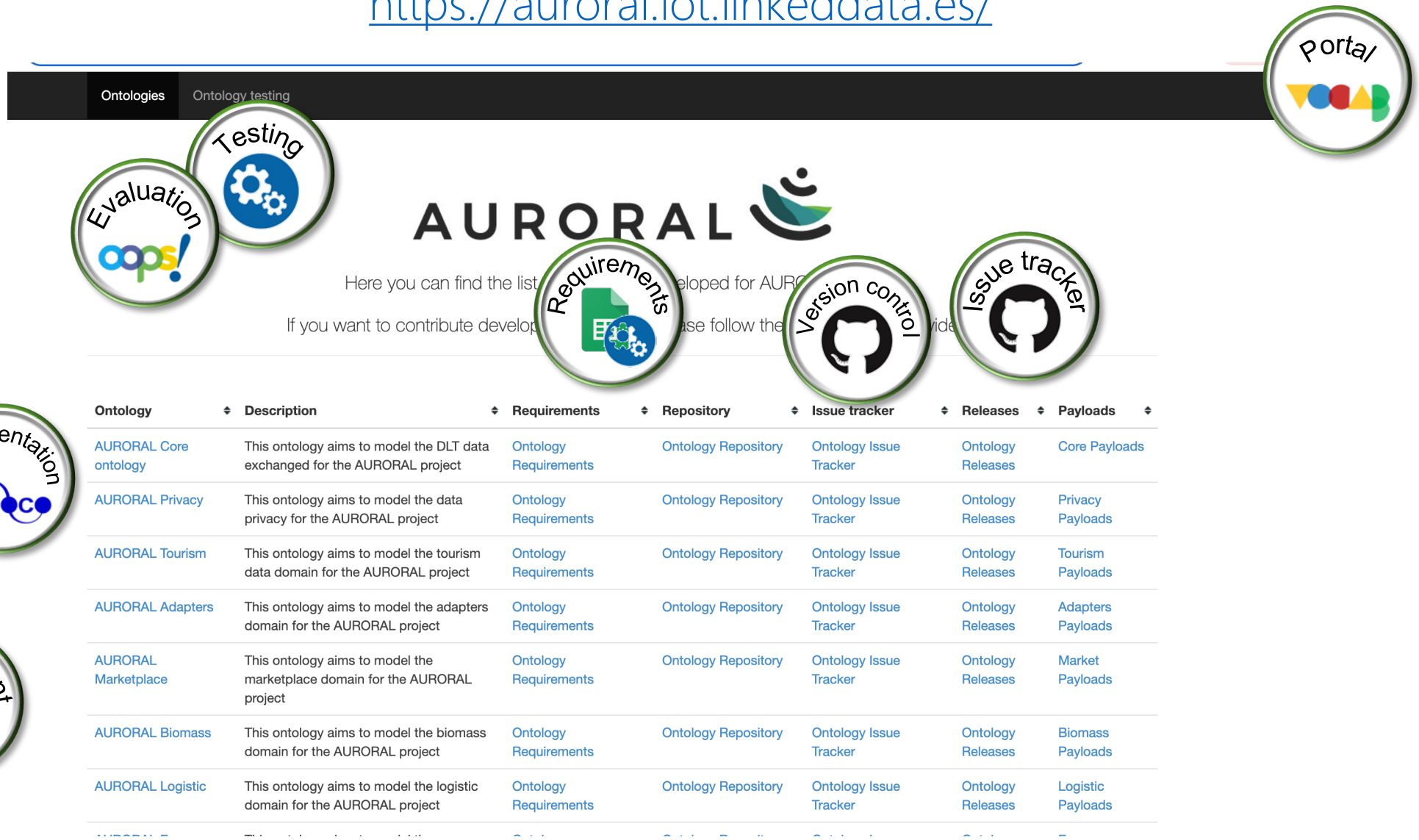


Chowlk

+ Diagrams and examples (Some ideas: https://arxiv.org/abs/2003.13084)
+ Some tools:
  + Draw.io
  + yEd graph editor
  + Microsoft Visio

- Own URI
- purl, w3id, etc.
- Content negotiation
- Registry

https://auroral.iot.linkeddata.es/



Ontology	Description	Requirements	Repository	Issue tracker	Releases	Payloads
AURORAL Core ontology	This ontology aims to model the DLT data exchanged for the AURORAL project	Ontology Requirements	Ontology Repository	Ontology Issue Tracker	Ontology Releases	Core Payloads
AURORAL Privacy	This ontology aims to model the data privacy for the AURORAL project	Ontology Requirements	Ontology Repository	Ontology Issue Tracker	Ontology Releases	Privacy Payloads
AURORAL Tourism	This ontology aims to model the tourism data domain for the AURORAL project	Ontology Requirements	Ontology Repository	Ontology Issue Tracker	Ontology Releases	Tourism Payloads
AURORAL Adapters	This ontology aims to model the adapters domain for the AURORAL project	Ontology Requirements	Ontology Repository	Ontology Issue Tracker	Ontology Releases	Adapters Payloads
AURORAL Marketplace	This ontology aims to model the marketplace domain for the AURORAL project	Ontology Requirements	Ontology Repository	Ontology Issue Tracker	Ontology Releases	Market Payloads
AURORAL Biomass	This ontology aims to model the biomass domain for the AURORAL project	Ontology Requirements	Ontology Repository	Ontology Issue Tracker	Ontology Releases	Biomass Payloads
AURORAL Logistic	This ontology aims to model the logistic domain for the AURORAL project	Ontology Requirements	Ontology Repository	Ontology Issue Tracker	Ontology Releases	Logistic Payloads

This slide has been taken from Raúl García Castro presentation at EMSE

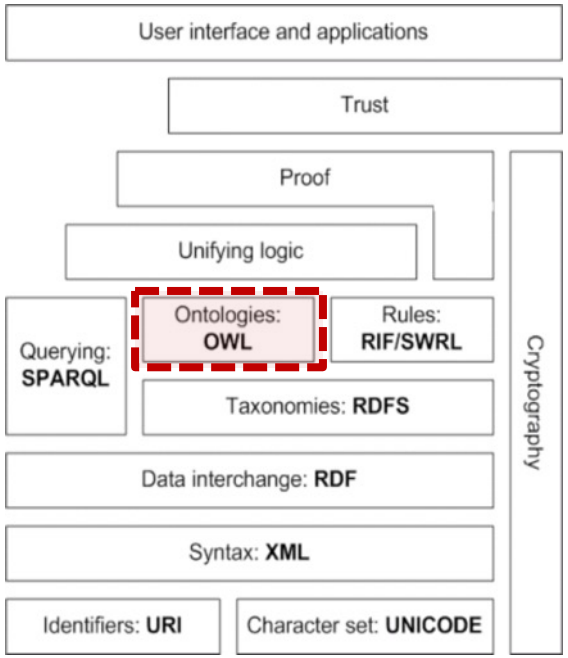
- Universities offer subjects
- A subject is delivered in one or more groups
- A subject for a given group is taught by only 1 professor
  - But in some departments this restriction does not apply
- A subject is offered only for 1 degree
- A university is located at some address
- An address has a street, a number and a postal code
- An address is located in a municipality
- A municipality could have borders with other municipalities
- A professor could be associate or assistant but not at the same time

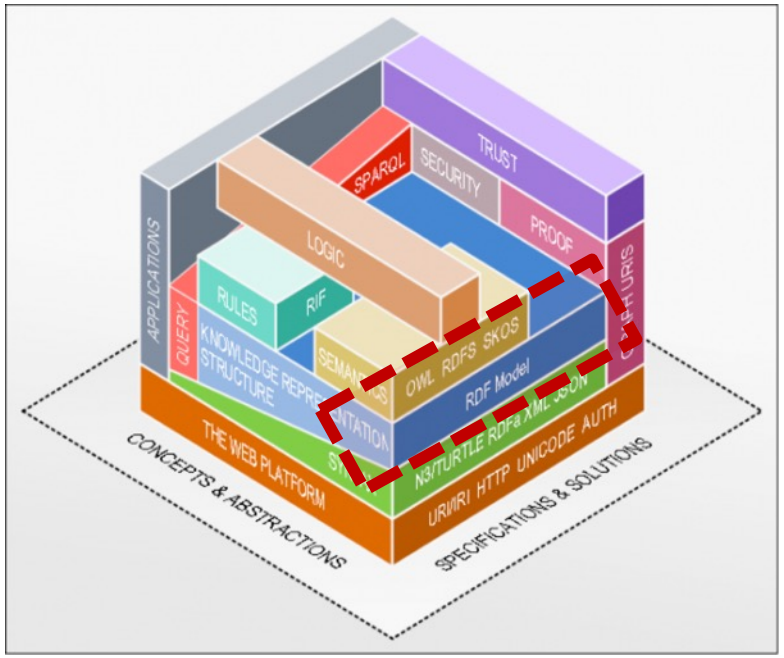
What can’t be modelled from the domain?

- Universities offer subjects
- A subject is delivered in one or more groups
- A subject for a given group is taught by **only 1** professor
  - But in some departments this restriction does not apply
- A subject is offered **only for 1** degree
- A university is located at some address
- An address has a street, a number and a postal code
- An address is located in a municipality
- A municipality could have borders with other municipalities.
- A professor could be associate or assistant but **not at the same time**

If A has border with B we might want to infer that B has border with A too

- RDFS is **too weak** to describe resources in sufficient detail
  - No **localised range and domain** constraints
    - Can't say that the range of hasChild is person when applied to persons and elephant when applied to elephants
  - No **existence/cardinality** constraints
    - Can't say that all *instances* of person have a mother that is also a person, or that persons have exactly 2 parents
  - No **boolean** operators
    - Can't say or, not, etc.
  - No **transitive, inverse or symmetrical** properties
    - Can't say that isPartOf is a transitive property, that hasPart is the inverse of isPartOf or that touches is symmetrical



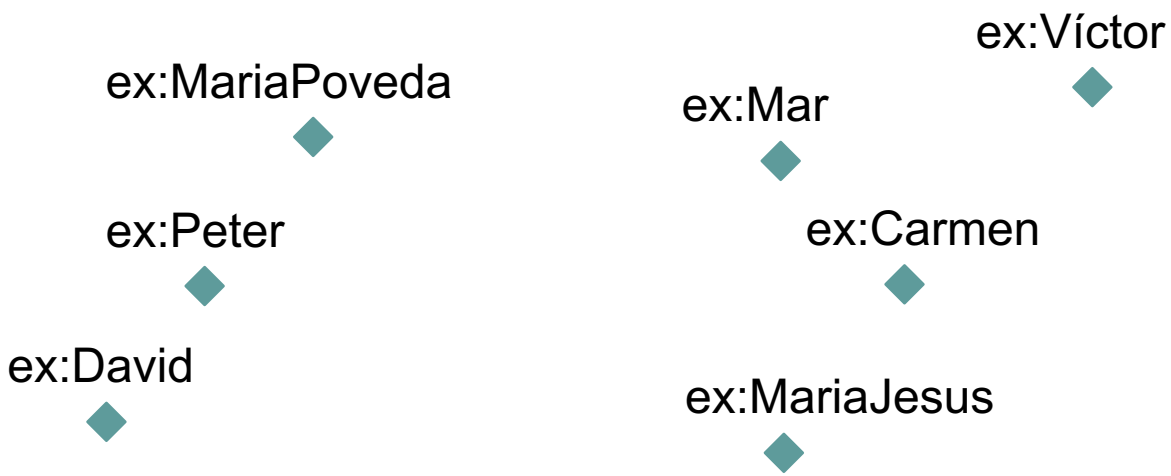


http://w3.org/DesignIssues/diagrams/sweb-stack/2006a.png

- OWL: **W**eb **O**ntology **L**anguage
- Goal
  - To describe the semantic of the information in a machine-readable way
- Based on Description Logics (DL)
  - To describe a domain based on its concepts (classes), roles (relationships) and individuals (instances)
    - Specific languages characterized by the constructs and axioms used to declare knowledge about classes, relations and individuals

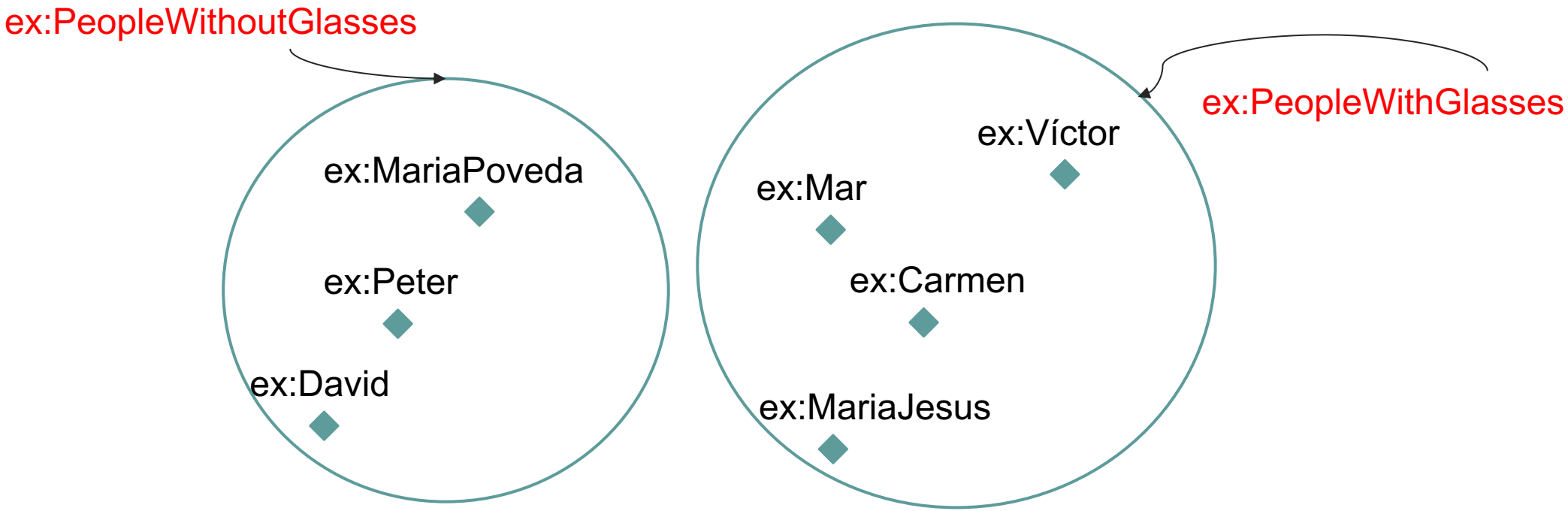
OWL is a FORMAL language

- What is a class?
  - A group of individuals that have a common characteristic



Universe

- What is a class?
  - A group of individuals that have a common characteristic

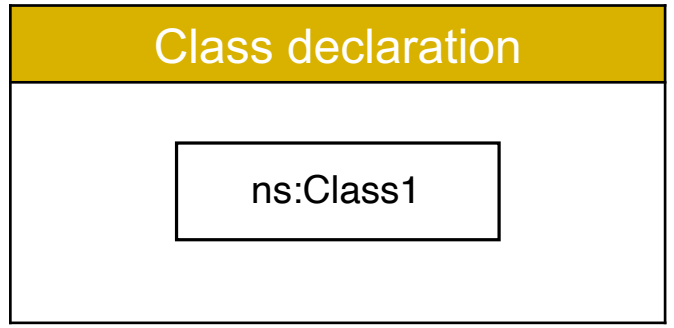


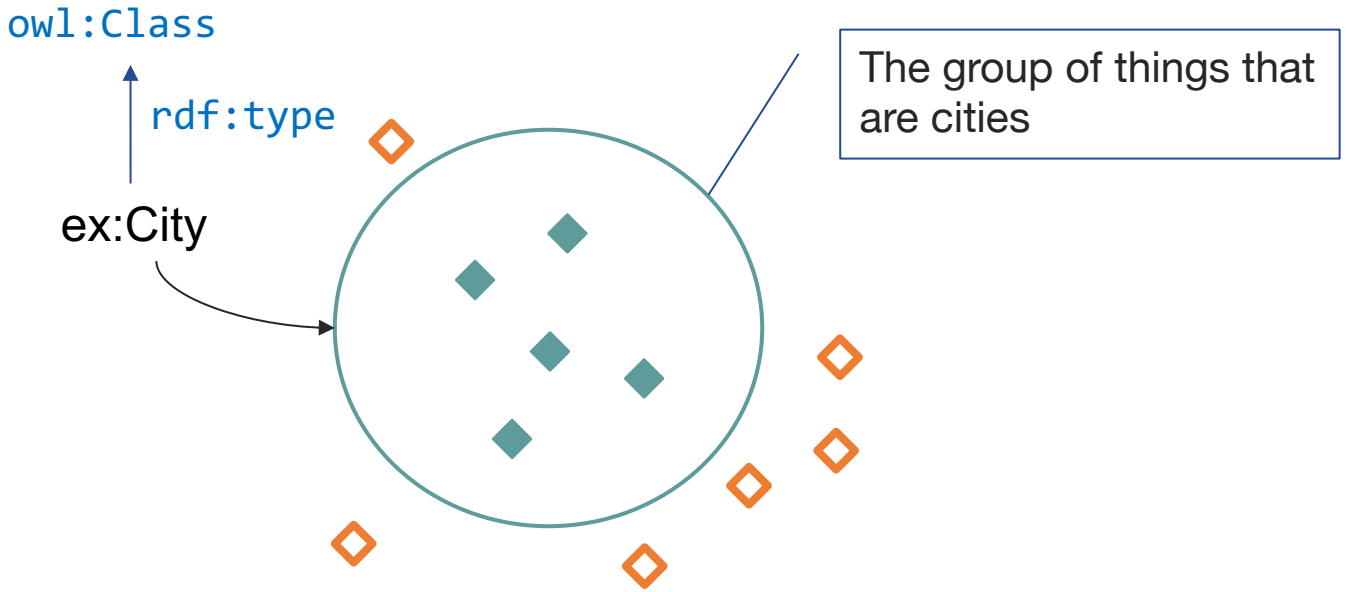
Universe

- Remember from set theory:
  - By extension:
    - {Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday}
  - By intension: “People with glasses in the room”

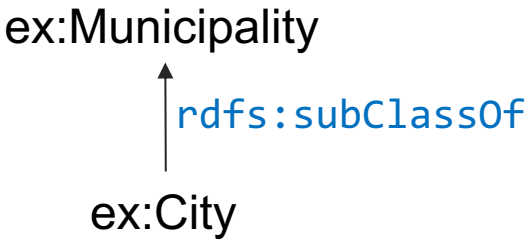
- … and using **OWL**?
  - With an identifier: assigning a URI
  - With an exhaustive enumeration of individuals
  - As constraints about a property
  - As union of classes
  - As intersection of classes
  - As complement of a class

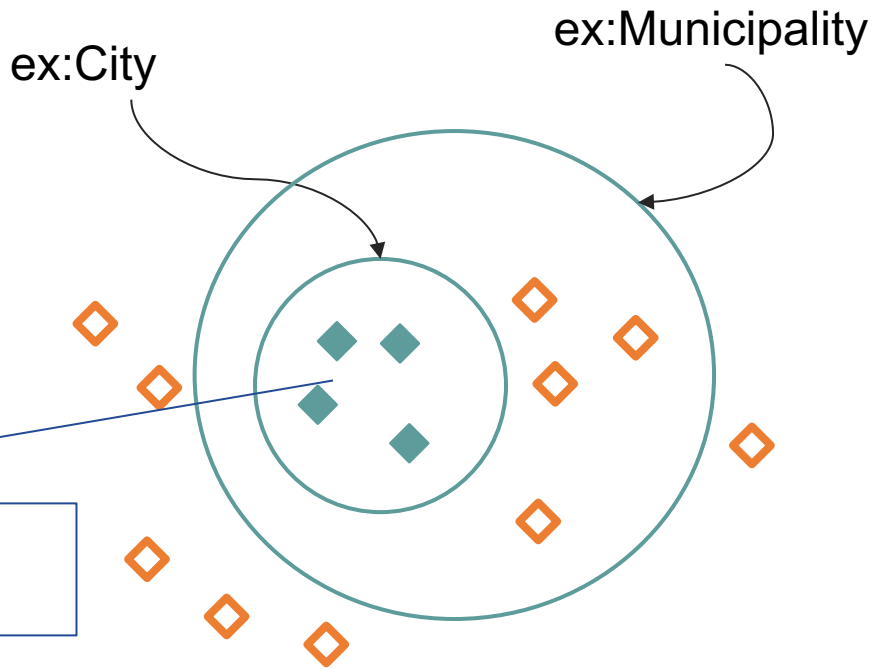
- **owl:Class**
  - Concepts of the domain (generally)
  - **Named**:
    - URI as identifier
  - Not named:
    - Enumeration, constraints over properties, intersection, union, complement.



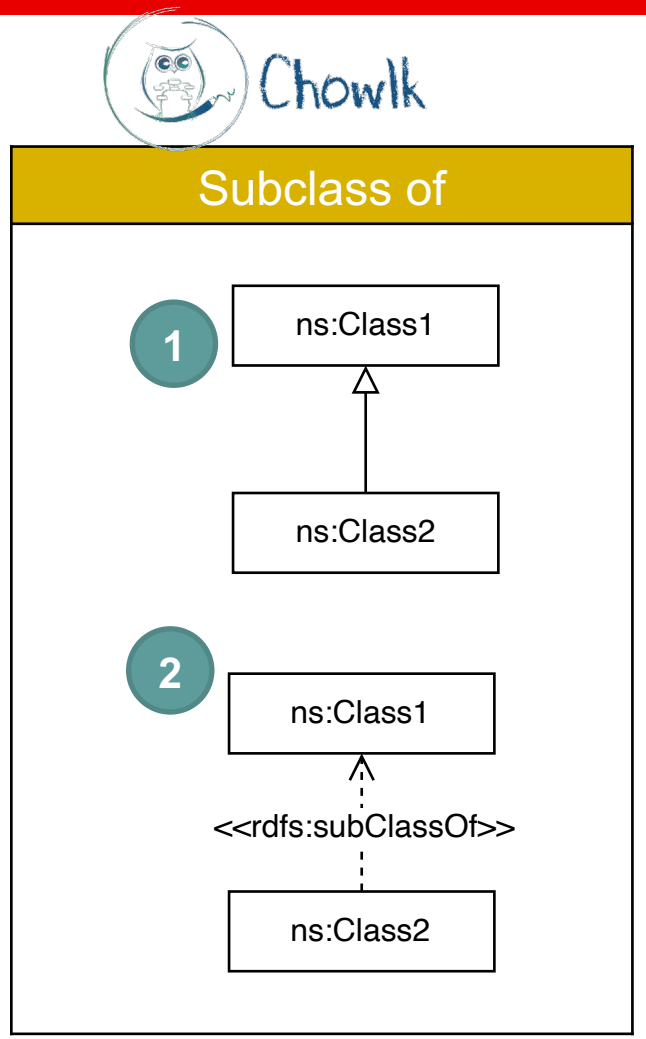


- **rdfs:subClassOf**
  - The individuals belonging to a class also belong to the parent classes in the **hierarchy**

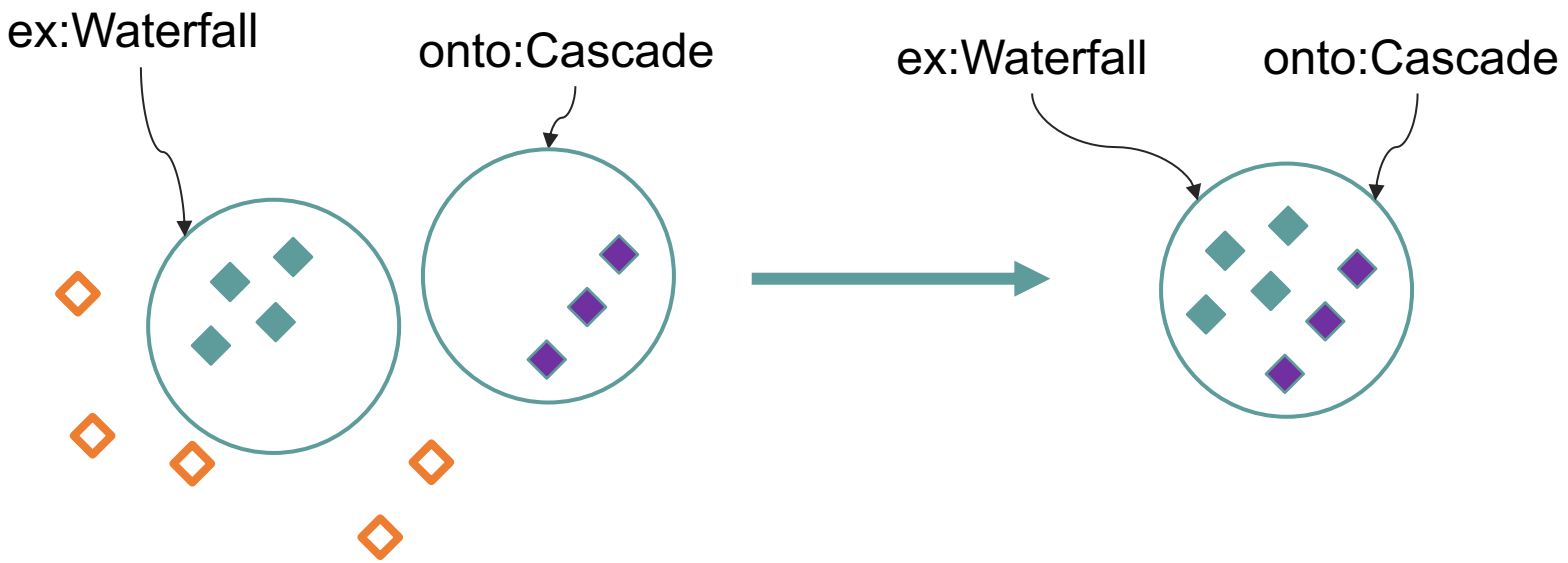


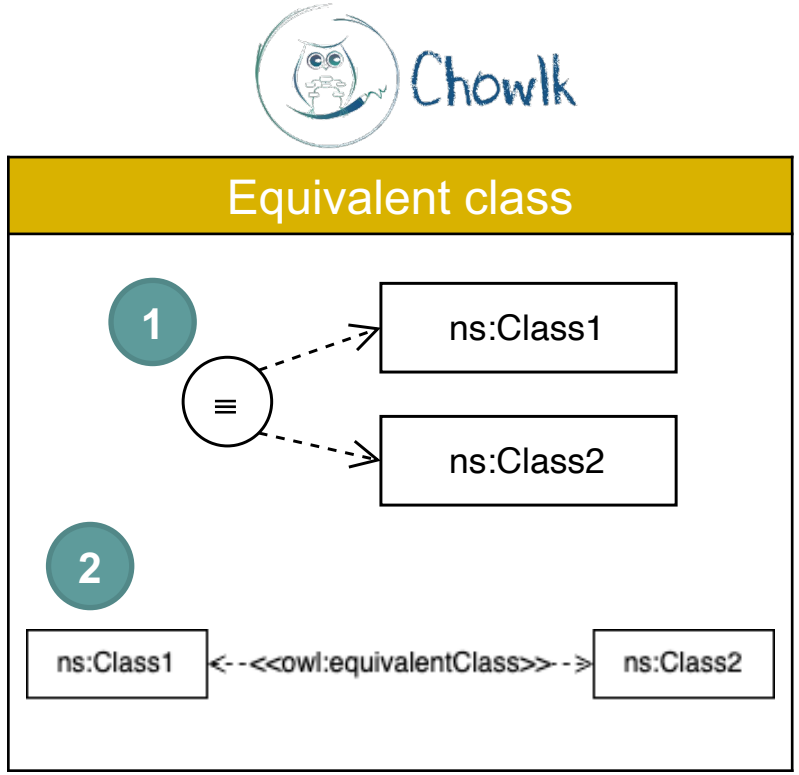


The group of things that are municipalities and are **also** cities

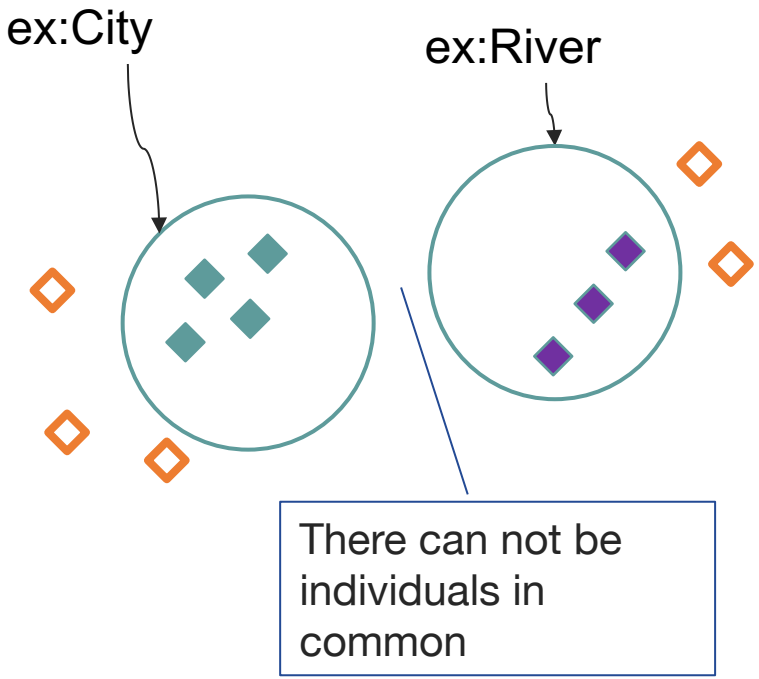


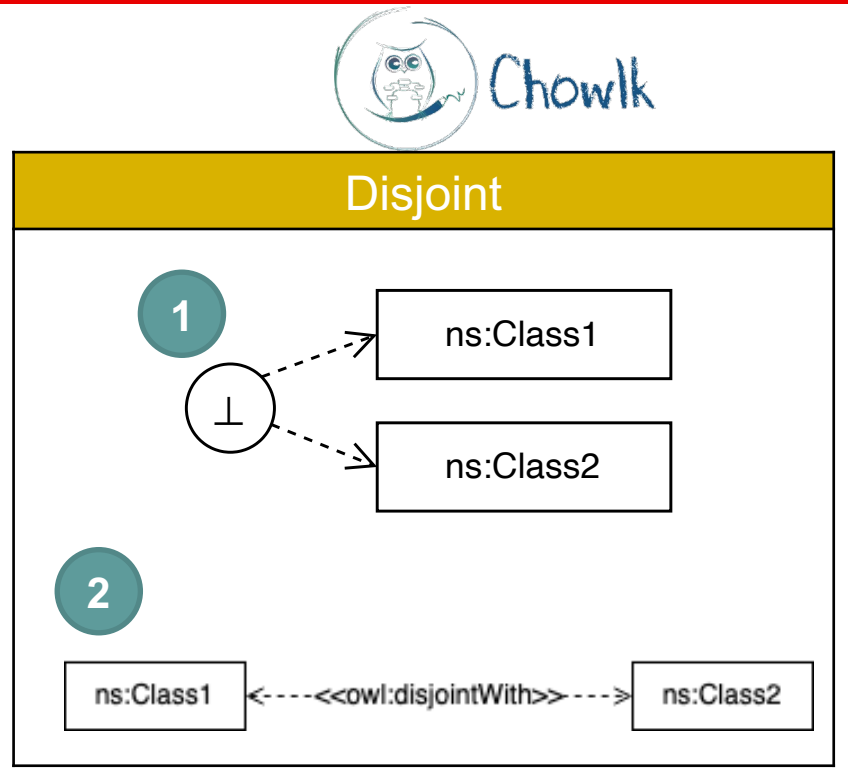
- **owl:equivalentClass**
  - The two classes contain exactly the same individuals
  - If an individual belongs to a clas it also belongs to the other



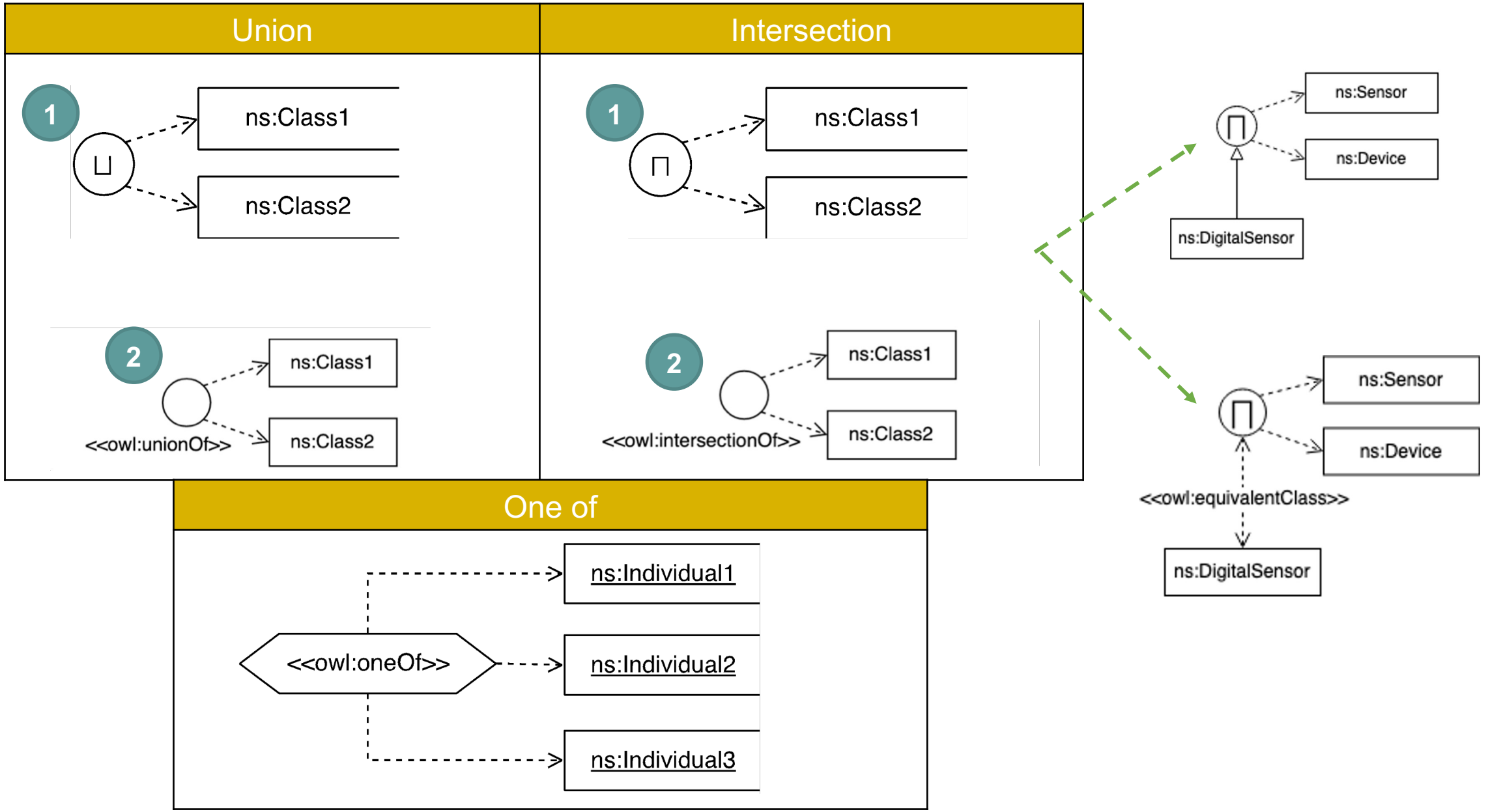


- **owl:disjointWith**
  - An individual can not belong to more than one of the involved classes
  - The intersection is the empty set

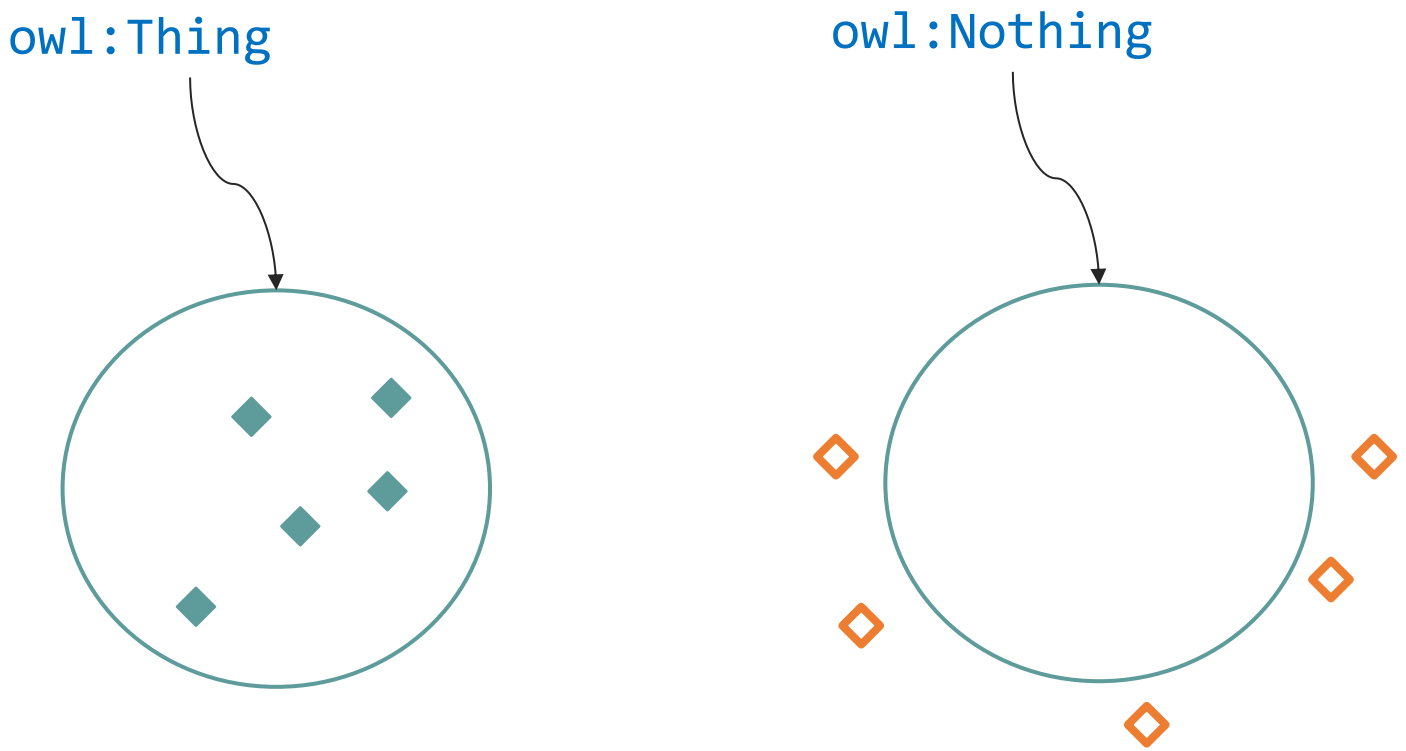




Also…


Union
1
ns:Class1
ns:Class2
2
ns:Class1
<<owl:unionOf>>
ns:Class2
Intersection
1
ns:Class1
ns:Class2
2
ns:Class1
<<owl:intersectionOf>>
ns:Class2
ns:Sensor
ns:Device
ns:DigitalSensor
ns:Sensor
ns:Device
<<owl:equivalentClass>>
ns:DigitalSensor
One of
ns:Individual1
<<owl:oneOf>>
ns:Individual2
ns:Individual3

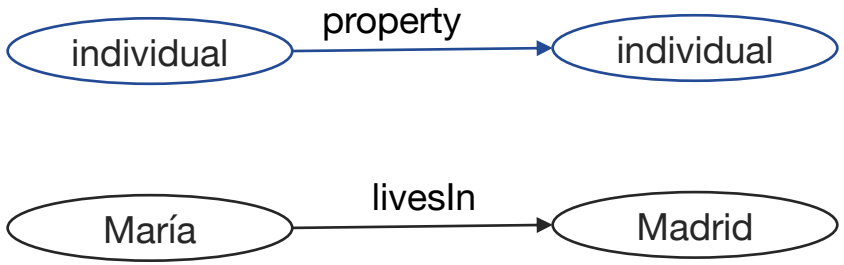
- OWL predefine two classes: Thing y Nothing. All and Nothing



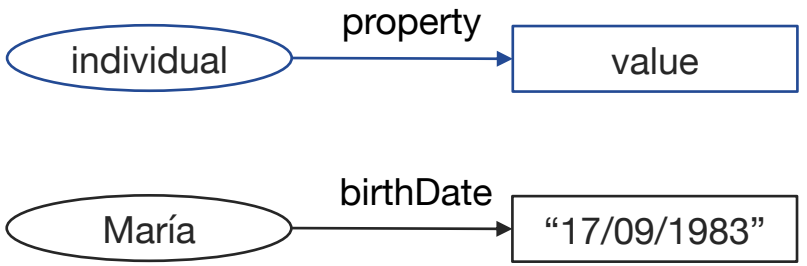
- All individuals in OWL are instances of owl:Thing.
- Then, all classes in OWL are subclassses of owl:Thing

rdf:Property

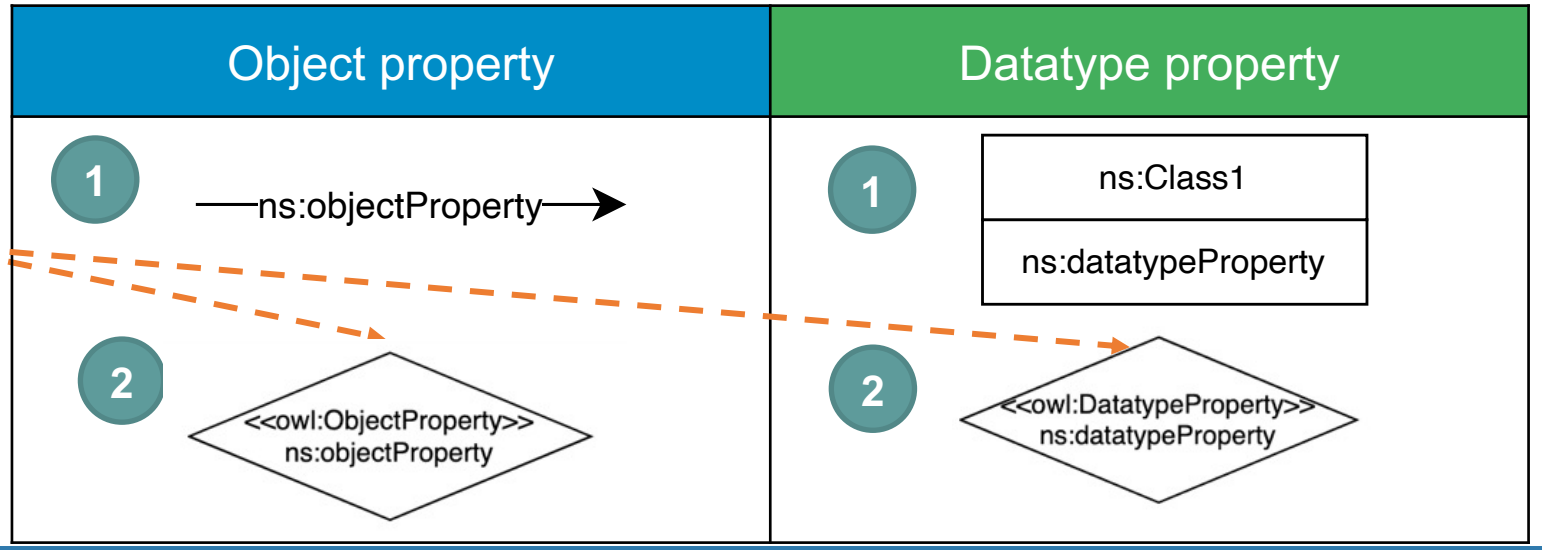
owl:ObjectProperty
≈ relationships



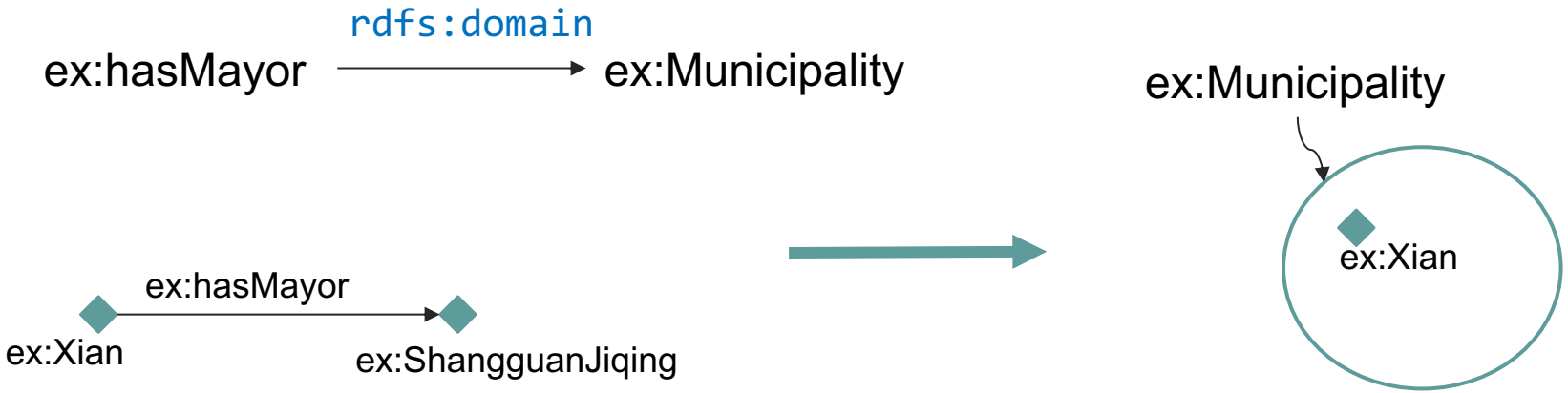
owl:DatatypeProperty
≈ attributes



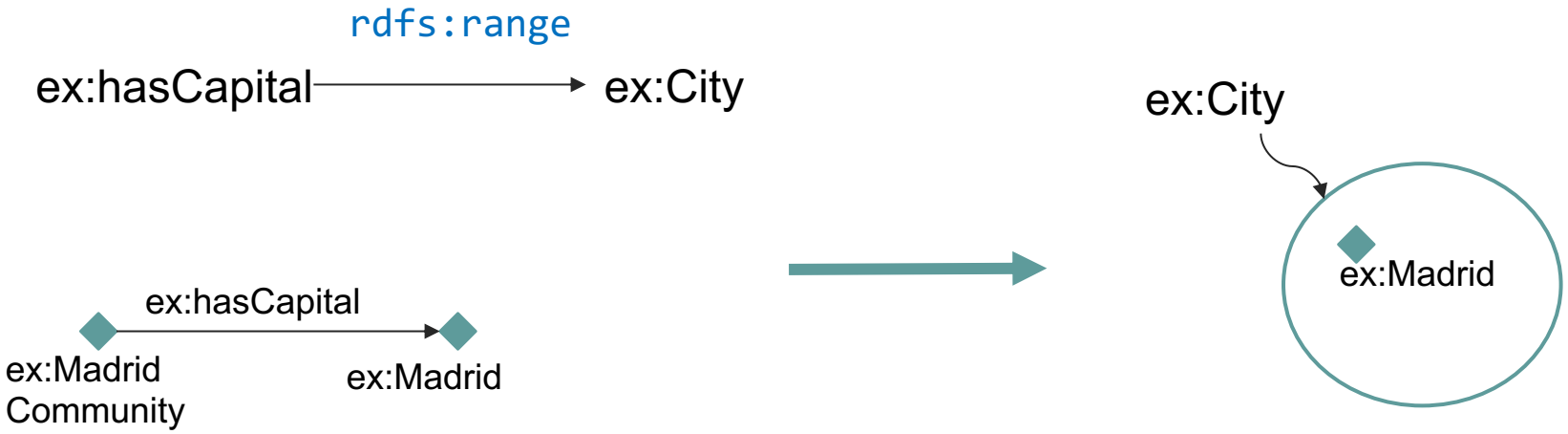
- Used for property hierarchies, inverse, equivalents, complex domain/range…

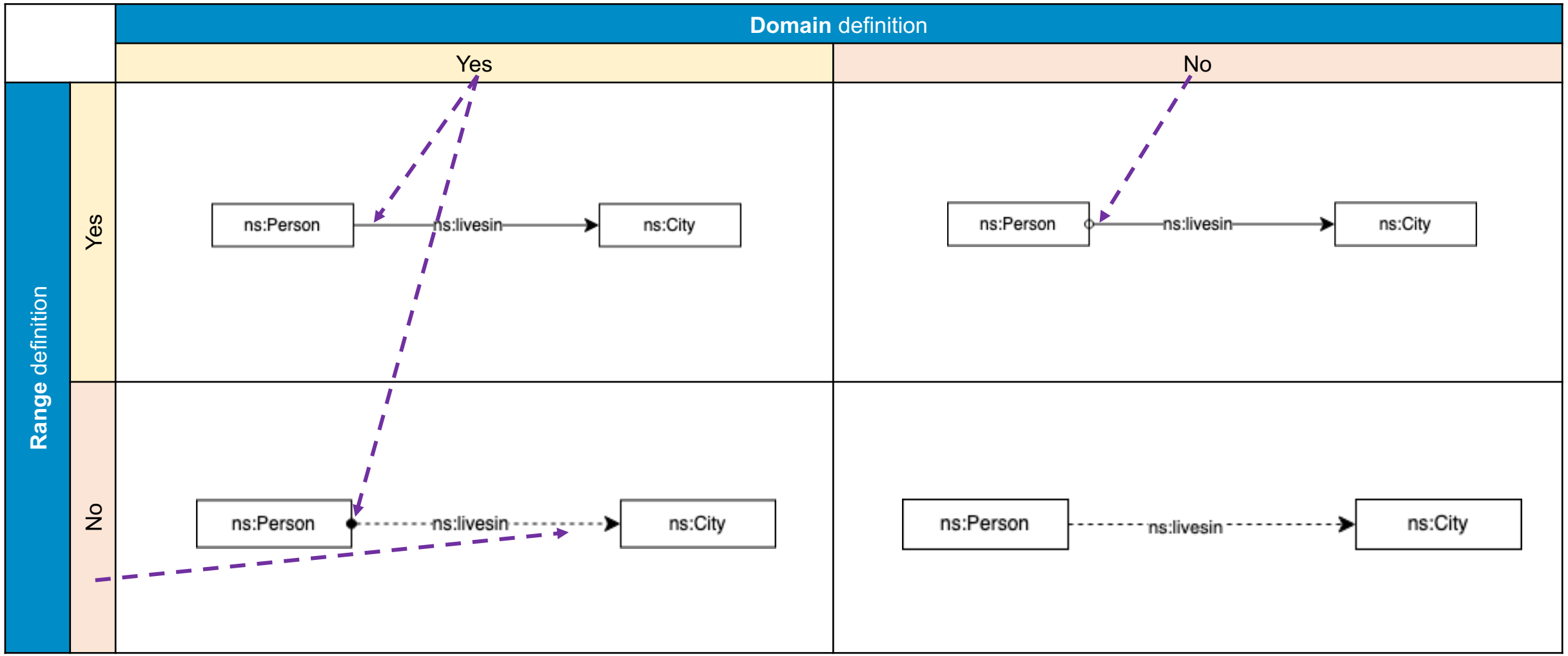


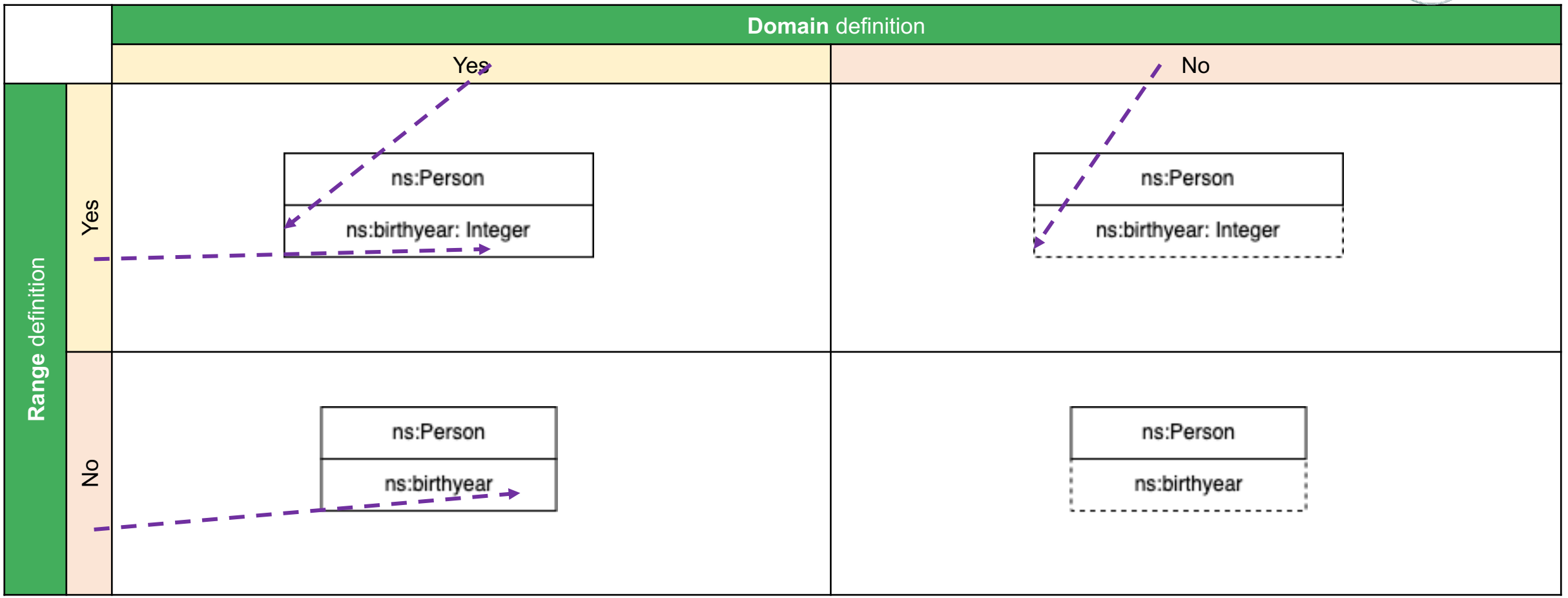
- **rdfs:domain**
  - All individuals that appear as **subject** in a triple with the given property **belongs to** the **class** defined as **domain** of the property
  - It is valid for relations and attributes



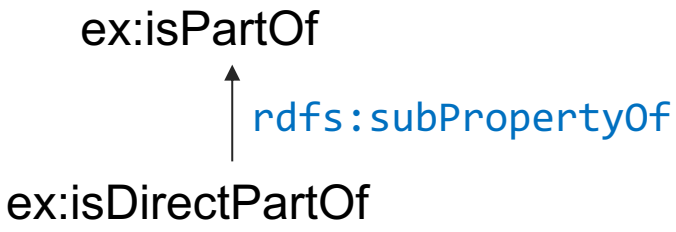
- **rdfs:range**
  - All individuals that appear as **object** in a triple with the given property **belongs to** the **class** defined as **range** of the property
  - It is valid for clases (applies to relations) or datatypes (applies to attributes)

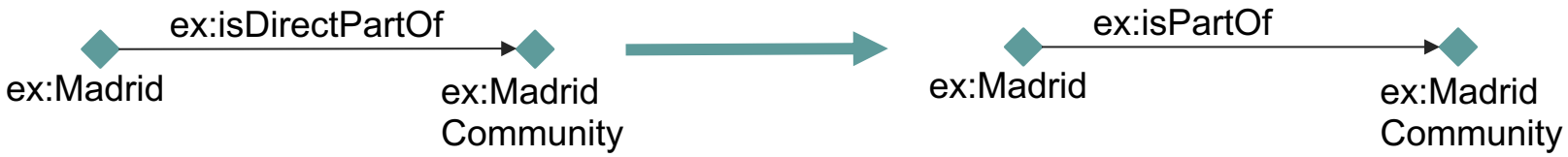


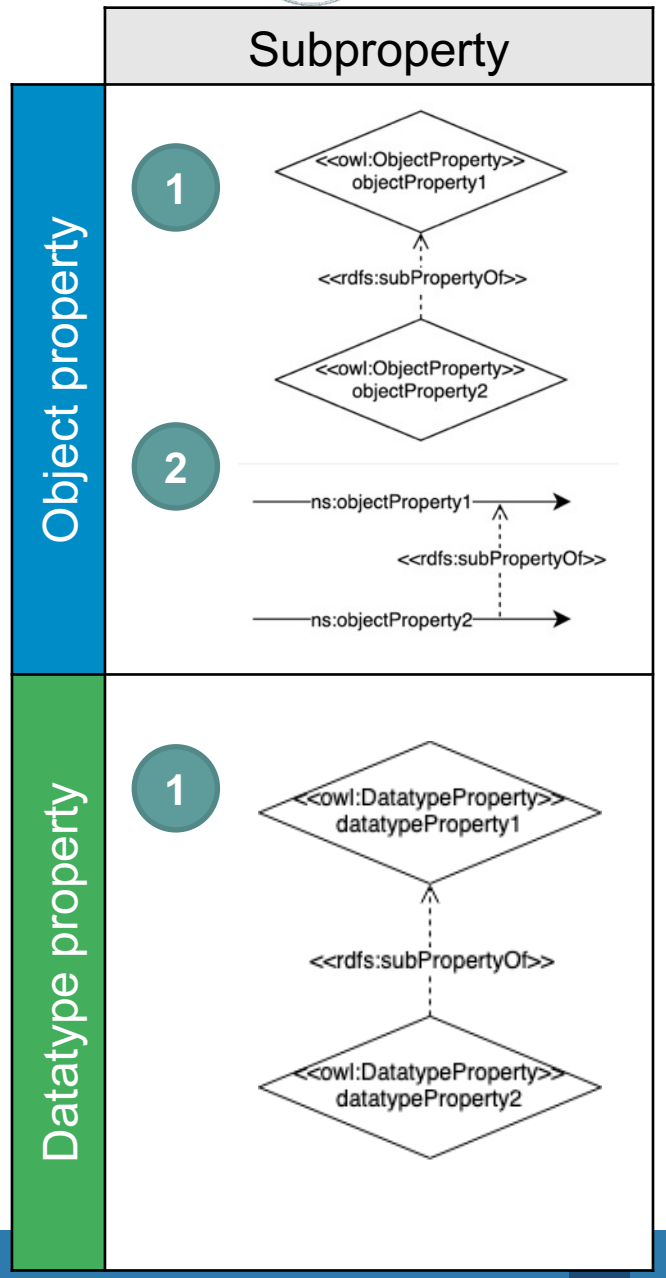

Domain definition
Yes
No
Range definition
Yes
No
ns:Person
ns:livesin
ns:City
ns:Person
ns:livesin
ns:City
ns:Person
ns:livesin
ns:City
ns:Person
ns:livesin
ns:City


Domain definition
Yes
No
Range definition
Yes
No
ns:Person
ns:birthyear: Integer
ns:Person
ns:birthyear: Integer
ns:Person
ns:birthyear
ns:Person
ns:birthyear

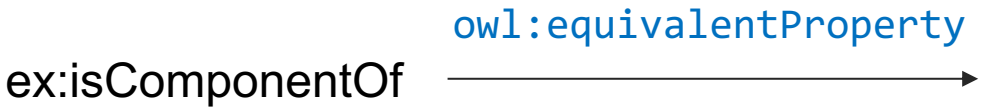
- **rdfs:subPropertyOf**
  - When there is a property between two elements (two individuals or an individual and a value), the parent property in the hierarchy is also true for the pair of elements
  - Aplicable to *object properties (relations)* and *datatype properties (attributes)*.

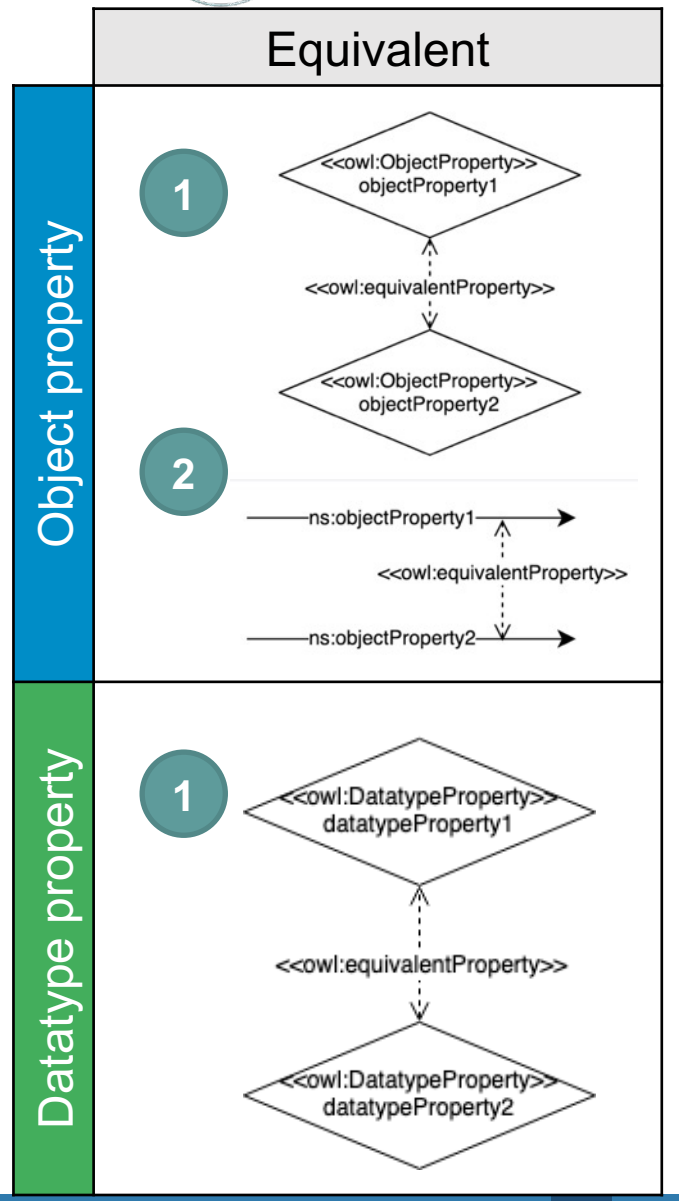




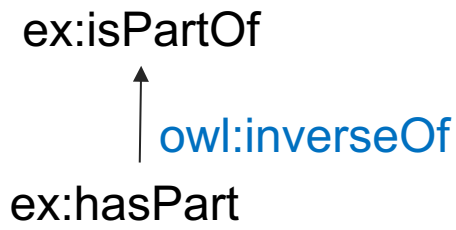


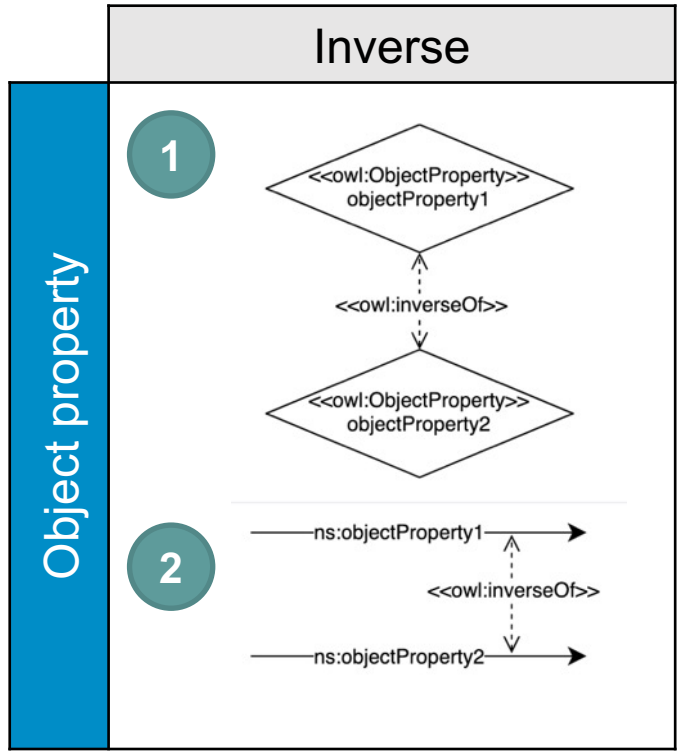
- **owl:equivalentProperty**
  - When there is a property between two elements (two individuals or an individual and a value), the equivalent property is also true for the pair of elements
  - Aplicable to ***object*** *properties (relations)* and ***datatype*** *properties (attributes)*.

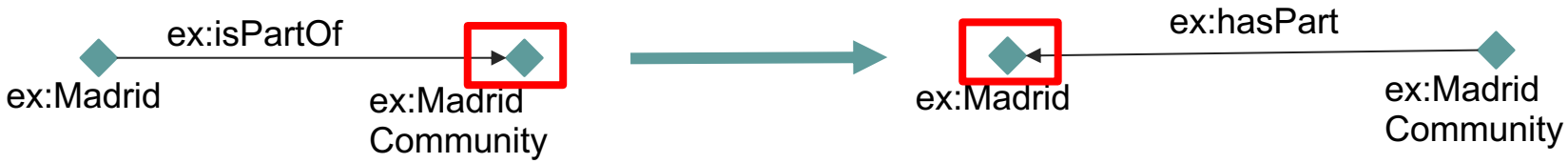




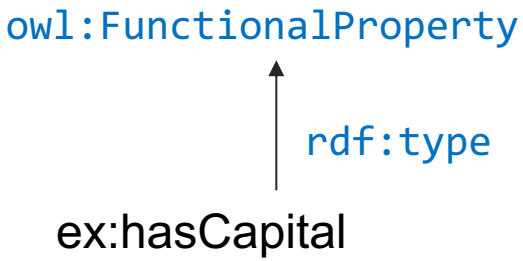
- **owl:inverseOf**
  - When there is a relations between two individuals A (subject) and B (object), the inverse relation is true between the individuals B (subject) and A (object).
  - Aplicable **only** to *object properties*.

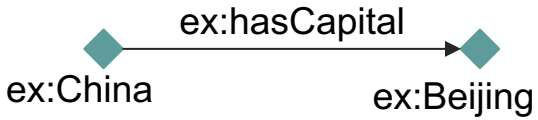


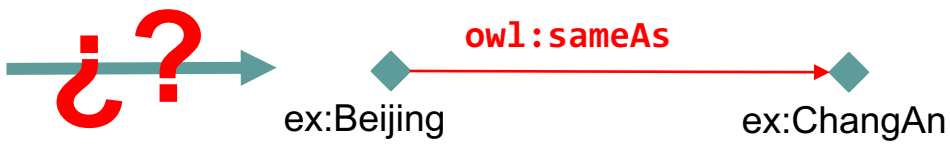


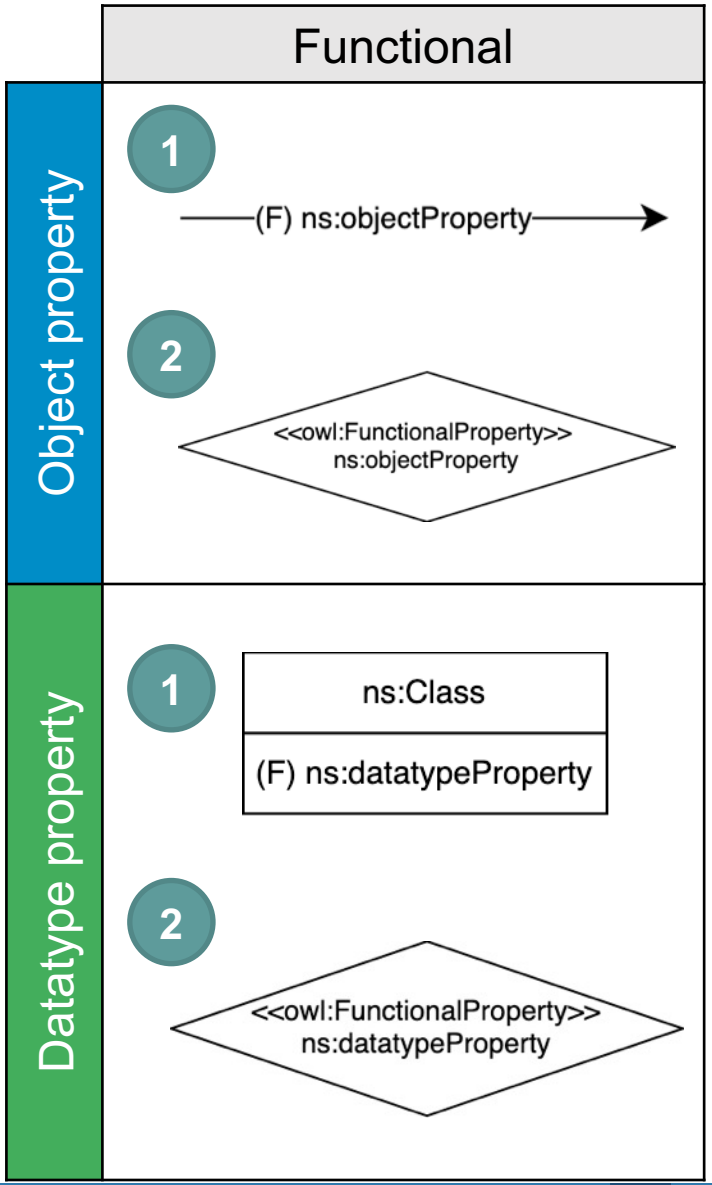


- **owl:FunctionalProperty**
  - It can only take one value per individual in the subject.
  - Aplicable to ***object*** *properties* and ***datatype*** *properties*.





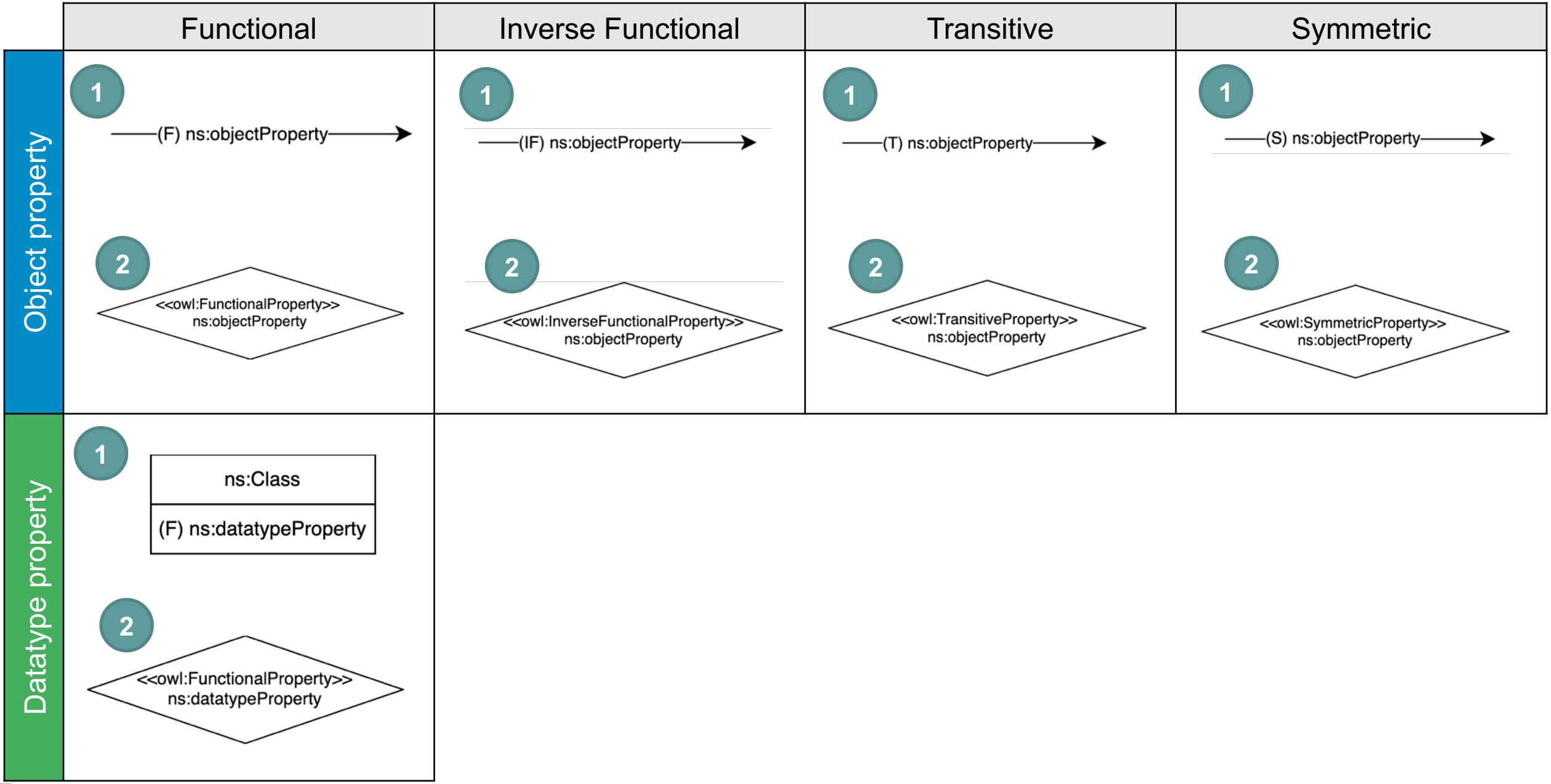




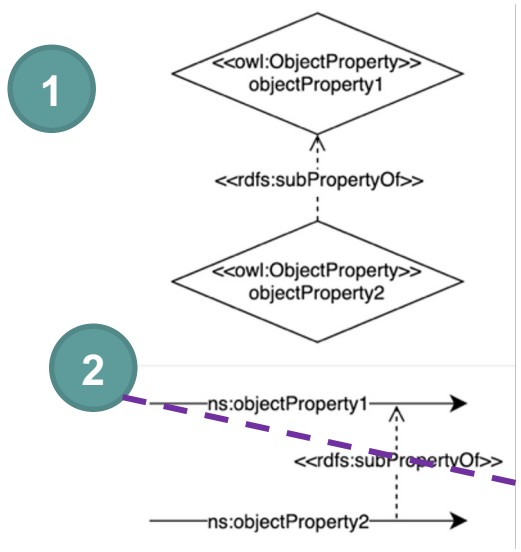
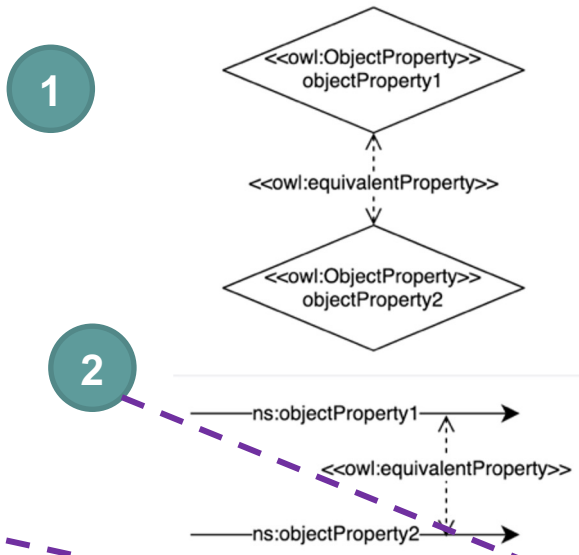
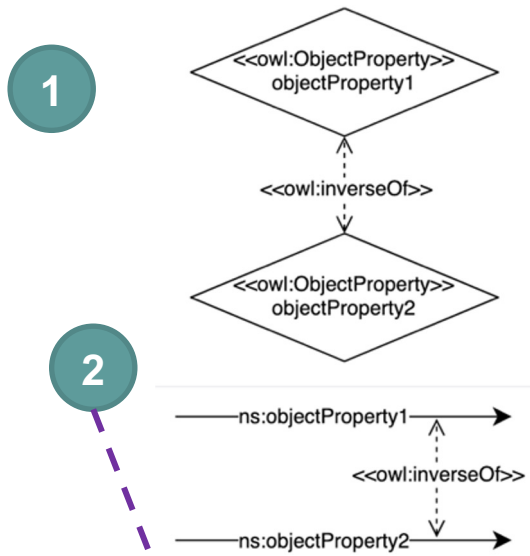
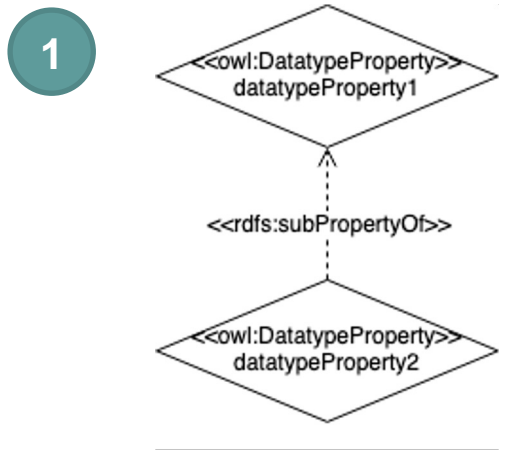
Also

- owl:InverseFunctionalProperty
- owl:TransitiveProperty
- owl:SymmetricProperty

Properties characteristics

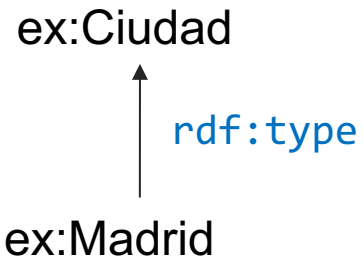


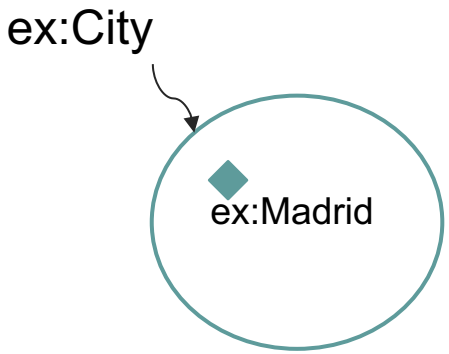
Relations between properties

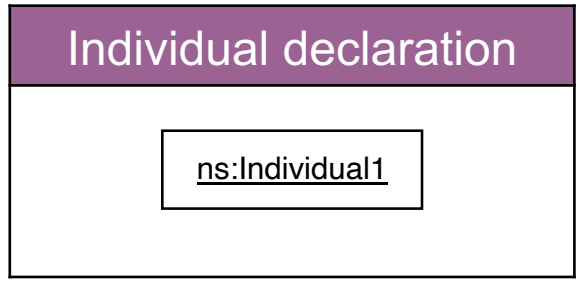
	Subproperty	Equivalent	Inverse
Object property			
Datatype property			

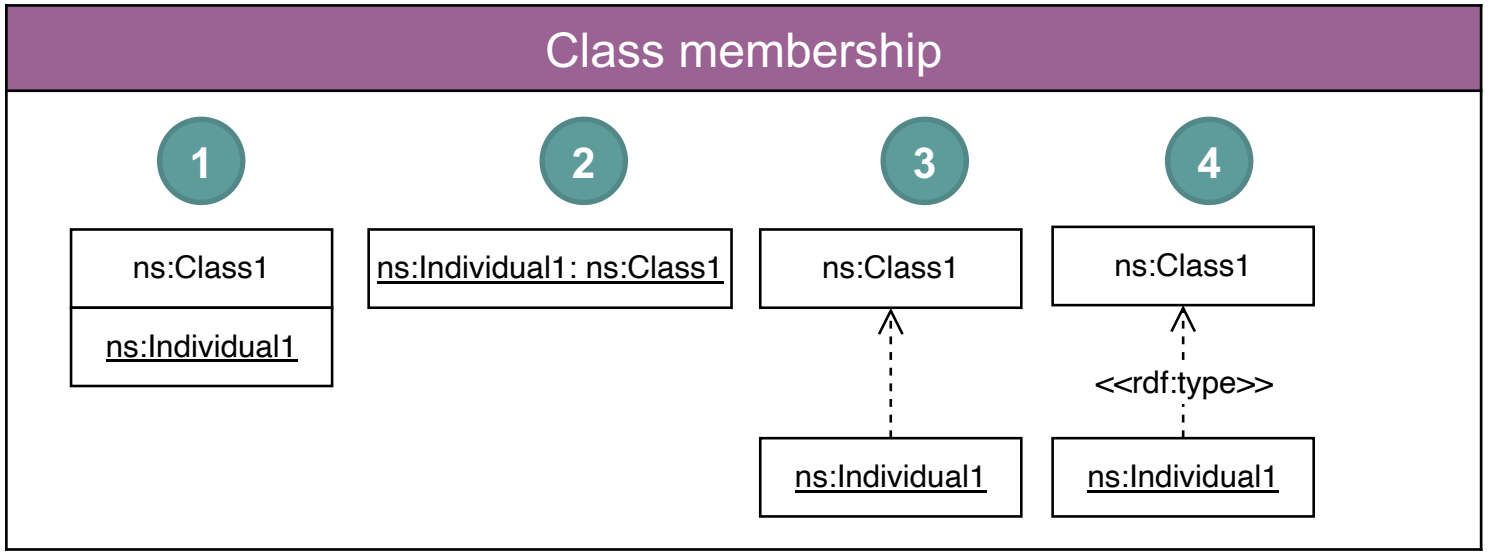
Not really… but technical issues

- **rdf:type**
  - Indicates the class or classes the individual belongs to

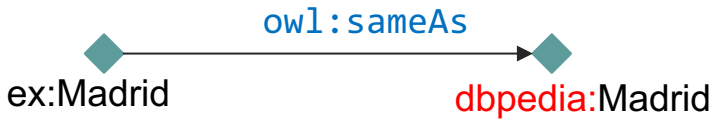




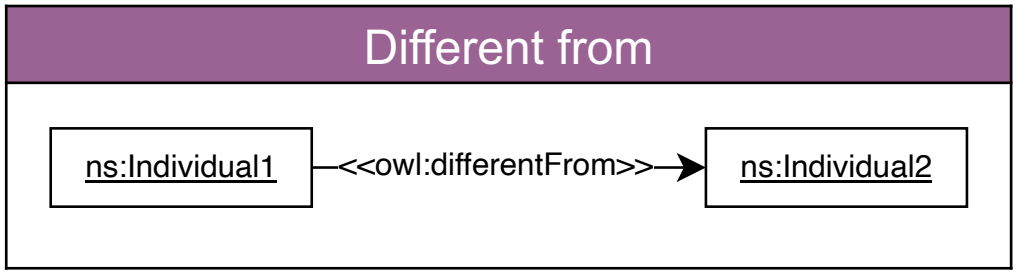




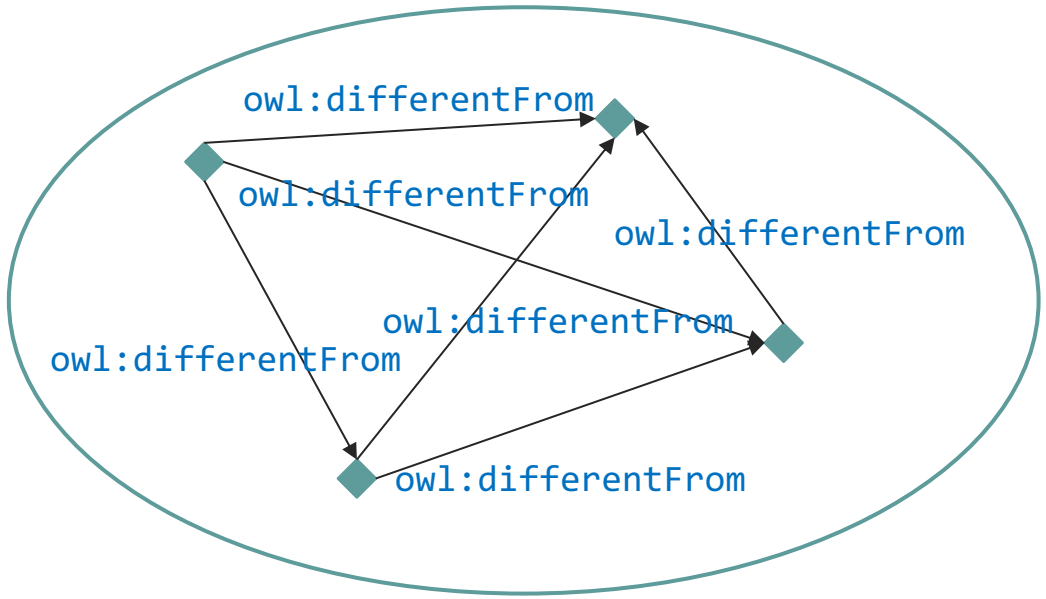
- **owl:sameAs**
  - Declare that two URIs identify the same individual
  - It is defined between instances or individuals

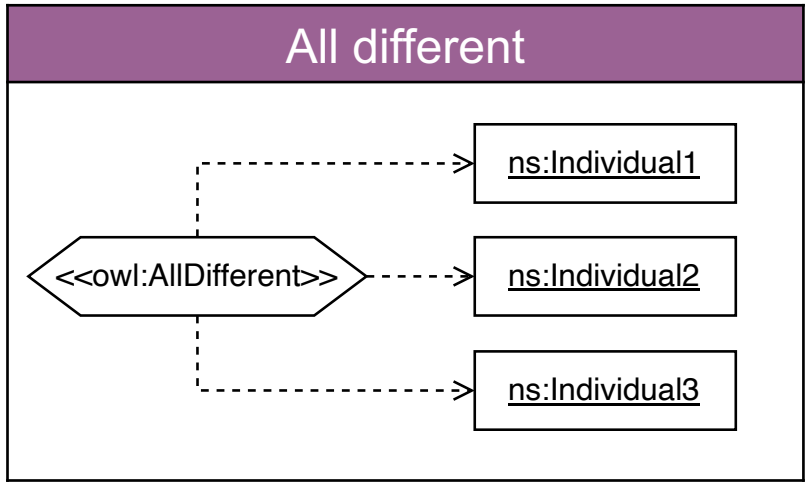


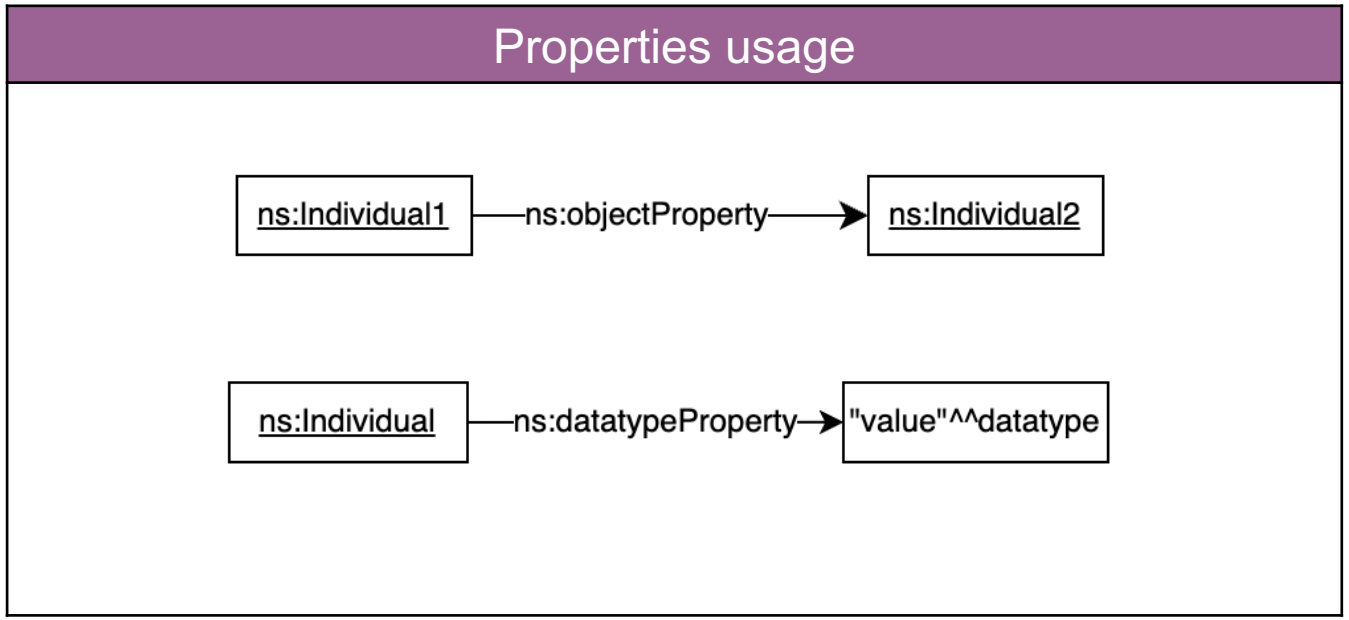
- **owl:differentFrom**
  - Declare that two URIs identify different individuals



- **owl:AllDifferent**
  - Declare that all URIs indicated identigy different individuals
  - Normally used to force the *unique name assumption*





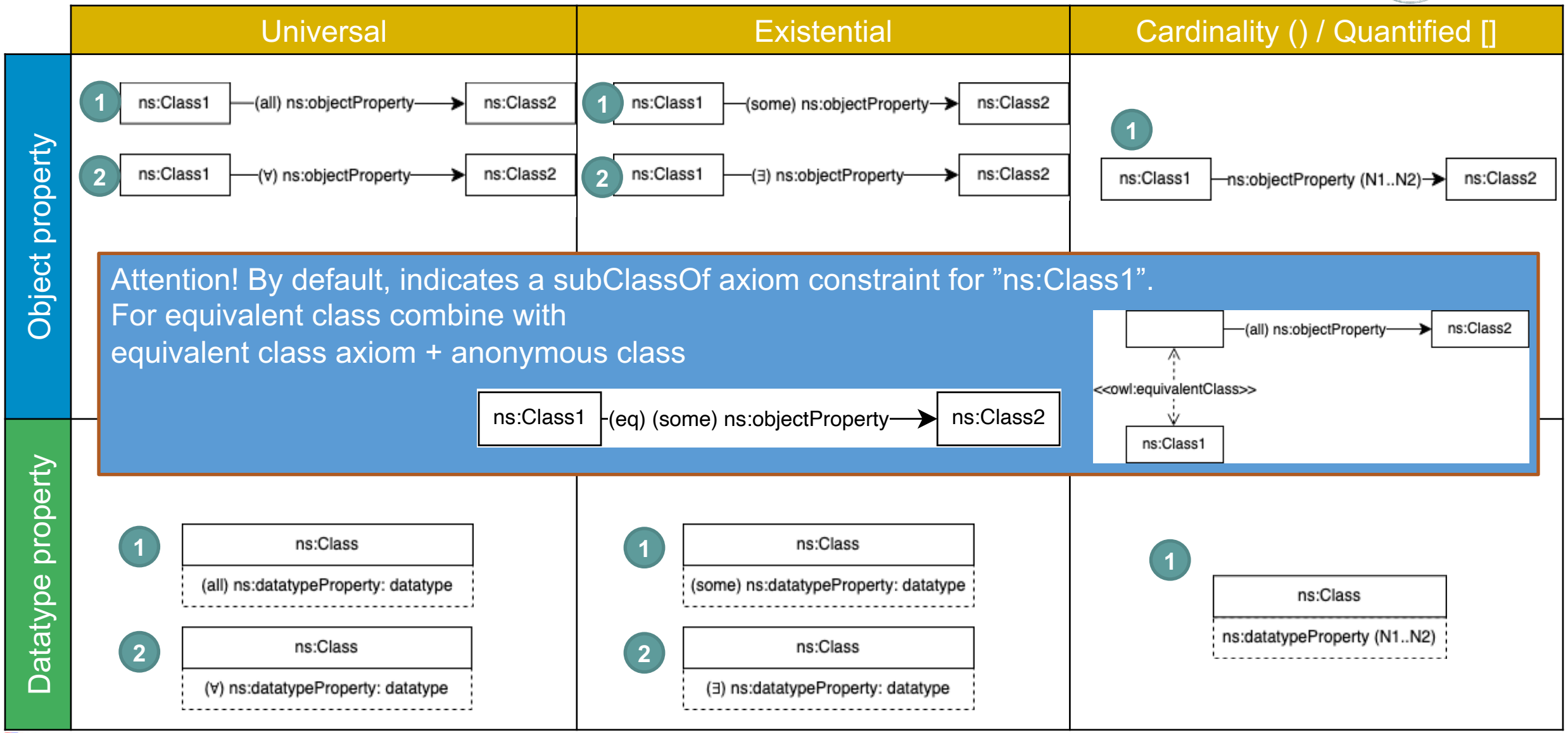

Properties usage
ns:Individual1
ns:objectProperty
ns:Individual2
ns:Individual
ns:datatypeProperty
"value"^^datatype

- *Open World Assumption vs Closed World Assumption*
  - OWL follows OWA
  - The lack of evidence about a fact does not imply that the fact is false.
- *Non unique name assumption*
  - Different names do not identified necessarily different individuals.
- OWL 2
  - Additional constructs

- OWL
  - Classes
    - Axioms
  - Properties
    - Characteristics
  - Individuals
  - **Restrictions**
    - If there is still time, just have fun with Protégé

- *A constraint limit the individuals that can belong to a class: they have to satify some properties*
- *Example*
  - *”RedWine” is a class*
  - *”My Rioja bottle” would be an instance of that class. A bottle of milk would not be an instance of that class. But both would be instance of “Bottle”.*
- *“Is subclass of “red things””. The restrictions are imposed using **subclassOf** or **equivalentClass** axioms.*
  - *Because the sets described by the constrains is an anonymous class*

Slide taken from Víctor Rodríguez Doncel


Universal
Existential
Cardinality () / Quantified []
Object property
Datatype property
1 ns:Class1 (all) ns:objectProperty ns:Class2
2 ns:Class1 (∀) ns:objectProperty ns:Class2
1 ns:Class1 (some) ns:objectProperty ns:Class2
2 ns:Class1 (∃) ns:objectProperty ns:Class2
1 ns:Class1 ns:objectProperty (N1..N2) ns:Class2
Attention! By default, indicates a subClassOf axiom constraint for "ns:Class1".
For equivalent class combine with
equivalent class axiom + anonymous class
ns:Class1 (eq) (some) ns:objectProperty ns:Class2
(all) ns:objectProperty ns:Class2
<<owl:equivalentClass>>
ns:Class1
1 ns:Class (all) ns:datatypeProperty: datatype
2 ns:Class (∀) ns:datatypeProperty: datatype
1 ns:Class (some) ns:datatypeProperty: datatype
2 ns:Class (∃) ns:datatypeProperty: datatype
1 ns:Class ns:datatypeProperty (N1..N2)

Advanced ontologies and reasoning

María Poveda Villalón, Ontology Engineering Group
Universidad Politécnica de Madrid, Spain

[mpoveda@fi.upm.es] @MariaPovedaV SSoLDAC24