



Advanced ontologies and reasoning

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



✉ [mpoveda@fi.upm.es]

🐦 @MariaPovedaV

📍 SSoLDAC23

- 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 and Alba Fernández Izquierdo”
 - *Non-commercial*
 - *Share-Alike*

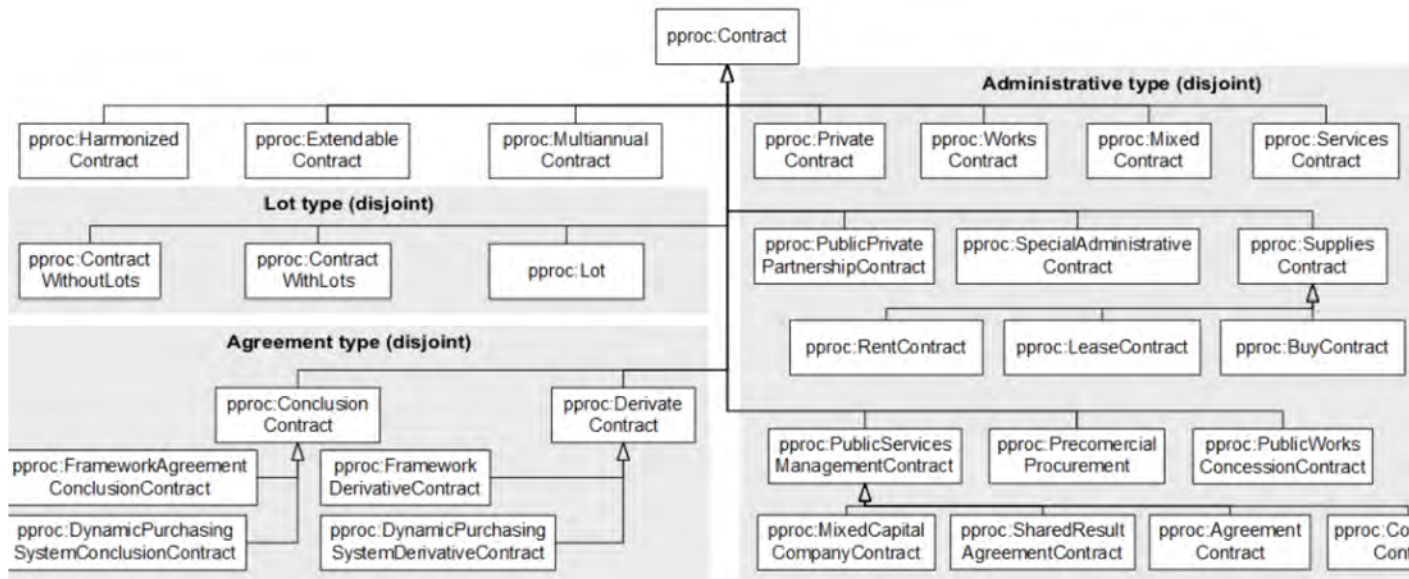
- The materials for this session has been elaborated by María Poveda Villalón reusing content generated by the following OEG colleagues:
 - Oscar corcho
 - Raúl García-Castro
 - Alba Fernández-Izquierdo
 - Etc.



A vocabulary, built with some consensus, and described formally

Define terms, how they are classified, their properties, relations, etc.

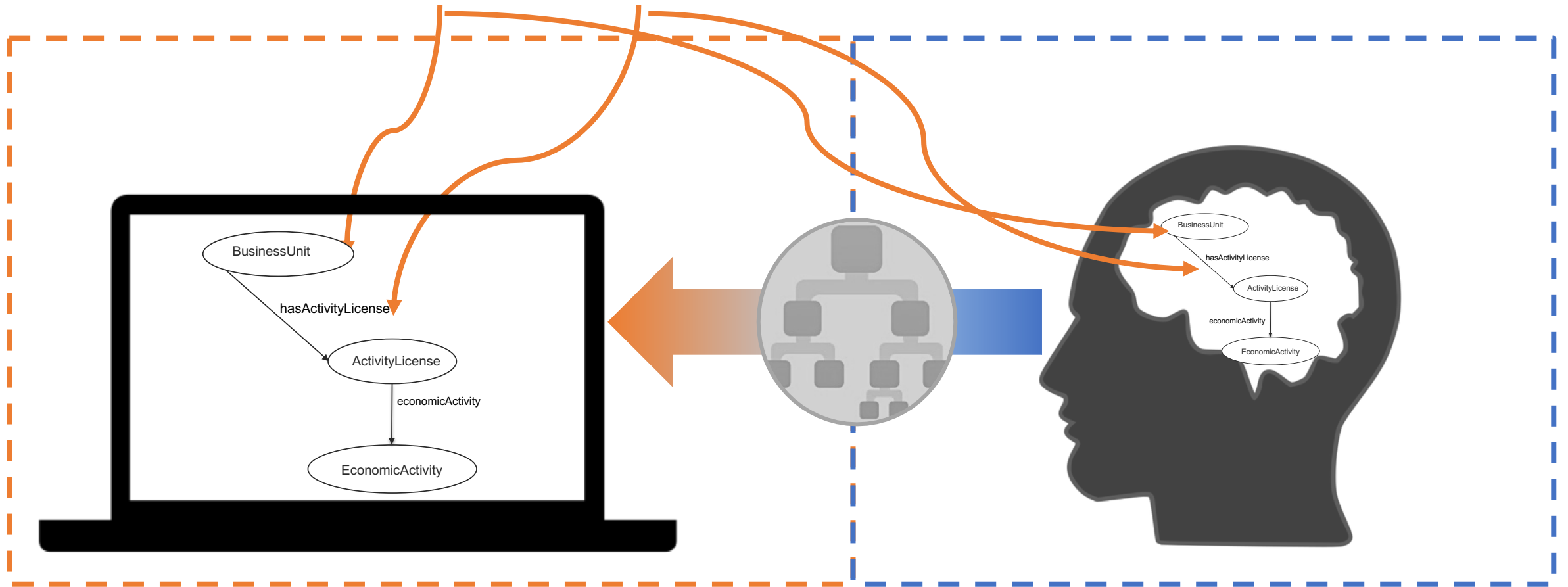
In W3C recommendations (languages) such as RDF Schema or the Web Ontology Language (OWL)

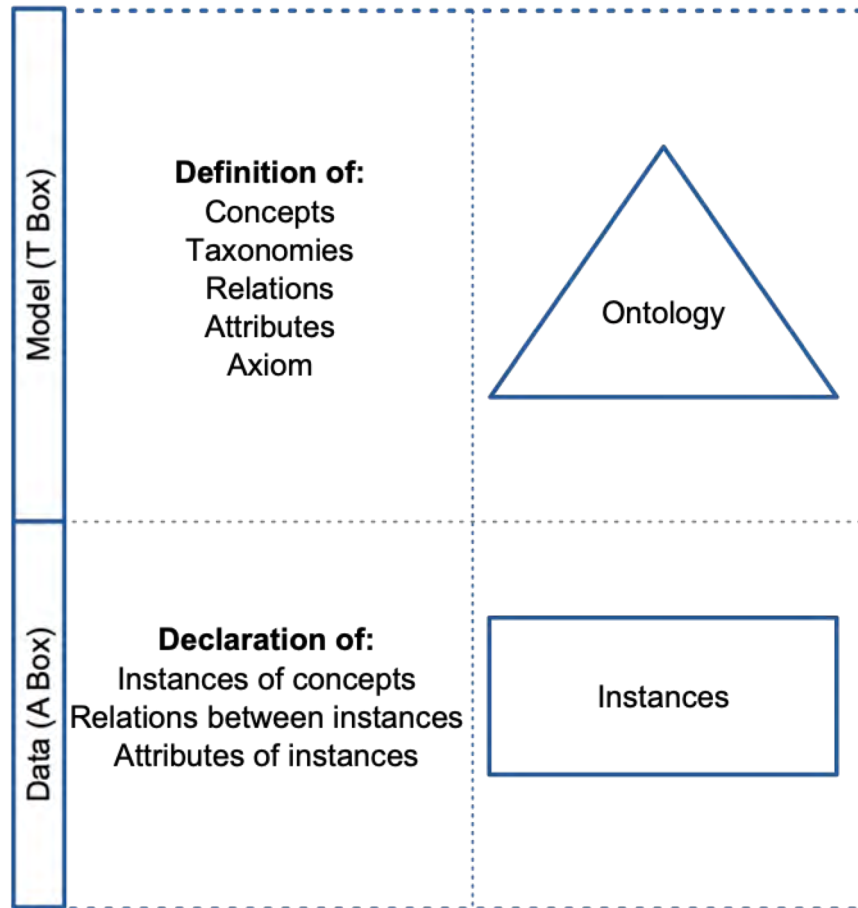


Slide taken from Oscar Corcho

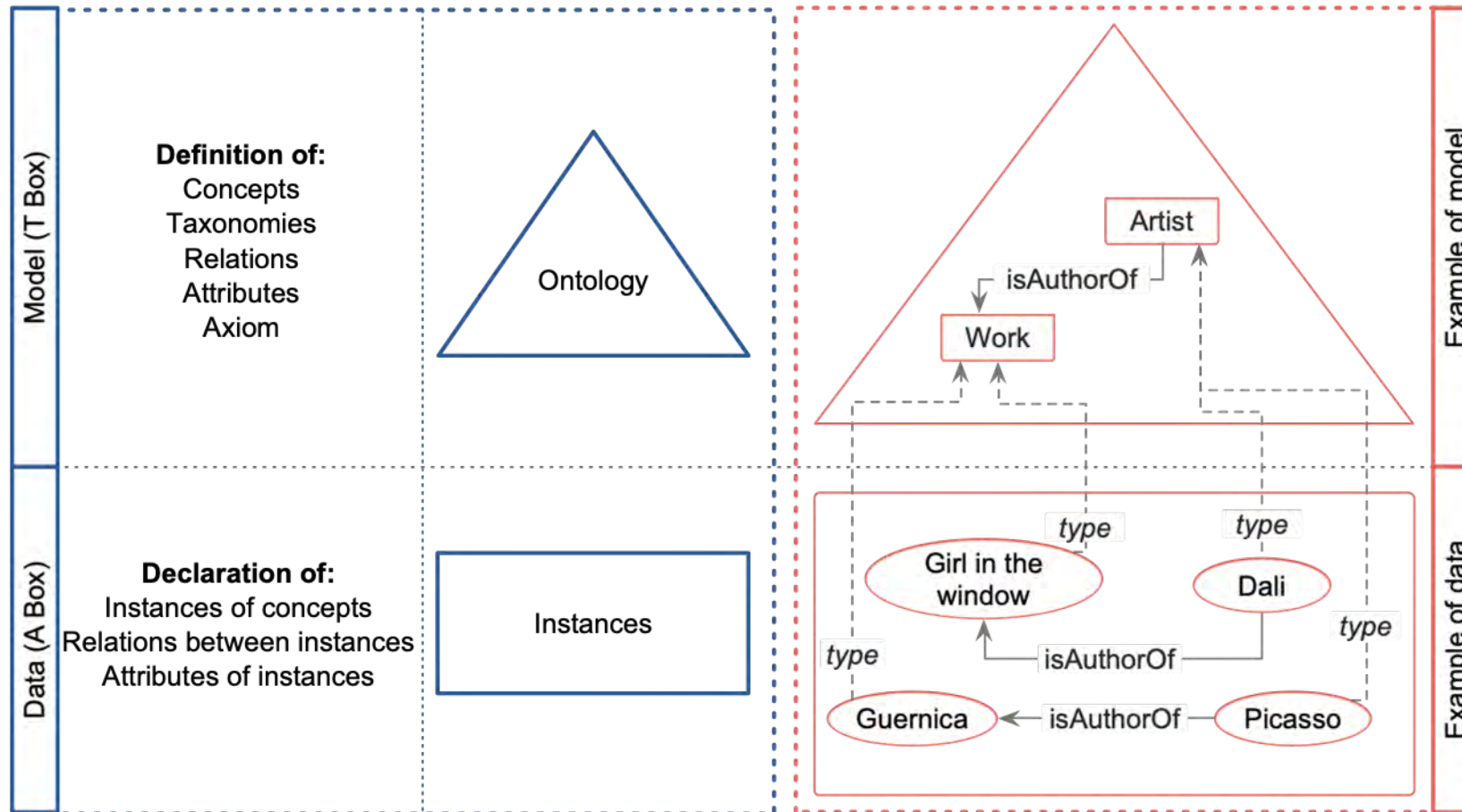
<https://www.w3.org/standards/semanticweb/ontology>

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

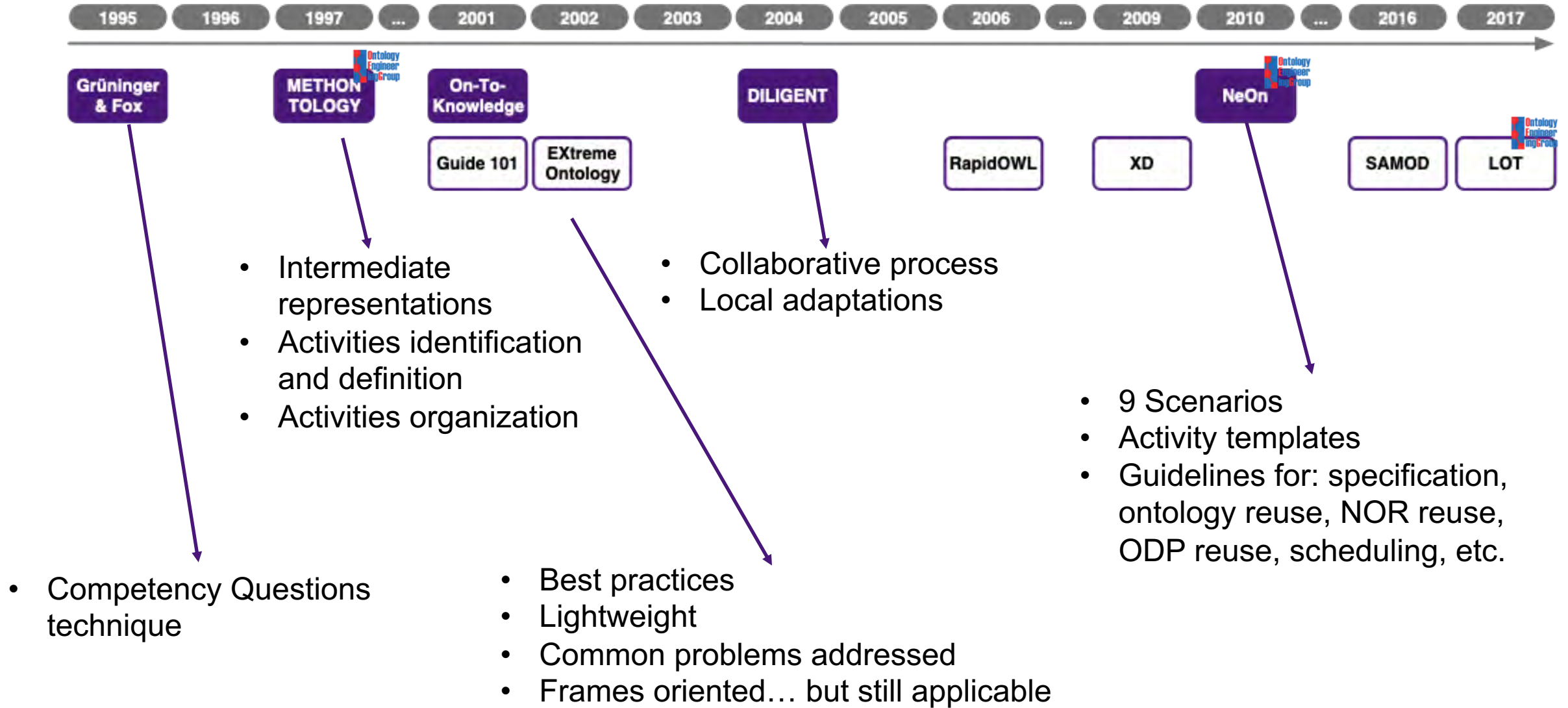




- A knowledge base is divided into:
 - **Tbox** (*Terminological KB*): set of axioms that define the domain:
 - $\text{Artist} \sqsubseteq \text{Person}$
 - $\text{Person} \equiv \text{Man} \sqcup \text{Woman}$
 - **Abox** (*Assertional KB*): set of axioms that describe a situation:
 - Picasso: Man
 - isAuthorOf (Picasso, Guernica)

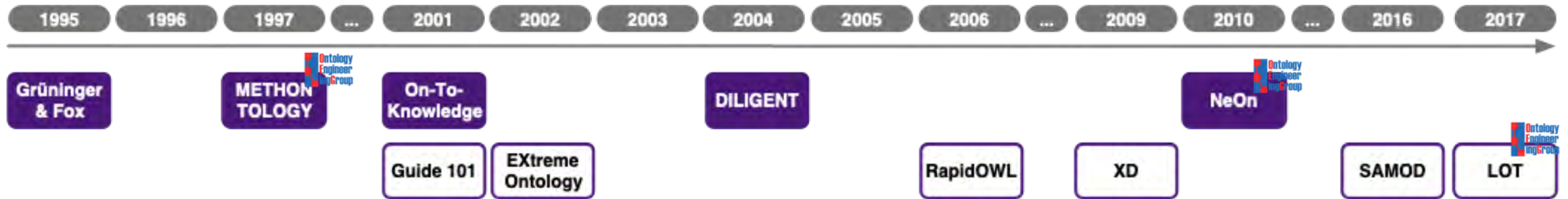


Some Ontology Development Methodologies



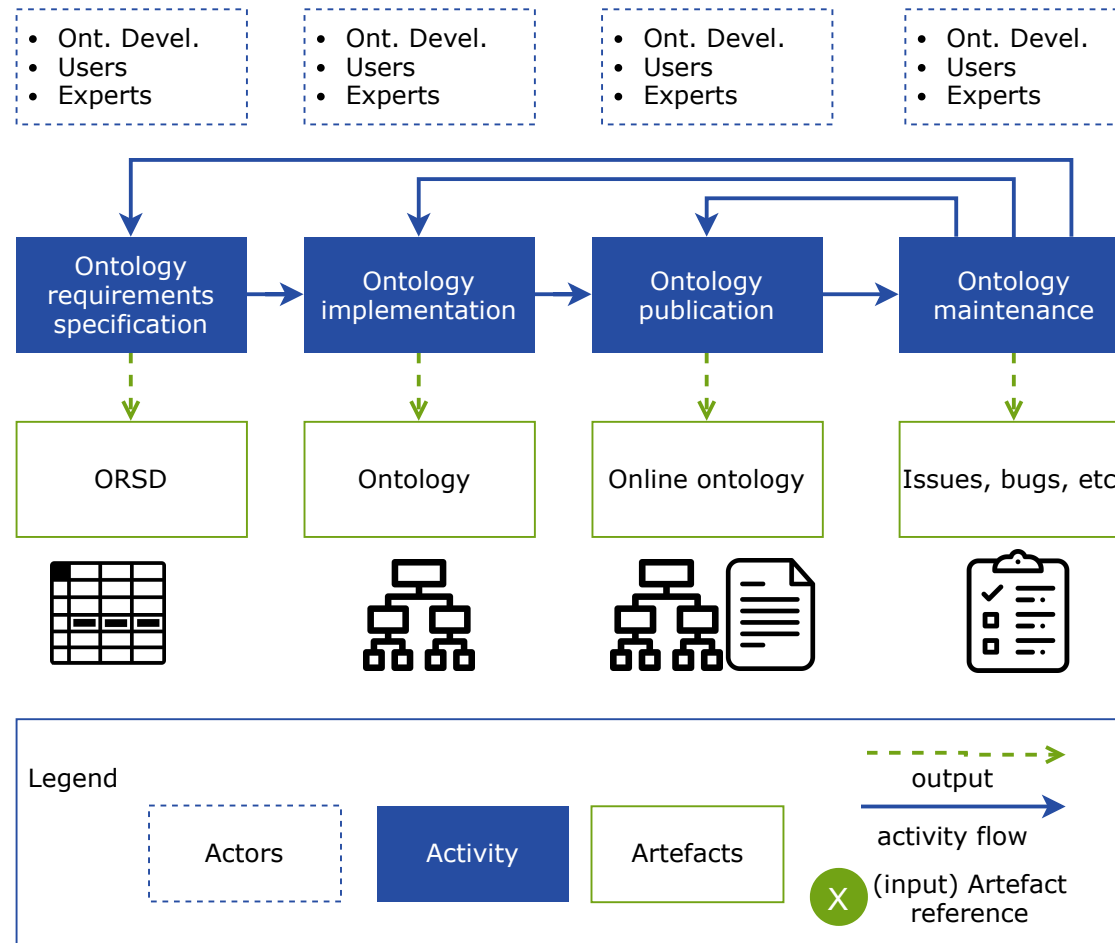
O. Development Methodologies

Ontology Development Lightweight Approaches



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

Ontology development process overview



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

Ontology requirements

- Ont. Devel.
- Users
- Experts

Unofficial Draft

TABLE OF CONTENTS

1. Introduction
 - 1.1 Vision
 - 1.2 How to get started
2. Terminology
3. Concepts & Building Blocks
 - 3.1 WoT Interface
 - 3.1.1 Resource Model and URIs
 - 3.1.2 Protocol Bindings
 - 3.1.3 Security Mechanisms
 - 3.1.3.1 Simple Request Authorization and Caller Authentication
 - 3.1.3.1.1 Communications via HTTP
 - 3.1.3.1.2 Communications via CoAP
 - 3.1.3.2 Advanced Request Authorization and Caller Authentication
 - 3.1.3.3 Message Authentication and Encryption

3.2 Thing Description

The WoT Thing Description (TD) provides the semantic metadata of a Thing as well as a functional description of its WoT Interface. For this, it relies on the Resource Description Framework (RDF) [rdf11-concepts] as an underlying data model. For now, [JSON-LD] is used as the default TD serialization format. The WoT IG defined a minimal vocabulary to express the capabilities of a Thing in terms of different interaction patterns: *Properties, Actions, and Events*. In addition, the TD provides metadata for the different communication bindings (e.g., HTTP, CoAP, etc.), mediaTypes (e.g., "application/json", "application/exi", etc.), and security (etc.). Fig. 3 Concepts of the Thing Description (TD) gives an overview of TD.

Ontology requirements

- Ont. Devel.
- Users
- Experts

Unofficial Draft

TABLE OF CONTENTS

1. Introduction
 - 1.1 Vision
 - 1.2 How to get started
2. Terminology
3. Concepts & Building Blocks
 - 3.1 WoT Interface
 - 3.1.1 Resource Model and URIs
 - 3.1.2 Protocol Bindings
 - 3.1.3 Security Mechanisms
 - 3.1.3.1 Simple Request Authorization and Caller Authentication
 - 3.1.3.1.1 Communications via HTTP
 - 3.1.3.1.2 Communications via CoAP
 - 3.1.3.2 Advanced Request Authorization and Caller Authentication
 - 3.1.3.3 Message Authentication and Encryption

3.2 Thing Description

The WoT Thing Description (TD) provides the semantic metadata of a Thing as well as a functional description of its WoT Interface. For this, it relies on the Resource Description Framework (RDF) [rdf11-concepts] as an underlying data model. For now, [JSON-LD] is used as the default TD serialization format. The WoT IG defined a minimal vocabulary to express the capabilities of a Thing in terms of different interaction patterns: *Properties, Actions, and Events*. In addition, the TD provides metadata for the different communication bindings (e.g., HTTP, CoAP, etc.), mediaTypes (e.g., "application/json", "application/exi", etc.), and security (etc.). Fig. 3 Concepts of the Thing Description (TD) gives an overview of TD.

EXAMPLE 3: More Capabilities

```

{
  "@context": [
    "http://w3c.github.io/wot/w3c-wot-td-context.jsonld",
    { "actuator": "http://example.org/actuator#" }
  ],
  "@type": "Thing",
  "name": "MyLEDThing",
  "base": "coap://myled.example.com:5683/",
  "security": {
    "cat": "token:jwt",
    "alg": "HS256",
    "as": "https://authority-issuing.example.org"
  },
  "interactions": [
    {
      "@type": ["Property", "actuator:onOffStatus"],
      "name": "status",
      "outputData": { "valueType": { "type": "boolean" } },
      "writable": true,
      "links": [
        { "href": "pwr", "mediaType": "application/exi" }
      ]
    },
    {
      "href": "http://mytemp.example.com:8080/status",
      "mediaType": "application/json"
    }
  ]
},
{
  "@type": ["Action", "actuator:fadeIn"],
  "name": "fadeIn",
  "inputData": {
    "valueType": { "type": "integer" },
    "actuator:unit": "actuator:ms"
  },
  "links": [
    { "href": "in", "mediaType": "application/exi" }
  ]
},
{
  "href": "http://mytemp.example.com:8080/in",

```

Use case specification

↓

Use cases

↓

- Ont. Devel.
- Users
- Experts

↓

Data exchange identification

↓

Data documentation & examples

Purpose and scope identification

↓

Ontology and

Towards a formal model of thing descriptions

This message: [Message body] [Respond] [More options]

Related messages: [Next message] [Previous message]

From: Dave Haggatt <dhr@w3.org>
 Date: Wed, 7 Dec 2016 18:20:25 +0000
 Message-Id: <D7140C17-4A50-48C3-92D3-ABED1399060F@w3.org>
 To: Public Web of Things IG <public-wot-ig@w3.org>

In today's Web of things Interest Group call, I was asked to provide a formal model of the RDF graphs for thing description. The question is what formalism to codify it in. One possibility could be the Shapes Constraint Language (SHACL), see: <http://www.w3.org/TR/shacl/> against a set of conditions. This could be used for validating a thing description against the following "grammar", for validating data, and for validating service compositions to check that the components are compatible. What other formalisms are there? The following are based upon requirements derived from a broad range of use cases.

Each thing must have a thing description.
 A thing description is a graph of RDF triples rooted in a given thing.
 A thing description must have URI with which to access the description.
 A thing may have meta-data, i.e. a set of predicate/value pairs.
 A thing may have zero or more properties, actions and events.
 Each property, action and event must have a string literal as its name.
 A property must have a data type.
 A property may have a default value.
 A property may be writable.
 A property may be required.
 Each property may itself have properties.
 Each property, action and event may have metadata.
 Core data types are null, boolean, integer, number, string, vector, thing, enum and union.
 A property may be a stream.
 A property may be a collection.
 A collection is either ordered or unordered, but not both.
 An enum is an unordered set of string literals.
 A union is an unordered set of data types.
 A vector is a set of items, where each item has a string literal for its name, and a non-negative integer for its index.
 A property may have constraints, which depend on its data type.
 An integer or number may have a min and a max value.
 A collection may have a min and a max length.
 Each action must define a request.
 Each action may define a response.
 A request may expect a sequence of zero, one or more responses.
 Each request and response must have a data type.
 Each event must have a data type.
 There are predefined events for signalling updates and life cycle changes.
 Metadata includes comments and communication metadata.
 A comment is a string literal and may be annotated with its human language.
 A thing may be associated with a service.
 A service provides a means to notify updates to properties and metadata.
 A service provides a means to signal events, action requests and responses.
 A service URI may contain named variables.
 A property may be a sink or source but not a combination of these.
 A sink is a stream of samples that applications can generate.
 A source is a stream of samples that applications can observe.
 A stream may have a sampling rate.
 A stream may have a latency.
 A stream may carry date stamps.

Functional Ontology Requirements formalization (optional)

↓

Test suite

<https://lists.w3.org/Archives/Public/public-wot-ig/2016Dec/0016.html>

- Following METHONTOLOGY

Confluence Spaces People Create ...

Concept list

Optional: language tags, English by default

Concept	Other names/ids	Description
Building		Building where the behavior occurs
Occupant		Occupants of the buildings
Behavior		Behaviors of the occupants
Space		Internal space of the building
Meeting		Meeting information if the space is communal
System		The system the occupant interact with (Windows, HVAC's, etc.)

Confluence Spaces People Create ...

Concept	Attribute	Description	Value type expected (integer, boolean, float, list of specific values, etc.)	Max cardinality	Ordering Needed (applicable for concepts with Max cardinality over 1)	Unit of measure (applicable only for "measure" data types)	Sensitive data (if applicable, e.g. personal data that need to be anonymized in BIF)	Timezone (applicable only for datetime data types)	Related standard (if applicable)
Space	description	Description of the space	String	1					obXML
Space	maxNumberOccupants	Maximum number of occupants	Integer	1					obXML
Space	minNumberOccupants	Minimum number of occupants	Integer	1					obXML
Meeting	meetingDuration	Duration of meeting	Integer	1		seconds			obXML

Confluence Spaces People Create ...

Relations: relations from objects/entities to objects/entities

Concept	Relation	Description	Target object (should appear in the "Concept list")	Max cardinality	Ordering Needed (applicable for concepts with Max cardinality over 1)	(ignore if not sure or not applicable) Other characteristics: symmetric, transitive, has some special behaviour or meaning? etc.	Needed by	standards
Building	hasSpace		Space					
Space	meeting		Meeting					
Space	hasSystem		System					
Space	usedBy		Occupant					

- Following Competency Questions technique

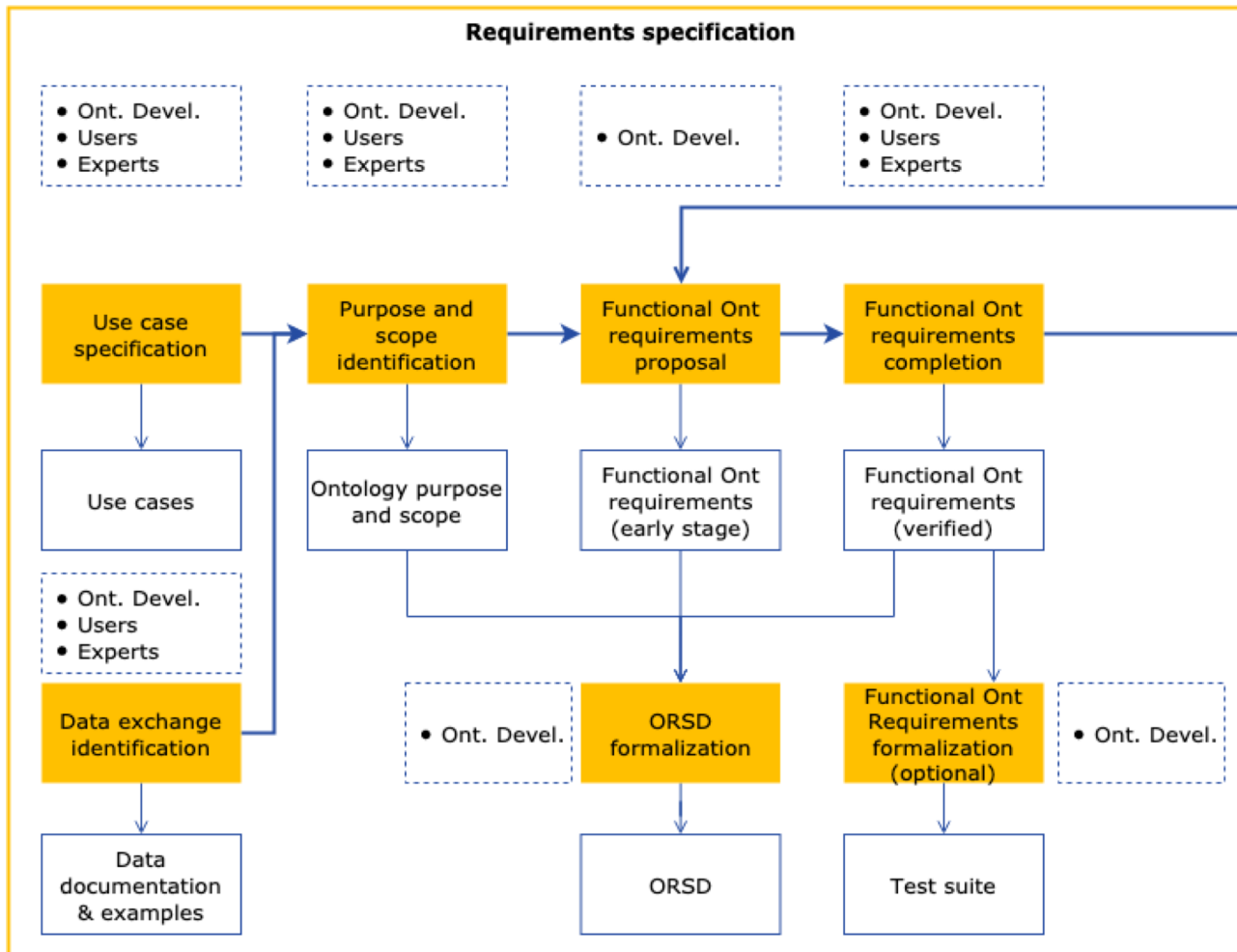
Scope: to support the registration, discovery and search of devices in IoT infrastruc

Requirement						
Identifier (domain+id)	Sprint	Competency Question / Natural language sentence (fact)	Answer	Status (Proposed, Accepted, Rejected, Deprecated)	Comments	Extracted from (provenance)
WoT1	1	What is a thing in the web thing context?	The abstract concept of a physical entity that can either be a real-world arteact, such as a device, or a virtual entity that represents physicality, such as a room or group of devices	A	In vicinity the things can be sensors and devices but also added-value services.	http://w3c.github.io/wot/current-practices/wot-practices.html
WoT2		What is a servient?	The addressable application endpoint of a Thing that makes it interactive by providing a WoT Interface and means to execute application logic			http://w3c.github.io/wot/current-practices/wot-practices.html
WoT3		What is a repository?	A registry for Thing Descriptions that provides a Web interface to register Thing Descriptions and look them up, for intance using SPARQL queries			http://w3c.github.io/wot/current-practices/wot-practices.html
WoT4	-	What is a thing description?	An RDF document (currently serialized in JSON-LD by default) that contains semantic and functional descriptions of a Thing	R	covered in core	http://w3c.github.io/wot/current-practices/wot-practices.html
WoT5	-	What is a WoT interface?	Def 1: Resource-oriented Web interface (often called "Web API") that allows access to servients over the network using different Protocol Bindings. Def 2: A WoT interface is also a web API that follows the recommendations of the WoT Interest Group.	R	https://github.com/mariapoveda/vicinity-ontology-wot/issues/5 Obsolete: Añadir a la descripción de WoT Interface. Para que un api web sea una WoT interface tiene que ser describle por un TD. It has to have at least one property.	http://w3c.github.io/wot/current-practices/wot-practices.html
WoT6		Things in the WoT architecture are represented by servients.				http://w3c.github.io/wot/current-practices/wot-practices.html
WoT7		Servient can represent virtual things as well.			How are the hosts represented? URIs? entities? strings?	http://w3c.github.io/wot/current-practices/wot-practices.html
WoT8		Servients are hosted anywhere, a smartphone, local gateway, or the cloud.				http://w3c.github.io/wot/current-practices/wot-practices.html
WoT9		Servient communicate to each other through a WoT interface				http://w3c.github.io/wot/current-practices/wot-practices.html
WoT10		Servient can have cliente or server roles or both.			Is this fixed or can change? Needs to be tracked? Need to be attached to time intervals?	http://w3c.github.io/wot/current-practices/wot-practices.html
WoT11	-	Each thing is described by WoT Thing Descriptions.	A thing might be described by more than one thing description	R	https://github.com/mariapoveda/vicinity-ontology-wot/issues/1 (covered in core)	http://w3c.github.io/wot/current-practices/wot-practices.html
WoT12		Thing Descriptions can be registered in Td repositories.			Which attributes or waht information should be attached to tha TD repository?	http://w3c.github.io/wot/current-practices/wot-practices.html
WoT13	-	TD repositories can be queried using for example SPARQL		R		http://w3c.github.io/wot/current-practices/wot-practices.html
WoT14	-	A WoT interface provides Web resources that implement the interaction patterns of Properties, Actions, and Events. Each web resource acts as a mediator/proxy (web interface) between the Thing and their clients.		R	https://github.com/mariapoveda/vicinity-ontology-wot/issues/5	http://w3c.github.io/wot/current-practices/wot-practices.html

Shared in online spreadsheets



Grüniger, M. and Fox, M. S. (1995). Methodology for the design and evaluation of ontologies. In IJCAI'95, Workshop on Basic Ontological Issues in Knowledge Sharing



Ontology Requirements Specification Document	
Purpose	
Scope	
Implementation Language (optional)	
Intended End-Users (optional)	
Intended Uses	
Ontology Requirements	
c. Non-Functional Requirements	
d. Functional Requirements: Lists or tables of requirements written as Competency Questions and sentences	
Pre-Glossary of Terms (optional)	
d. Terms from Competency Questions	
e. Terms from Answers	
f. Objects	



CORAL corpus provides examples and requirements patterns

<http://coralcorpus.linkeddata.es/>

Ontology Requirements Specification Document Template	
1 Purpose	<i>The general goal of the BIMERR ontology and data model is to facilitate data sharing and interoperability among the BIMERR components through the BIMERR Interoperability Framework.</i>
2 Scope	<i>The scope of the BIMERR ontologies is limited to the data shared through the BIF and external data sources needed in the energy efficiency domain and related domains like: KPIs, project management, weather, occupancy behavior, information objects, building geometry, building elements, materials and renovation measurements.</i>
3 Implementation Language	<i>Ontology Web Language</i>
4 Intended End-Users	<i>BIMERR components and application developers</i> <i>BIMERR end-users and stakeholders</i>
5 Intended Uses	<i>Data model generation</i> <i>External data sources integration</i>
6 Ontology Requirements	
a. Non-Functional Requirements	<i>Annotated in English</i> <i>Linked to standards when possible</i> <i>Open license</i> <i>Modular</i> <i>Online availability</i>
b. Functional Requirements: Groups of Competency Questions	<i>This section is provided for each specific domain in the confluence pages.</i>

ORSD template

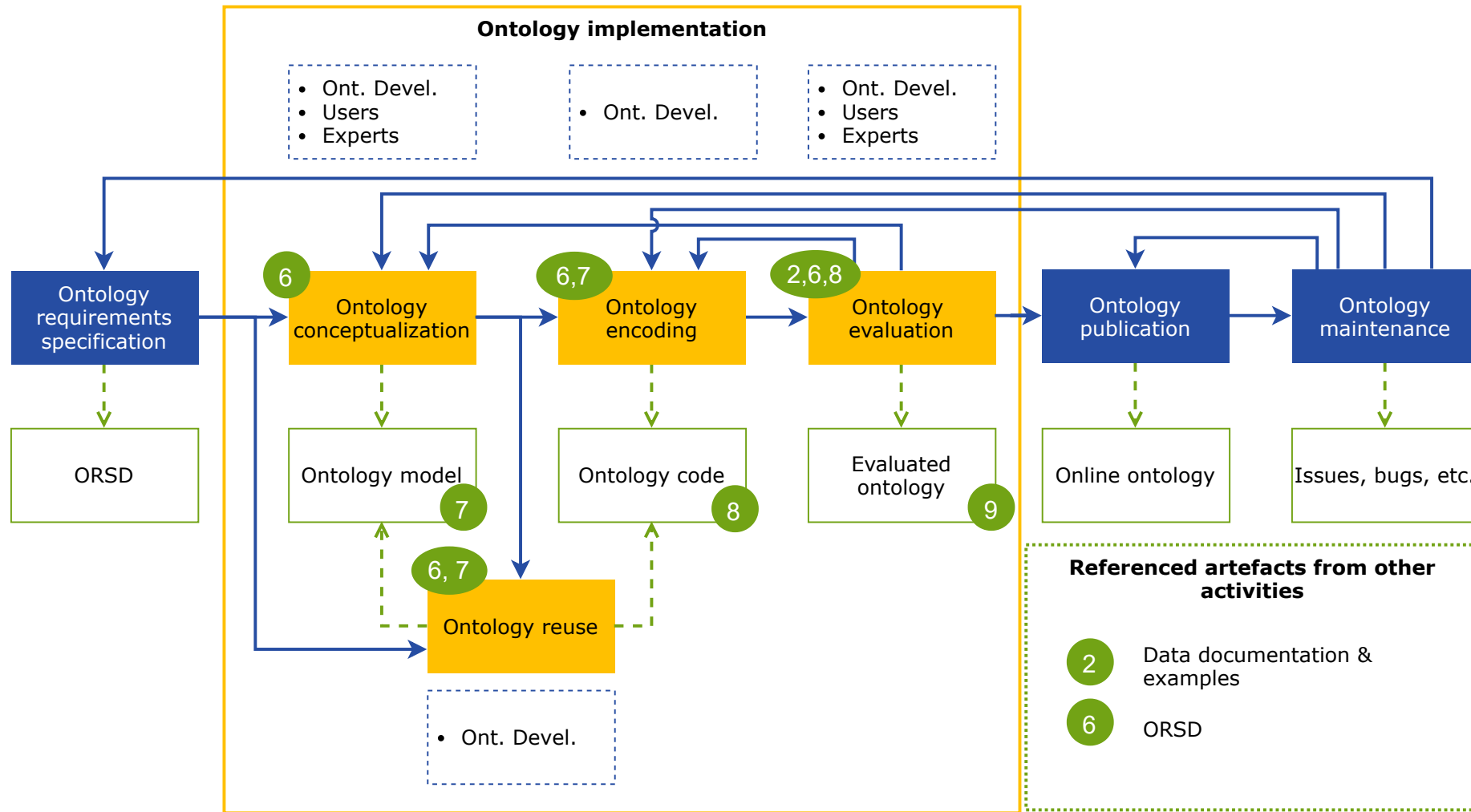
<https://github.com/oeg-upm/LOT-resources/tree/master/ORSD>

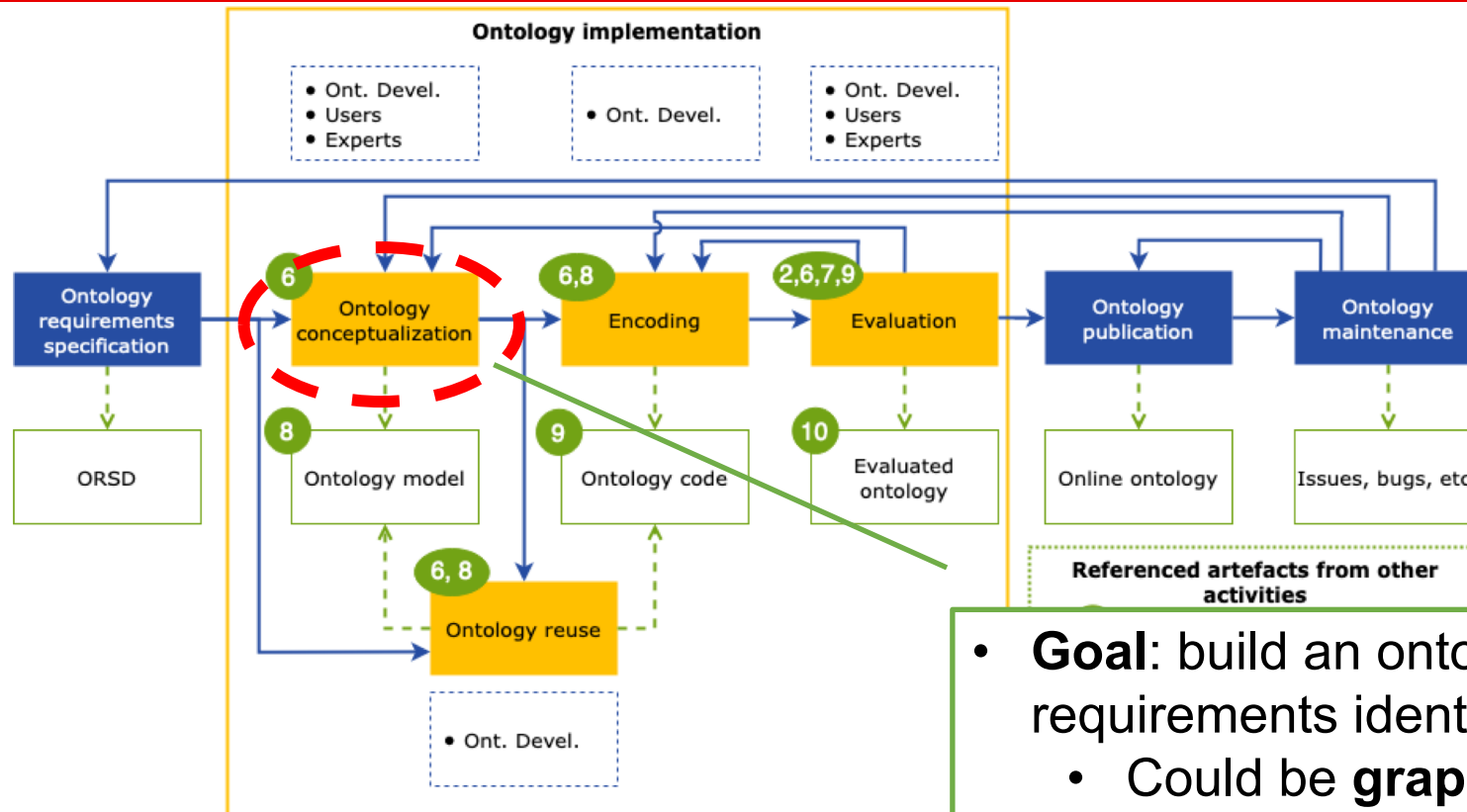
Concept list
Optional: language tags, English by default

Concept	Other names/ids	Description
Building		Building where the behavior occurs
Occupant		Occupants of the buildings
Behavior		Behaviors of the occupants
Space		Internal space of the building
Meeting		Meeting information if the space is communal
System		The system the occupant interact with (Windows, HVAC's, etc)

Relations: relations from objects/entities to objects/entities

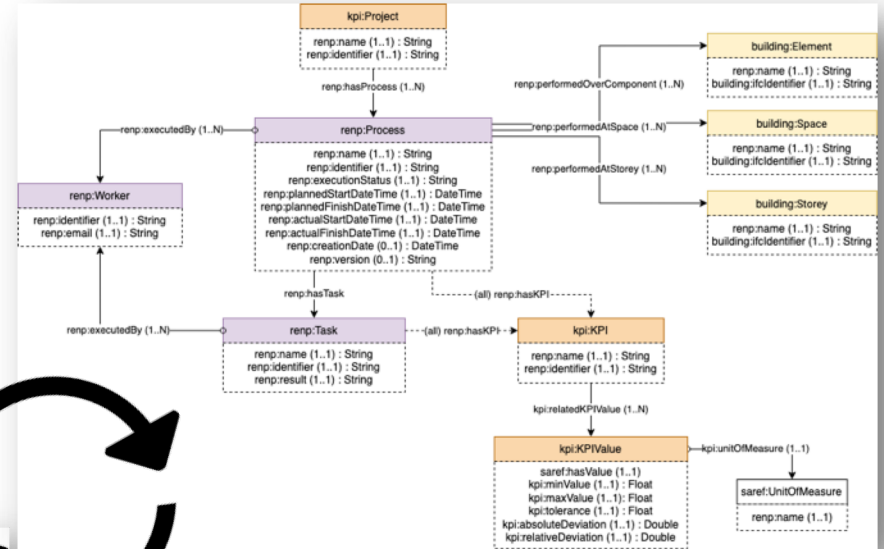
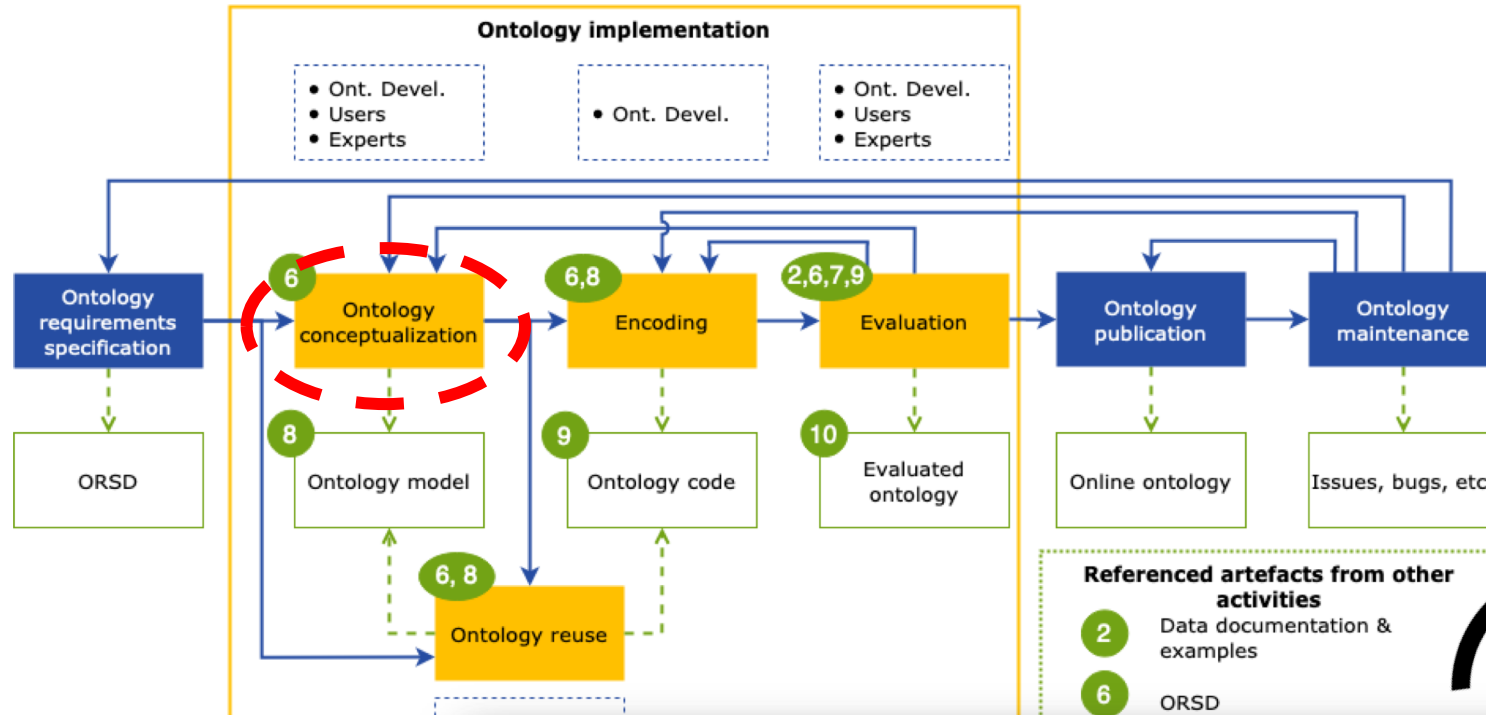
Concept	Relation	Description	Target object (should appear in the "Concept list")	Max cardinality	Ordering Needed (applicable for concepts with Max cardinality over 1)	(ignore if not sure or not applicable) Other characteristics: symmetric, transitive, has some special behaviour or meaning? etc.
Building	hasSpace		Space			
Space	meeting		Meeting			
Space	hasSystem		System			
Space	usedBy		Occupant			



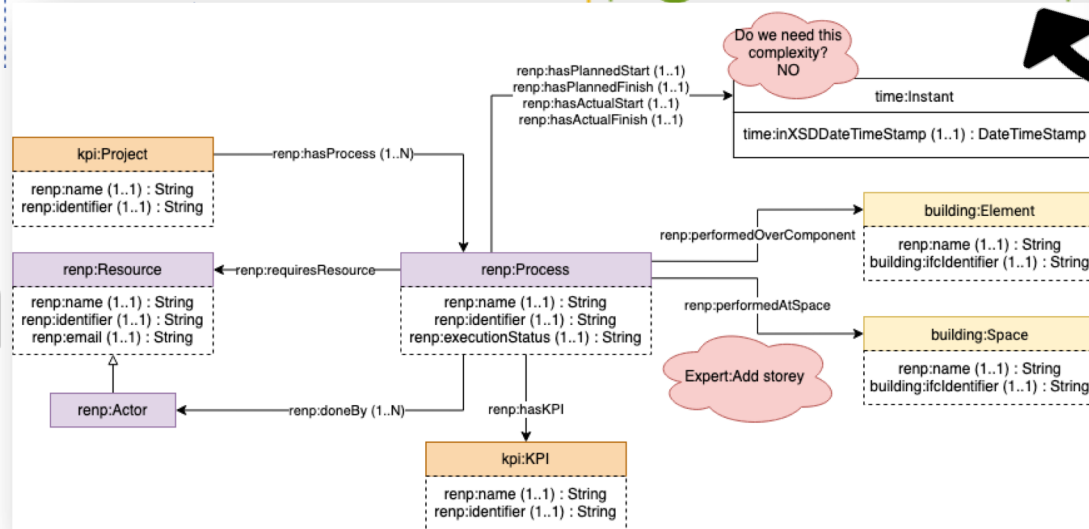


- **Goal:** build an ontology model from the ontological requirements identified.
 - Could be **graphical** or described in a formal system
- You can use
 - Blackboard
 - Pen & pencil
 - Drawing tools (diagrams.net, visio, yEd...)

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.

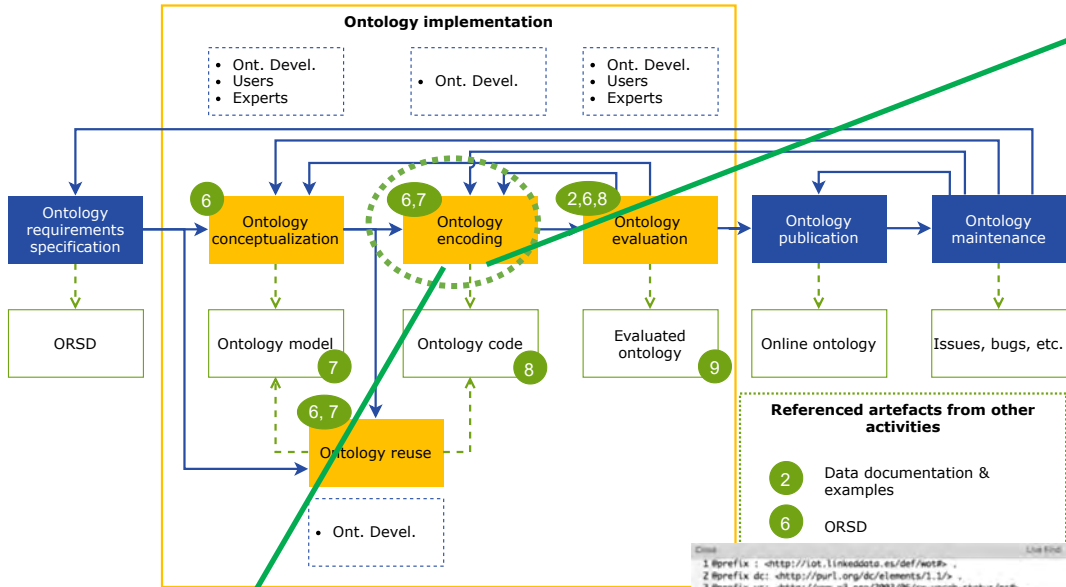


Domain Expert

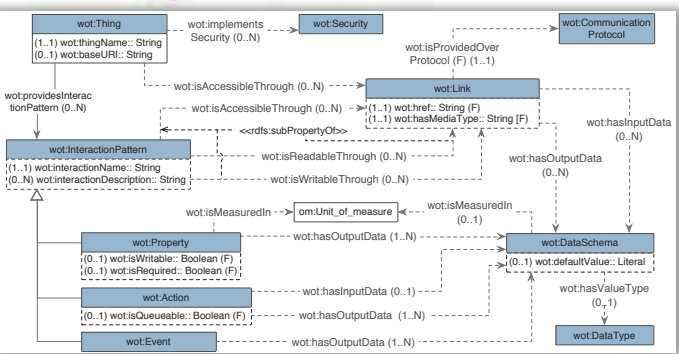
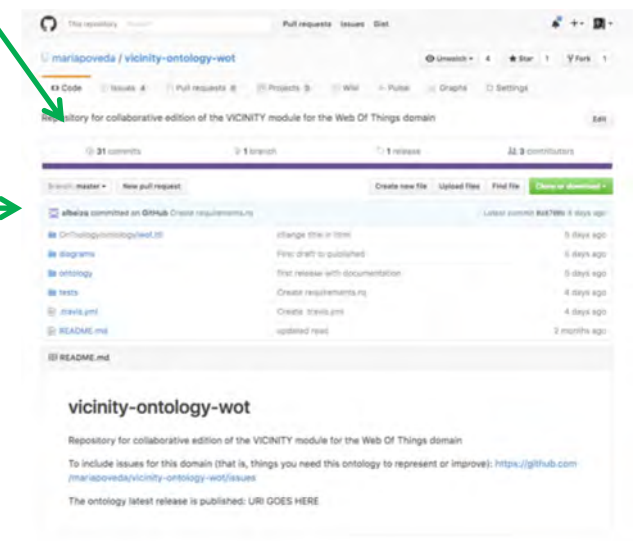


Ontology Engineer

Openly managed in GitHub

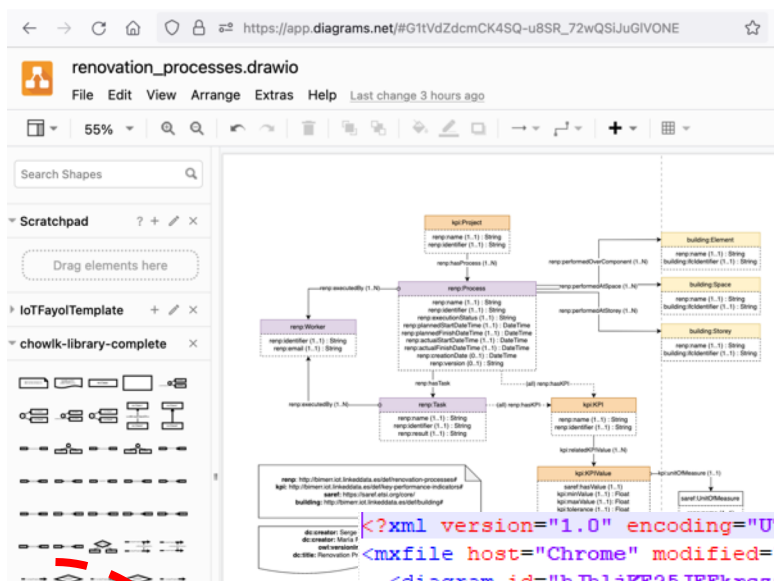


```
1 #prefix : <http://lot.linkeddata.es/def/wot#> .
2 #prefix dc: <http://purl.org/dc/elements/1.1/> .
3 #prefix vs: <http://www.w3.org/2003/06/iv-vocab-status/ns#> .
4 #prefix geo: <http://www.w3.org/2003/01/geo/terms#> .
5 #prefix owl: <http://www.w3.org/2002/07/owl#> .
6 #prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
7 #prefix xml: <http://xmlns.com/xml/1/> .
8 #prefix xml: <http://www.w3.org/2001/03/xml-ns#> .
9 #prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
10 #prefix foaf: <http://xmlns.com/foaf/0.1/> .
11 #prefix prov: <http://www.w3.org/ns/prov#> .
12 #prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
13 #prefix vann: <http://purl.org/vocab/vann/> .
14 #prefix saref: <https://w3id.org/saref#> .
15 #prefix dcterm: <http://purl.org/dc/terms/> .
16 #base <http://lot.linkeddata.es/def/wot#> .
17
18 <http://lot.linkeddata.es/def/wot#> rdfs:type owl:Ontology ;
19   dcterm:creator <http://purl.org/net/mariapoveda#> ;
20   dc:title "Vicinity model for Web of Things" ;
21   vann:preferredNamespacePrefix "wot" ;
22   dcterm:license <http://purl.org/NET/rdflib/license/cc-by4.0/> ;
23   owl:versionInfo "1.0" ;
24   vann:preferredNamespaceURI "http://lot.linkeddata.es/def/wot" ;
25   rdfs:comment "This ontology aims to model the Web of Things domain according to the w3c Interest Group <http://w3c.github.io/wot/>" ;
26
27   dc:publisher <http://www.oeg-upm.net/> .
28
29 #####
30 # Annotation properties
31 #####
32
33 ## http://www.w3.org/2003/06/iv-vocab-status/ns#term_status
34 vs:term_status rdfs:type owl:AnnotationProperty .
35
36
37 #####
38 # Object Properties
39 #####
40
41 ## http://lot.linkeddata.es/def/wot#describes
42 <http://lot.linkeddata.es/def/wot#describes> rdfs:type owl:ObjectProperty ;
43   owl:inverseOf <http://lot.linkeddata.es/def/wot#isDescribedBy> ;
44   rdfs:label "describesThing" ;
45
```



GitHub repository

<https://github.com/mariapoveda/vicinity-ontology-wot>



Convert into OWL with Chowk



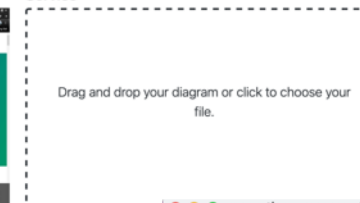
Instructions

1. Download the Chowk template ([complete](#) or [lightweight](#) version).
2. Open diagrams.net (web or desktop)
3. In diagrams.net go to File > Open Library from > Device ...
4. Select the library downloaded.
5. Make your conceptualization using the block that will appear on the side bar.
6. Download the diagram in xml format.
7. Drag and drop your model in the Service dropping area and download your TTL file.

How to use it



Service



Serge Chávez Ferla
Contact email: serge.chavez.ferla@upm.es
Latest revision: July, 2021
Licensed under the [Apache License 2.0](#)

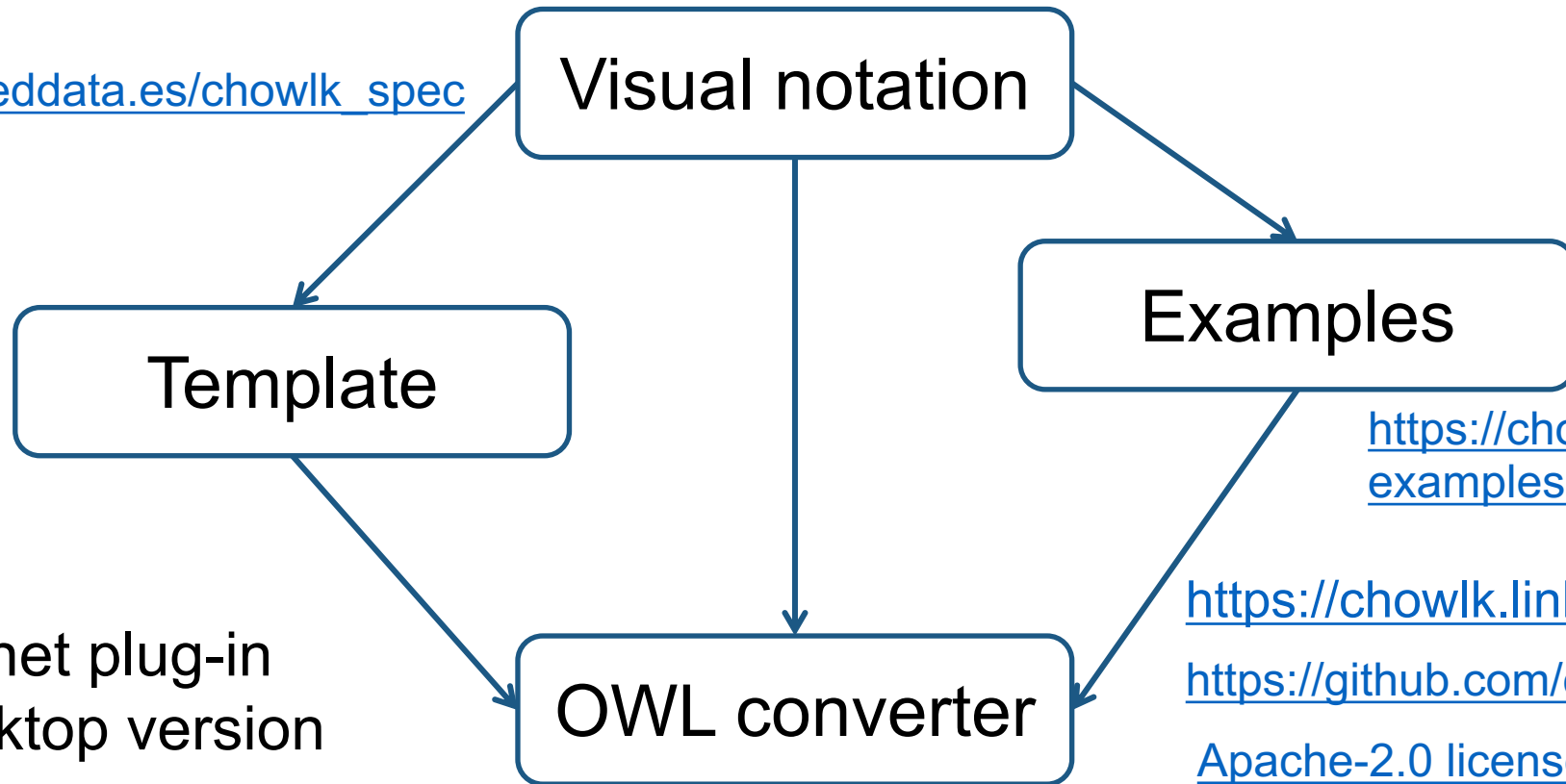
Continue in Protégé (if you want)

Export XML from diagrams.net

```
<?xml version="1.0" encoding="UTF-8"?>
<mxfile host="Chrome" modified="2020-11-16T14:41:32.661Z" agent="
<diagram id="hJhljKE25JFFkrsz6_xK">
<mxGraphModel dx="20674" dy="19061" grid="1" gridSize="1" gui
<root>
<mxCell id="0" />
<mxCell id="1" parent="0" />
<mxCell id="2" style="edgeStyle=orthogonalEdgeStyle;round
<mxGeometry relative="1" as="geometry">
<mxPoint x="-17271" y="-17490" as="targetPoint" />
</mxGeometry>
</mxCell>
<mxCell id="3" value="building:Building" style="rounded=(
<mxGeometry x="-17377" y="-17470.23999999999998" width="2
</mxCell>
<mxCell id="4" value="(all) bot:hasStorey" style="edgeSty
<mxGeometry x="-0.1466" y="2" relative="1" as="geometry
<Array as="points">
<mxPoint x="-17470" y="-17457" />
<mxPoint x="-17470" y="-17403" />
</Array>
<mxPoint as="offset" />
</mxGeometry>
</mxCell>
</mxCell>
```



https://chowlk.linkeddata.es/chowlk_spec



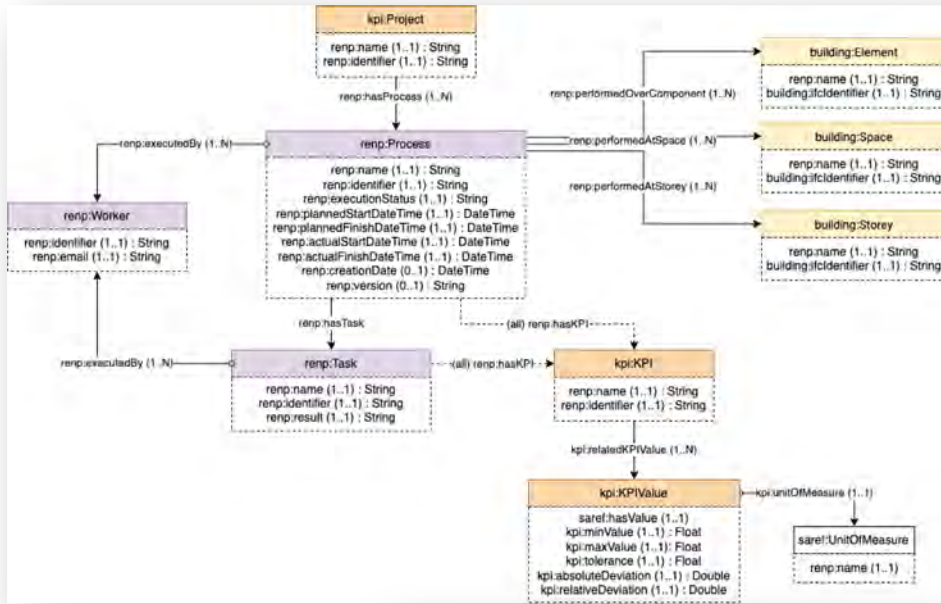
<https://chowlk.linkeddata.es/examples.html>

<https://chowlk.linkeddata.es>

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

[Apache-2.0 license](#)

+ diagrams.net plug-in
Only for desktop version



Class hierarchy (inferred):

- owl:Thing
- Element
- HealthSafetyIssue
- KPI
- KPIValue**
- Process
- Space
- Storey
- Task
- UnitOfMeasure
- Worker

Annotations: KPIValue

- rdfs:label [language: en] KPI value
- rdfs:comment [language: en] Key performance indicator values calculated for a specific renovation project and scenario.

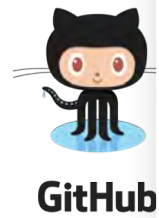
Description: KPIValue

- absoluteDeviation exactly 1 rdfs:Literal
- hasValue exactly 1 rdfs:Literal
- maxValue exactly 1 rdfs:Literal
- minValue exactly 1 rdfs:Literal
- relativeDeviation exactly 1 rdfs:Literal
- tolerance min 1 rdfs:Literal
- unitOfMeasure exactly 1 owl:Thing

oeg-upm / bimerr-renovation-process

Repository for the renovation process ontology.

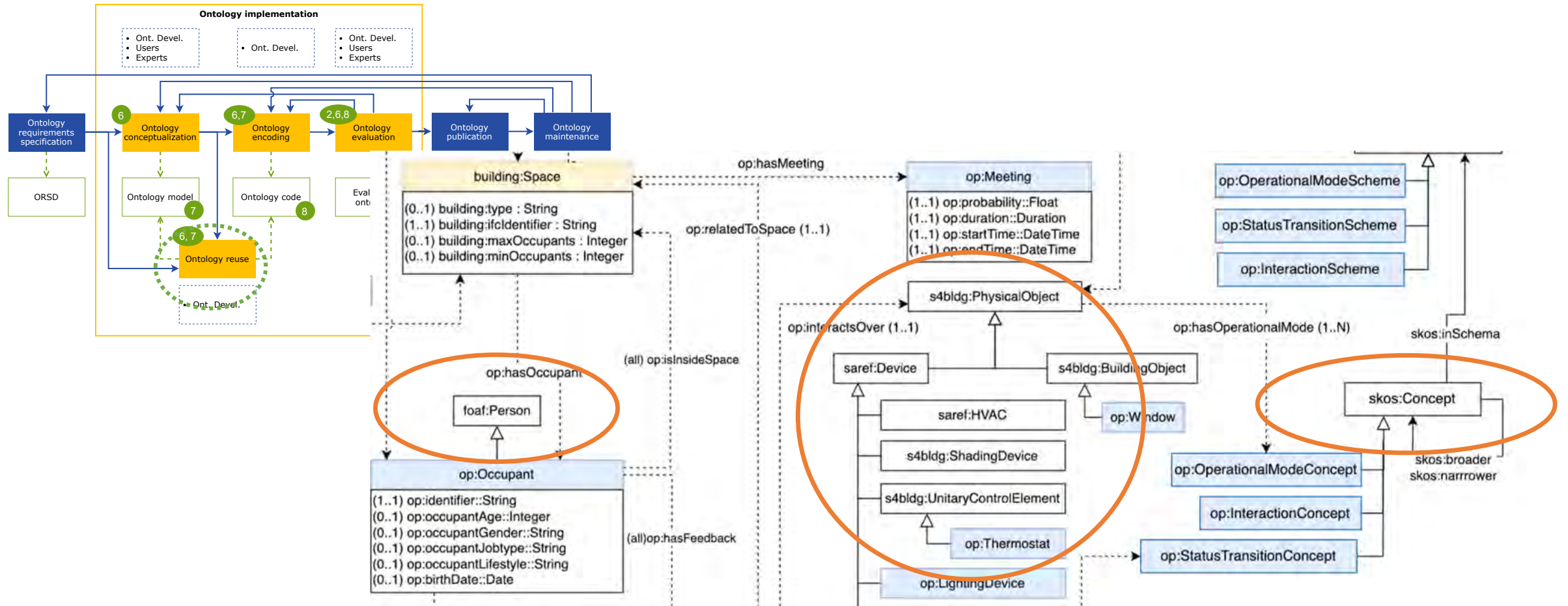
Current version of the ontology model



```
@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#> rdfs:type owl:Ontology ;
<http://purl.org/dc/elements/1.1/creator>
  <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/rdflib/license/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
```



- Reusing knowledge resources



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

<https://lov.linkeddata.es>

- Mission: promote and facilitate the **reuse** of **well documented** vocabularies in the **Linked Data ecosystem**
- Vocabularies registry and index
- Datalift
 - <http://datalift.org/>
- Started at 2011
- Hosted by OEG



A screenshot of the Linked Open Vocabularies (LOV) website. The page has a white background with a teal header. At the top, there are navigation links for "VOCABS", "TERMS", "AGENTS", and "SPARQL/DUMP". Below the header is a teal banner with the text "Linked Open Vocabularies (LOV)". Underneath the banner are several utility buttons: "+ Suggest", "Documentation", "g+ Follow", a search bar, and a user profile icon. The main content area features a large circular bubble chart titled "595 Vocabularies in LOV". The chart shows various vocabularies represented by colored circles of different sizes, with larger circles indicating more prominent vocabularies. Labeled vocabularies include "vann", "foaf", "skos", "dcterms", "dce", "cc", "vs", "schema", "geo", "prov", "gr", "event", "time", "org", "void", "adms", "dctype", "stn", "qb", "acc", "gn", "doap", "vocab", "skos", "foaf", "vann", "dcterms", "dce", "cc", "vs", "schema", "geo", "prov", "gr", "event", "time", "org", "void", "adms", "dctype", "stn", "qb", "acc", "gn", "doap", "vocab". To the right of the bubble chart is a "Latest insertion" section listing recent additions with their names and dates. Below that is a "Latest Updates" section listing recent updates. At the bottom of the main content area is a "Category Tags" section with a grid of tags such as "Methods", "Metadata", "Catalogs", "Support", "Geography", "API", "Society", "Quality", "RDF", "Industry", "Services", "People", "Vocabularies", "Environment", "General & Upper", "Time", "IoT", "Events", "Geometry", "Multimedia", "FRBR", "Biology", "W3C Rec", "SPAR", "Government", "PLM", "Academy", "eBusiness", "Tag", "Travel". The footer of the page is dark grey and contains the "Linked Open Vocabularies" logo, a "DOCUMENTATION" section with links for "About", "API documentation", "Source code", and "Contact", a "PUBLICATION" section with links for "Semantic Web Journal '16", "ERCIM News '14", and "Library Hi Tech '13", and the Open Knowledge Foundation logo with a Creative Commons license icon.

Data Catalog Vocabulary (dcat)

Metadata

URI	http://www.w3.org/ns/dcat
Namespace	http://www.w3.org/ns/dcat#
homepage	http://www.w3.org/TR/vocab-dcat/
Description	DCAT is an RDF vocabulary designed to facilitate interoperability between data catalogs published on the Web @en
Language	<div style="display: flex; gap: 5px;"> <div>Arabic (ar)</div> <div>Greek (el)</div> <div>English (en)</div> <div>Spanish (es)</div> <div>French (fr)</div> <div>Japanese (ja)</div> </div>
Contributor	<div style="display: flex; gap: 5px;"> <div>Richard Cyganiak (http://google.com/+RichardCyganiak)</div> <div>Phil Archer (https://plus.google.com/103670676337547906055)</div> <div>Fadi Maali</div> </div>



Connection with other applications

Statistics

Classes	7
Properties	17
Datatypes	0
Instances	0

Expressivity

RDF RDFS

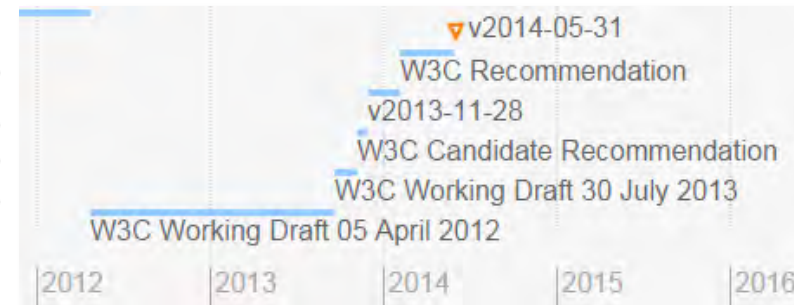
Tags

W3C Rec

Relationships



Versions



VOCABS TERMS AGENTS SPARQL/DUMP

TERMS art work

3120 results

Vocabulary	Count
cc:Work (cc)	0.609
con:office (con)	0.556
dm2e:Work (dm2e)	0.493
doc:Work (doc)	0.493
pext:Art (pext)	0.491
bibo:Book (bibo)	0.473
bf:Work (bf)	0.468
mrel:acp (mrel)	0.390

Filters

Type

- property/class
- property (2250)
- class (870)

Tag

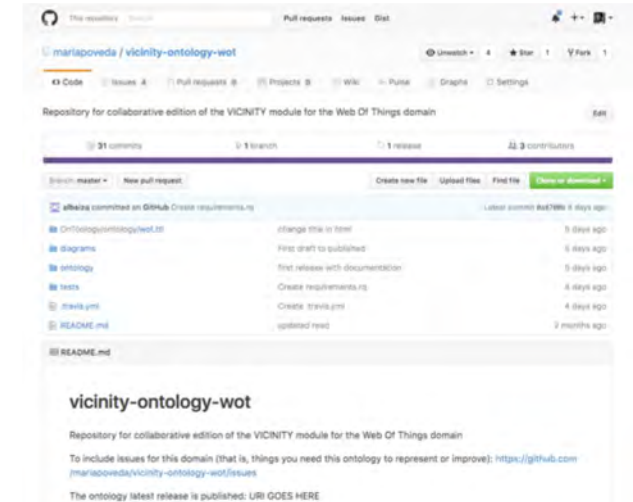
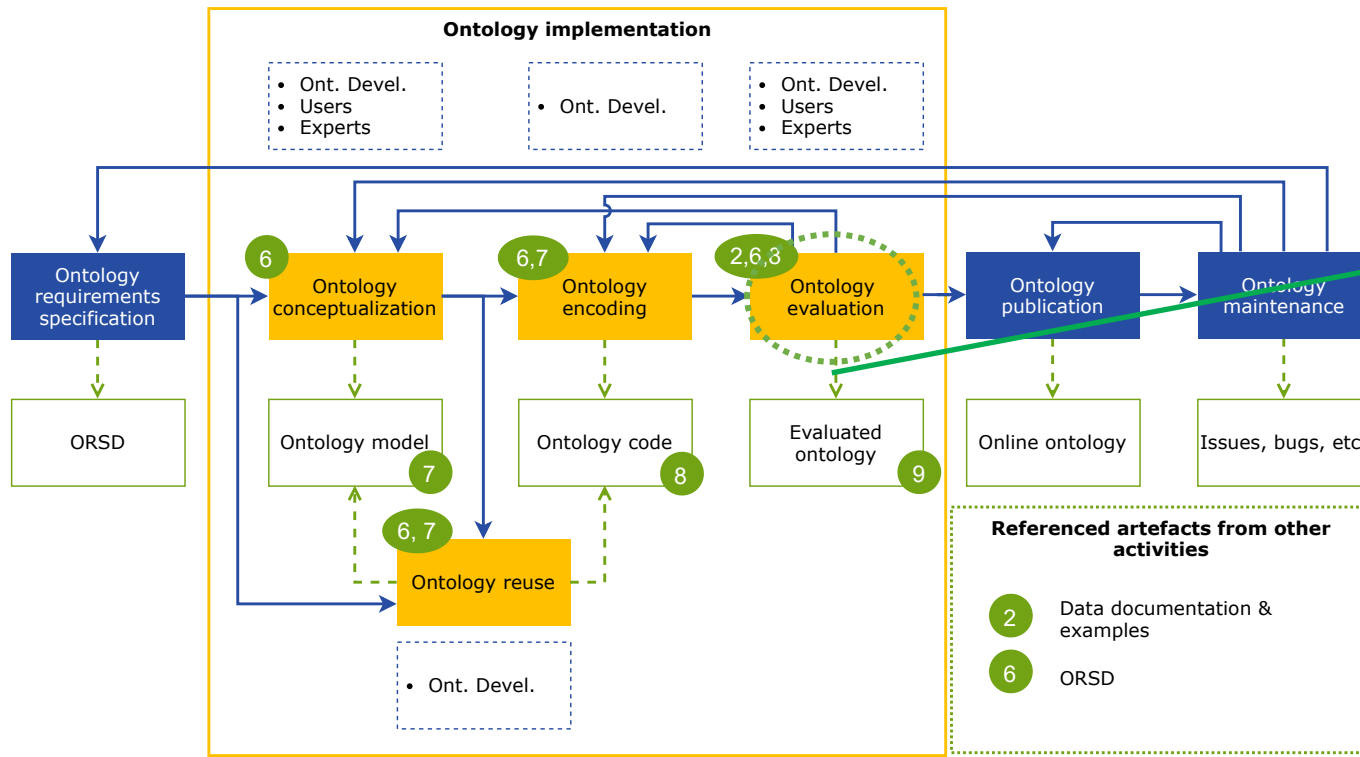
- FRBR (958)
- Catalogs (498)
- Metadata (368)
- Government (234)
- Events (145)
- Society (135)
- General & Upper (110)
- Tag (110)
- Multimedia (87)
- SPAR (79)

Vocabulary

- bf (376)
- rdaw (297)
- rdarel (263)
- ai2a (233)
- rdax (196)

Ranked

- Term appearing in primary and secondary annotations
- Vocabulary popularity in LOV
- Term use in LOD



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/>)



- Implements the **48** detection methods for **33** pitfalls
 - Pitfalls selection
 - Selection by dimensions and aspects
- Web user interface <http://oops.linkeddata.es/>
- Web service <http://oops-ws.oeg-upm.net/>

Ontology Pitfall Scanner!

OOPS! (Ontology Pitfall Scanner!) helps you to detect some of the most common pitfalls appearing when developing ontologies. To try it, enter a URI or paste an OWL code.

URI input
Example: <http://data.semanticweb.org>

OWL code input
If you checked this checkbox, if you paste OWL code, the scanner will check this code for errors.

Pitfall name

Pitfall frequency

Importance level

Results for P04: Creating unconnected ontology elements.

Results for P05: Defining wrong inverse relationships.

Results for P08: Missing annotations.

Results for P11: Missing domain or range in properties.

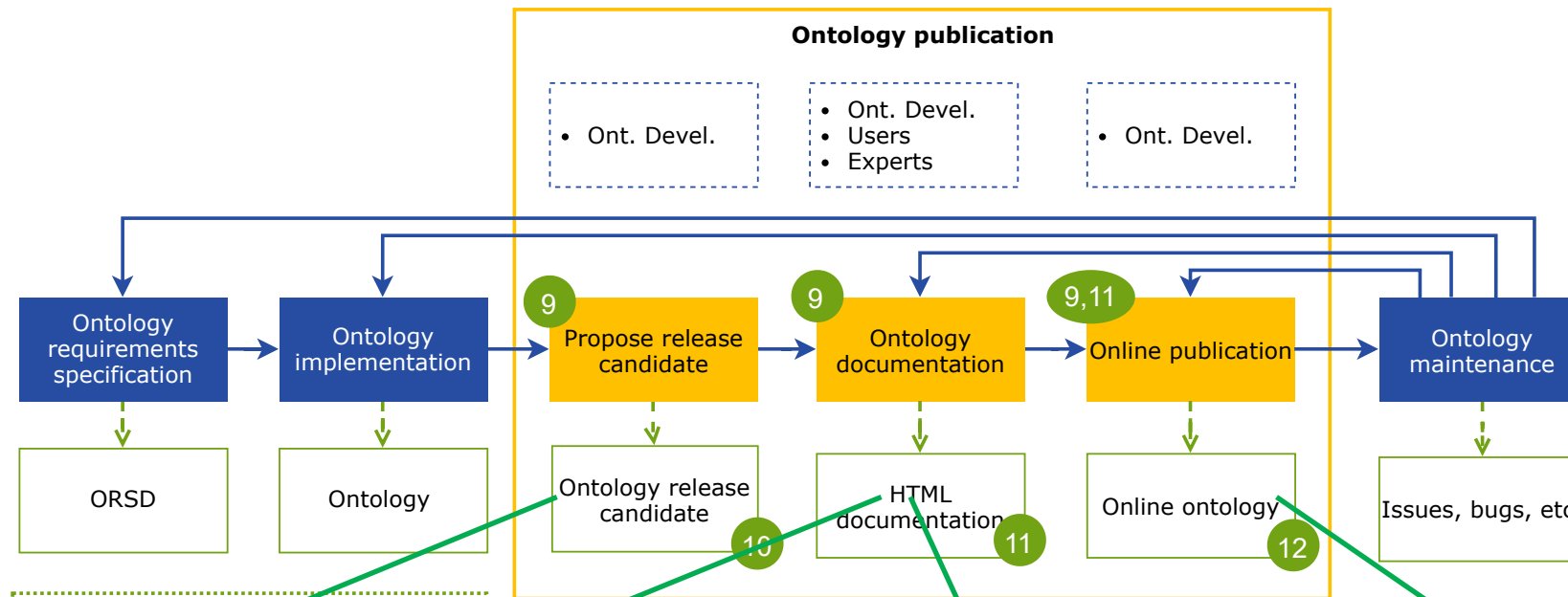
Results for P12: Equivalent properties not explicitly declared.

Results for P13: Inverse relationships not explicitly declared.

This pitfall appears when any relationship (except for those that are defined as symmetric properties) does not have an inverse relationship (owl:inverseOf) defined within the ontology.

- OOPS! has the following suggestions for the relationships without inverse:
 - > <http://data.semanticweb.org/ns/swc/ontology#hasPart> could be inverse of <http://data.semanticweb.org/ns/swc/ontology#isLocationFor>
 - > <http://data.semanticweb.org/ns/swc/ontology#isLocationFor> could be inverse of <http://ontology#hasLocation>
 - > <http://swrc.ontoware.org/ontology#participant> could be inverse of <http://swrc.ontoware.org/ontology#participant>
- Sorry, OOPS! has no suggestions for the following relationships without inverse:
 - > <http://www.w3.org/2002/12/cal/ical#component>
 - > <http://www.w3.org/2002/12/cal/ical#dtstamp>
 - > <http://www.w3.org/2002/12/cal/ical#dtstart>

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:oops="http://www.oeg-upm.net/oops#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#" >
  <rdf:Description rdf:about="http://www.oeg-upm.net/oops#suggestion" >
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class" />
  </rdf:Description>
  <rdf:Description rdf:about="http://www.oeg-upm.net/oops#fdea1aa6-71d6-4557-
a17a-dc3244ff536b" >
    <oops:hasCode rdf:datatype="http://www.w3.org/2001/XMLSchema#string">P10</
oops:hasCode>
    <oops:hasName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Missing
disjointness [1, 2, 3]</oops:hasName>
    <oops:hasDescription rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
The ontology lacks disjoint axioms between classes or between properties
that should be defined as disjoint.</oops:hasDescription>
    <rdf:type
rdf:resource="http://www.oeg-upm.net/oops#pitfall" />
    <oops:hasImportanceLevel rdf:datatype="http://www.w3.org/2001/XMLSchema#
string">Important</oops:hasImportanceLevel>
    <oops:hasNumberAffectedElements rdf:datatype="http://www.w3.org/2001/
XMLSchema#integer">1</oops:hasNumberAffectedElements>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.oeg-upm.net/oops/496ae03d-48c6-406d-8
d07-530bf05c9a61" >
    <oops:hasPitfall rdf:resource="http://www.oeg-upm.net/oops/fdea1aa6-71d6
-4557-a17a-dc3244ff536b" />
    <rdf:type rdf:resource="http://www.oeg-upm.net/oops#response" />
  </rdf:Description>
  <rdf:Description rdf:about="http://www.oeg-upm.net/oops#pitfall" >
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class" />
  </rdf:Description>
</rdf:RDF>
```



Referenced artefacts from other activities
 9 Evaluated ontology



GitHub



- HTML generation from OWL code
- Multilingual
- Separated sections

+ Diagrams and examples (Some ideas: <https://arxiv.org/abs/2003.13084>)

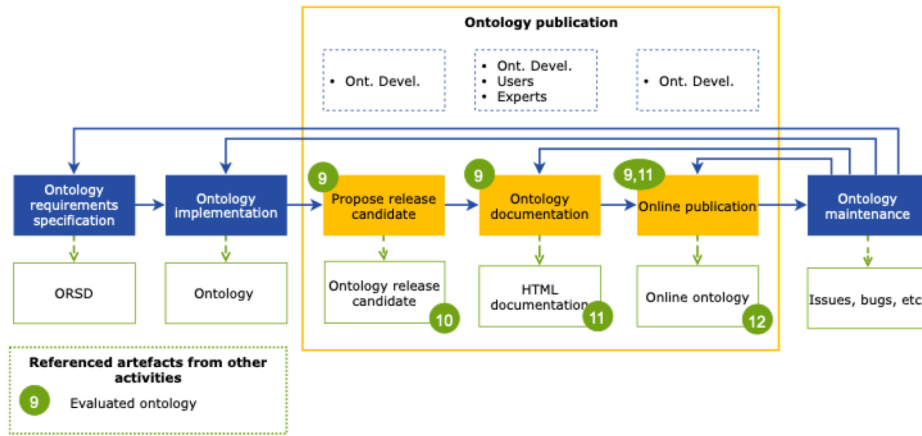
- + Some tools:
- + Draw.io
 - + yEd graph editor
 - + Microsoft Visio



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



<https://lov.linkeddata.es>



- Mission: promote and facilitate the reuse of well documented vocabularies in the Linked Data ecosystem

- Vocabularies registry and index

- Datalift

- <http://datalift.org/>



- Started at 2011

- Hosted by OEG

Openly reported in
GitHub issue tracker:
 new needs, bugs, etc.

Erroneous domain definitions #38
 Closed vcharpenay opened this issue on Jun 12, 2017 · 2 comments

vcharpenay commented on Jun 12, 2017

Some domain axioms seem erroneous:

- `:providesInteractionPattern rdfs:domain :InteractionPattern . I suppose you mean rdfs:range ?`
- `:name rdfs:domain :Thing` leads to the fact that all interaction patterns are also things, which is unwanted, I guess.

In general, are domain/range axioms supposed to remain in the ontology or will they eventually be removed?

mariapoveda commented on Jun 12, 2017

Thanks for the comments I'll update the ontology.
 I'd rather to keep them in the ontology.

mariapoveda added a commit that referenced this issue on Jun 12, 2017

0.0.7 replace erroneous domains issue #38

mariapoveda commented on Jun 12, 2017

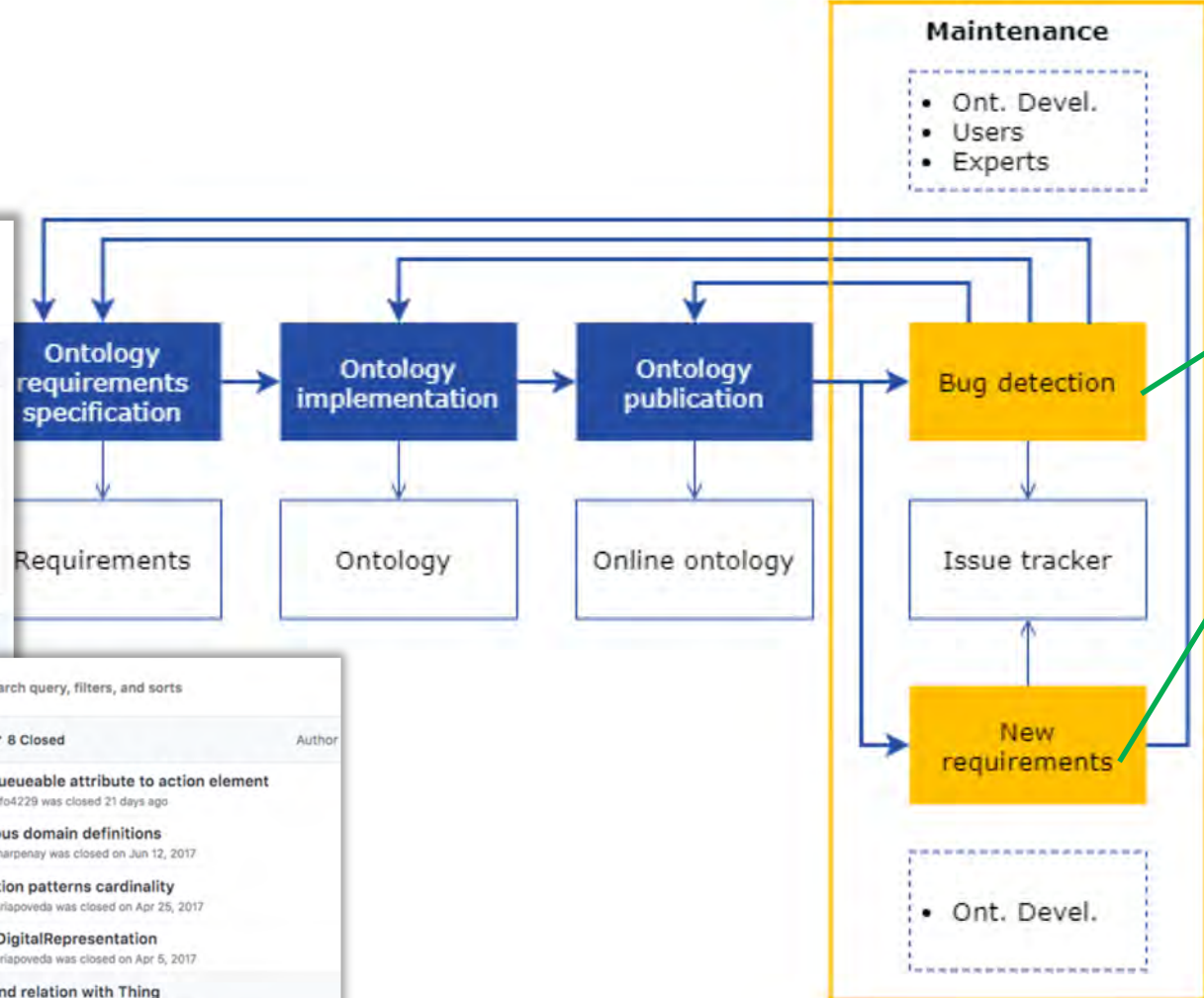
Closed in [ea38b5a](#)

mariapoveda closed this on Jun 12, 2017

Clear current search query, filters, and sorts

3 Open ✓ 8 Closed Author

- add a queueable attribute to action element #43 by sulfo4229 was closed 21 days ago
- Erroneous domain definitions #38 by vcharpenay was closed on Jun 12, 2017
- Interaction patterns cardinality #30 by mariapoveda was closed on Apr 25, 2017
- Delete DigitalRepresentation #20 by mariapoveda was closed on Apr 5, 2017
- WoT5 and relation with Thing #5 by mariapoveda was closed on Feb 16, 2017
- WoT1 terminology doubt #4 by mariapoveda was closed on Mar 7, 2017
- WoT15 #2 by mariapoveda was closed on Feb 16, 2017
- WoT11 #1 by mariapoveda was closed on Feb 16, 2017



- Maintenance**
- Ont. Devel.
 - Users
 - Experts

- Ont. Devel.

<https://cogito.iot.linkeddata.es/>



Ontologies
Ontology testing



Here you can find the ontologies developed for the project
If you want to contribute development, please follow the project



Requirements

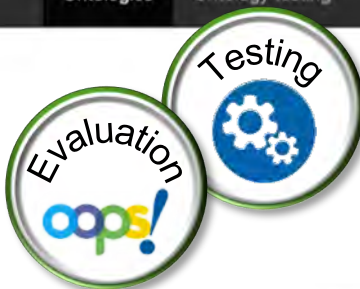


Version control

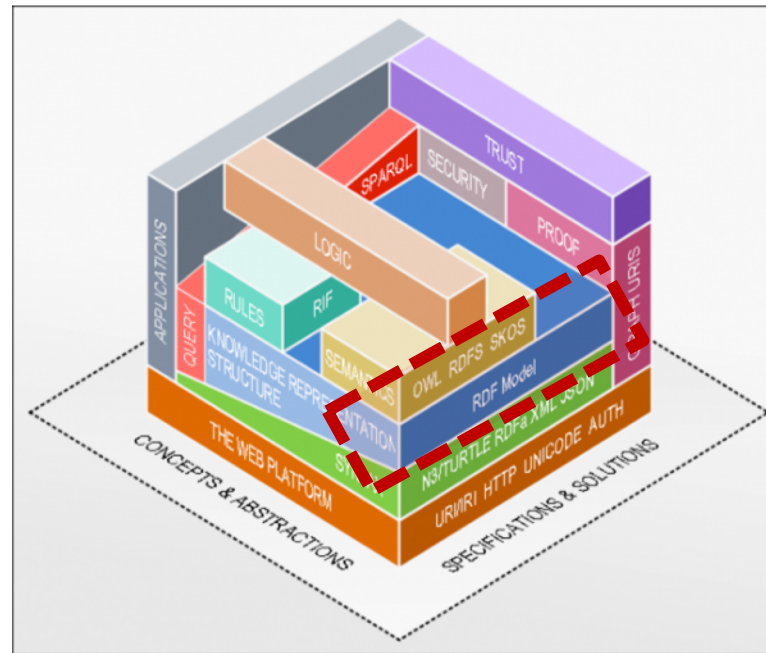
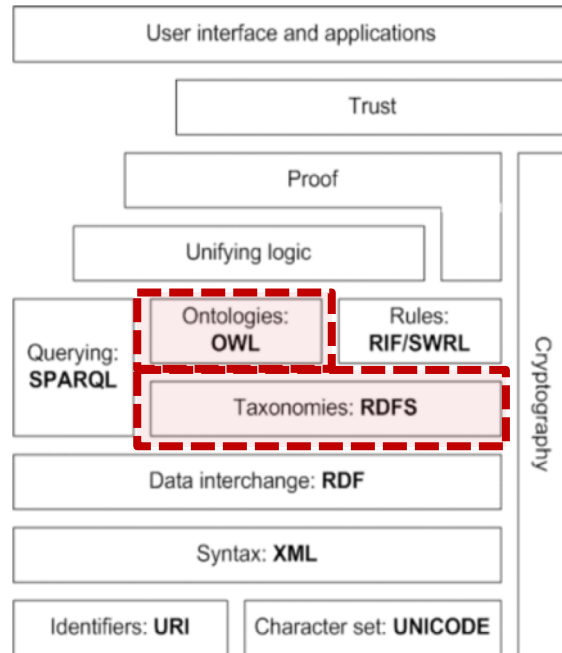


Issue tracker

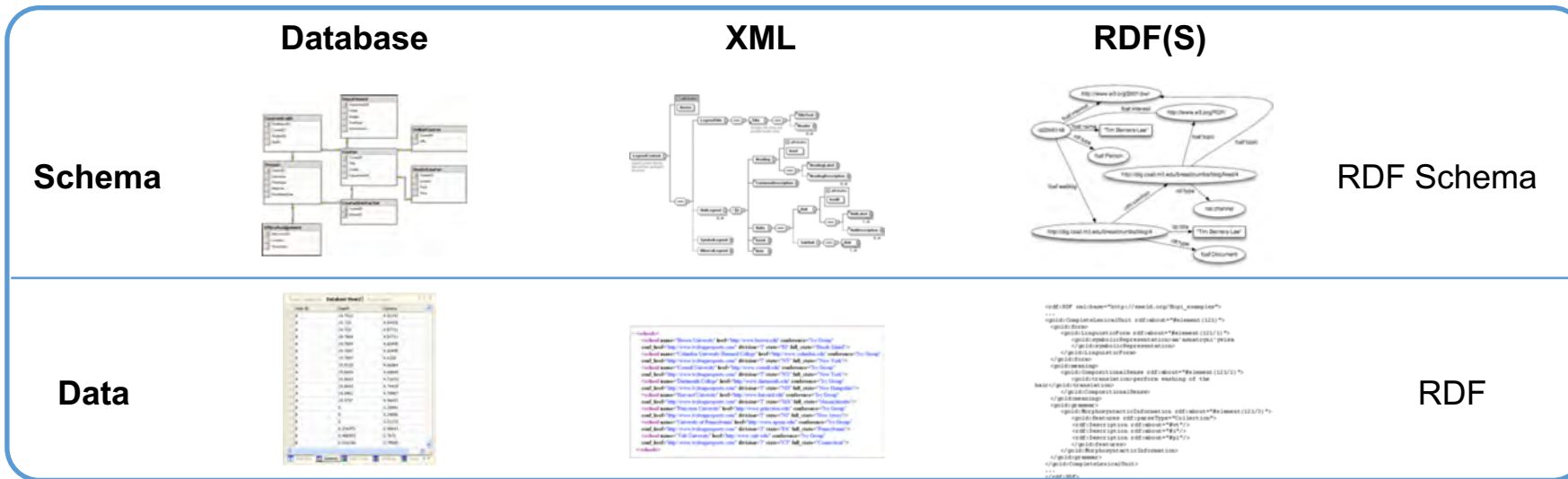
Ontology	Description	Requirements	Repository	Issue tracker	Releases
COGITO Process ontology	This ontology aims to model the construction process in the COGITO ontology	Ontology Requirements	Ontology Repository	Ontology Issue Tracker	Ontology Releases
COGITO Facility ontology	This ontology aims to model the construction data exchanges in the COGITO project	Ontology Requirements	Ontology Repository	Ontology Issue Tracker	Ontology Releases
COGITO Resources ontology	This ontology aims to model the resources in the COGITO project	Ontology Requirements	Ontology Repository	Ontology Issue Tracker	Ontology Releases
COGITO Quality ontology	This ontology aims to model the construction quality domain in the COGITO project	Ontology Requirements	Ontology Repository	Ontology Issue Tracker	Ontology Releases
COGITO Safety ontology	This ontology aims at modelling the safety in the construction domain in the COGITO project	Ontology Requirements	Ontology Repository	Ontology Issue Tracker	Ontology Releases
COGITO IoT ontology	This ontology aims at modelling the IoT	Ontology Requirements	Ontology Repository	Ontology Issue Tracker	Ontology Releases



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



■ RDF: Resource Description Framework



■ W3C Recommendation



- Model
- Syntax
- Semantics

Slide taken from “RDF, RDF Schema y SPARQL” by O. Corcho, R. García-Castro”

- You **can't coin** names in someone else's **namespace**
- RDF, RDF(S) and OWL provide standard constructs
 - → you **don't** need to **re-invent** them, you just provide the domain terms
 - And if you do, no one else would understand 😊

Classes

- **rdfs:Class**
 - Concepts of the domain (generally)
 - **Classes with name:**
 - URI as identifier

RDF declaration	Diagram proposal
 <p>ex:City $\xrightarrow{\text{rdf:type}}$ rdfs:Class</p>	 <p>ex:City</p>

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

RDF declaration	Diagram proposal
<p>ex:City $\xrightarrow{\text{rdfs:subClassOf}}$ ex:Municipality</p>	<pre>classDiagram class ex:City class ex:Municipality ex:City -- > ex:Municipality</pre> <p>The diagram shows two rectangular boxes. The top box is labeled 'ex:Municipality' and the bottom box is labeled 'ex:City'. A solid line connects the bottom box to the top box, ending in an open arrowhead pointing upwards, which represents a generalization relationship (subclassing).</p>



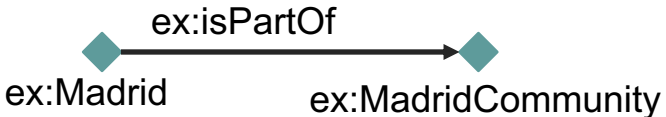
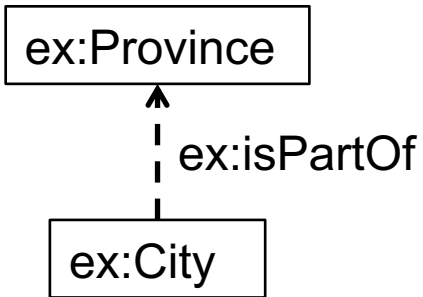
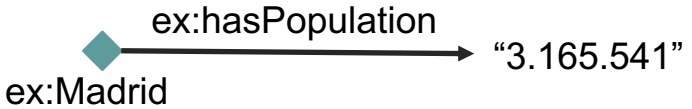
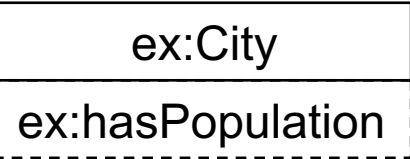
1. Define the classes in the proposed domain
2. Define hierarchies in the proposed domain

- 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 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

Properties

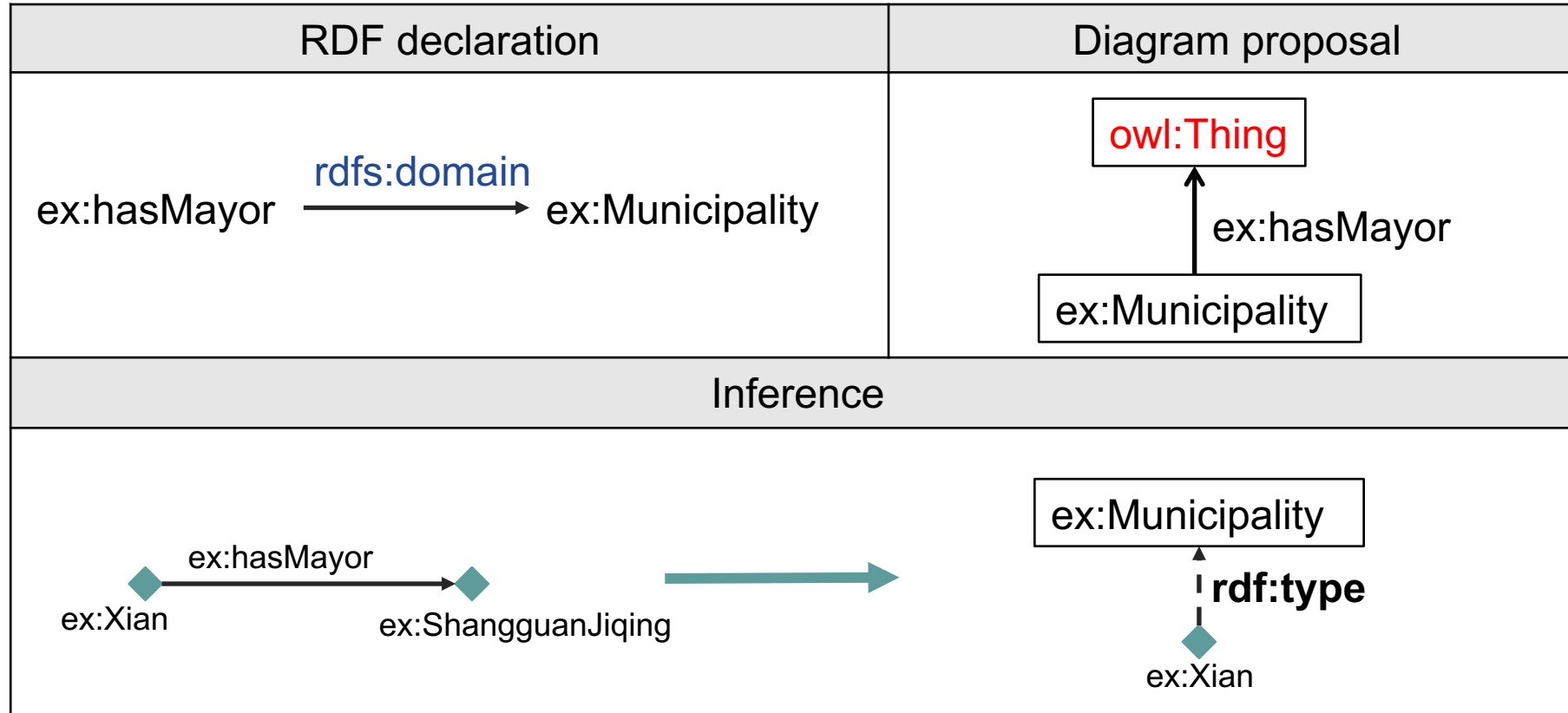
▪ **rdf:Property**

- Relation between individuals
- Relación between individuals and data values

RDF declaration	Diagram proposal
<p> <code>ex:isPartOf</code> $\xrightarrow{\text{rdf:type}}$ <code>rdf:Property</code> </p> <p>Relation expected to be use between individuals</p>  <pre> graph LR A[ex:Madrid] -- ex:isPartOf --> B[ex:MadridCommunity] </pre>	 <pre> graph BT A[ex:Province] -.-> ex:isPartOf B[ex:City] </pre>
<p> <code>ex:hasPopulation</code> $\xrightarrow{\text{rdf:type}}$ <code>rdf:Property</code> </p> <p>Example of relation expected to be use between individuals and a data values</p>  <pre> graph LR A[ex:Madrid] -- ex:hasPopulation --> B["3.165.541"] </pre>	 <pre> graph TD A[ex:City] --- B[ex:hasPopulation] </pre>

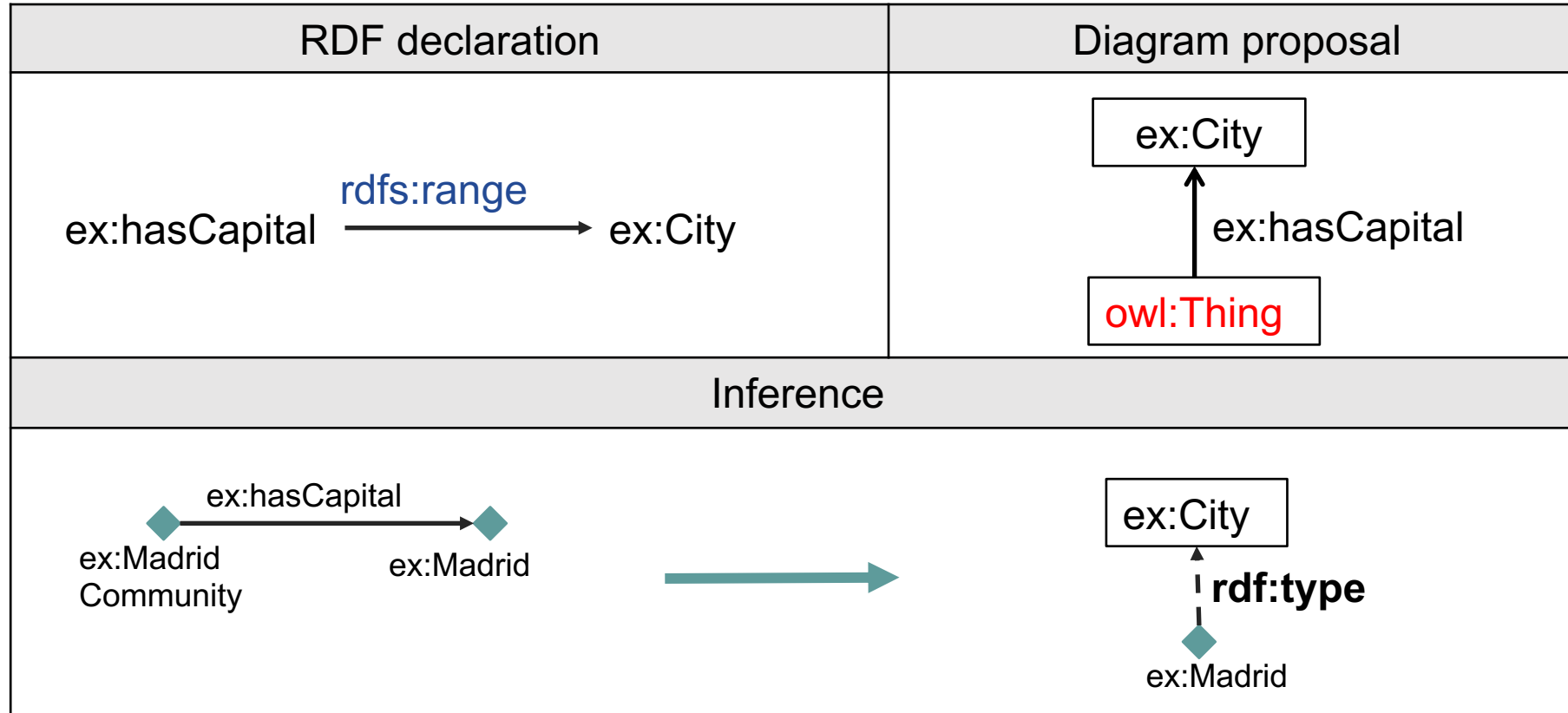
- **rdfs:domain**

- 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



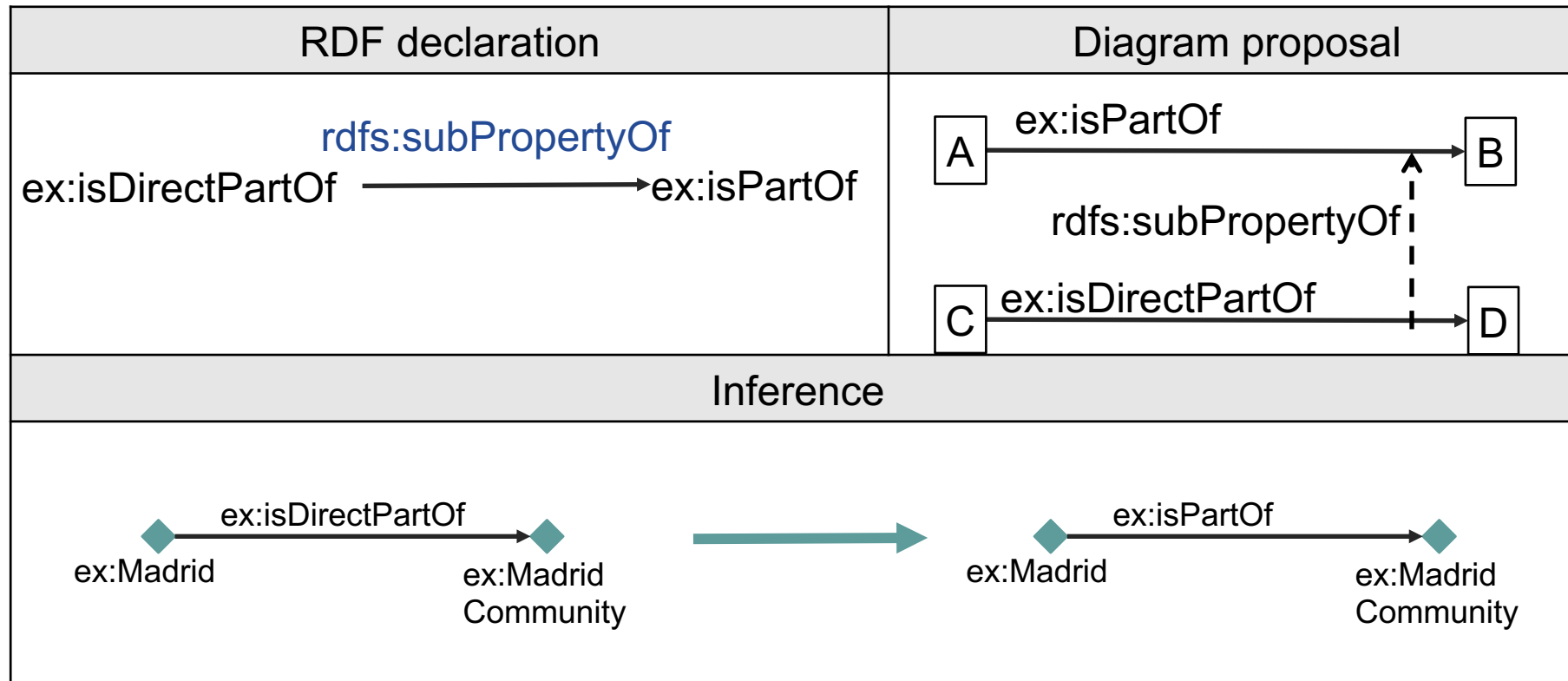
▪ `rdfs:range`

- 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 classes (applies to relations) or datatypes (applies to attributes)



▪ 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
- Applicable to *object properties (relations)* and *datatype properties (attributes)*.





Update your model including properties

Identify properties between concepts

Decide whether to add domain and/or range

Individuals

- **rdf:type**
 - Indicate the class or classes of an individual (instance)

RDF declaration	Diagram proposal
<p style="text-align: center;"> <code>ex:Madrid</code> $\xrightarrow{\text{rdf:type}}$ <code>ex:City</code> </p>	

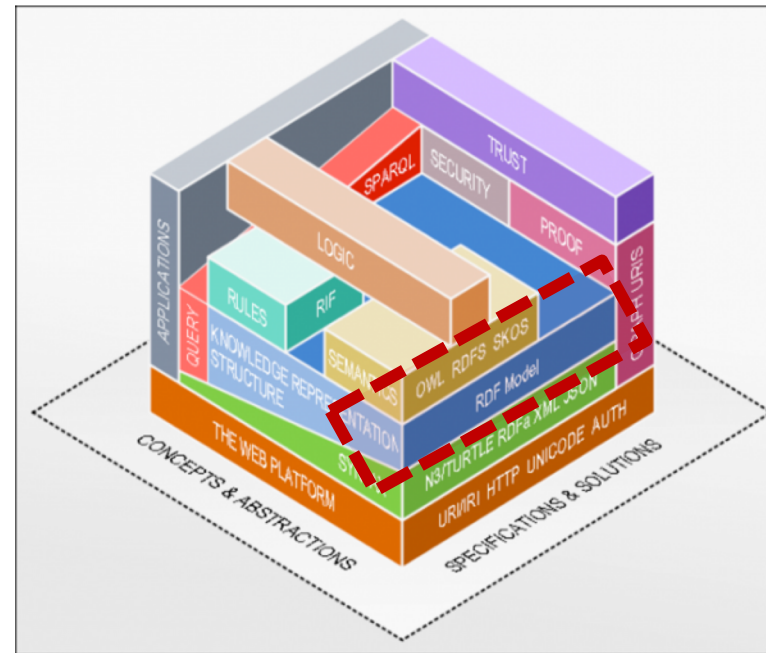
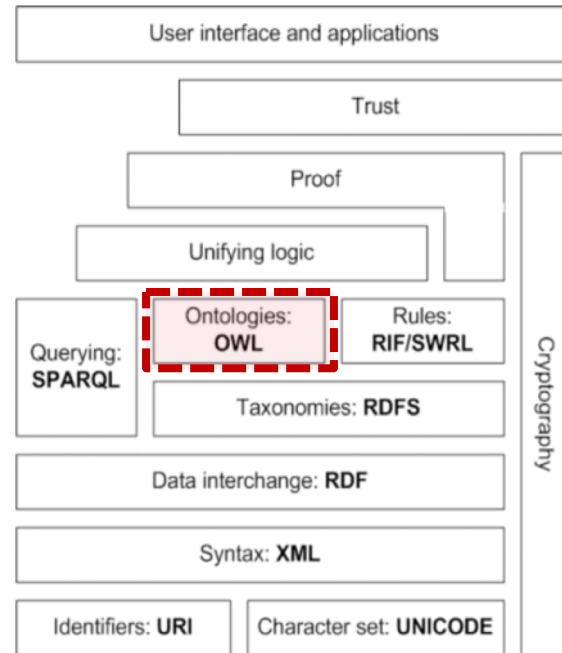


What can't be modelled from the domain?

- 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 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



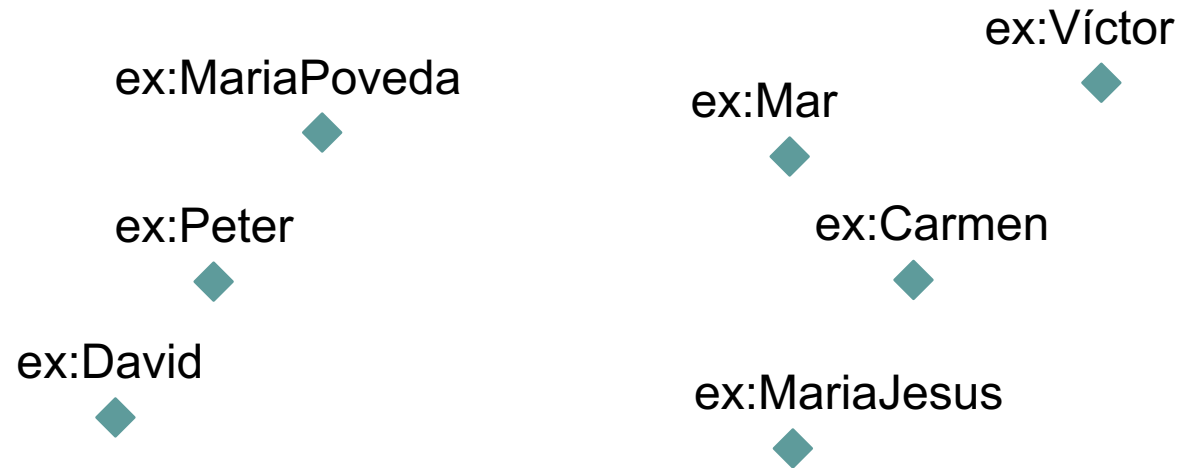
<http://w3.org/DesignIssues/diagrams/sweb-stack/2006a.png>

- OWL: Web Ontology Language
- 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

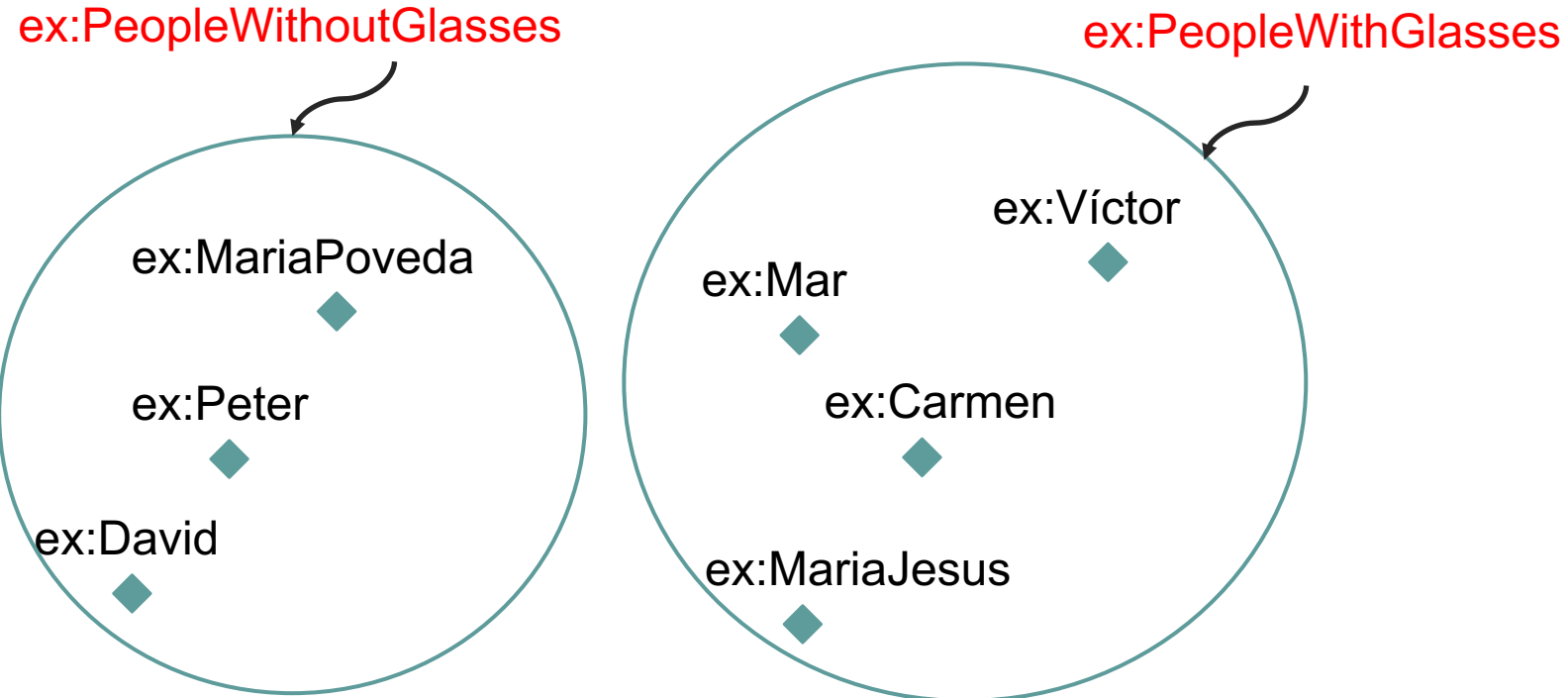
- OWL
 - **Classes**
 - Axioms
 - Properties
 - Characteristics
 - Individuals
 - Restrictions

- 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



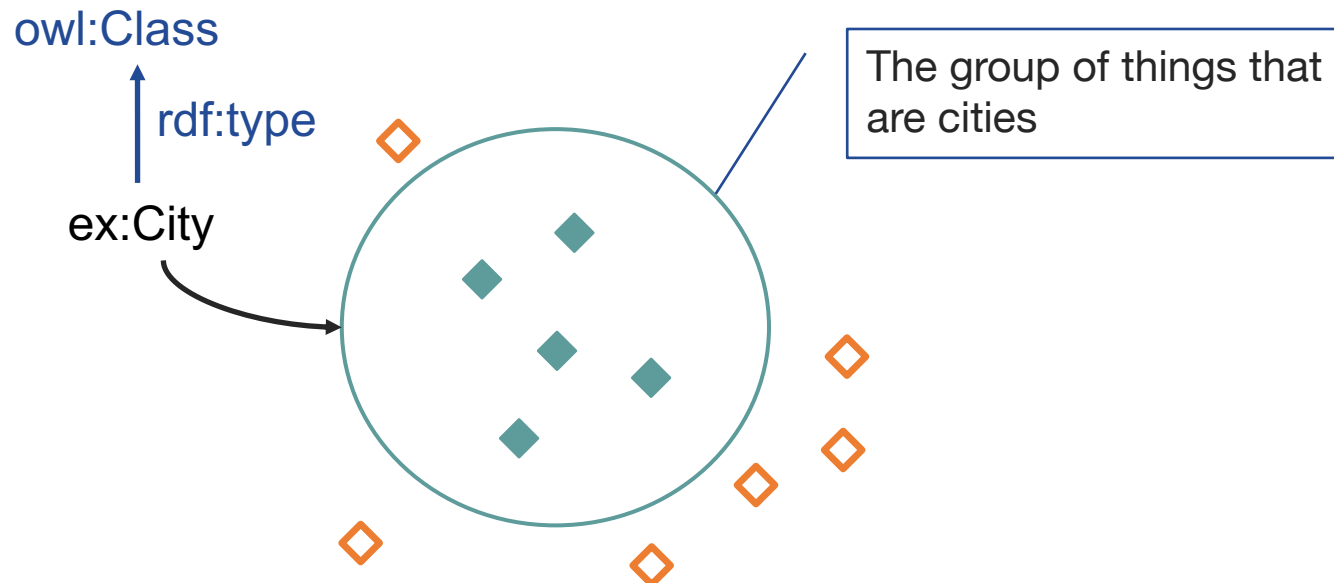
Universe

- 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

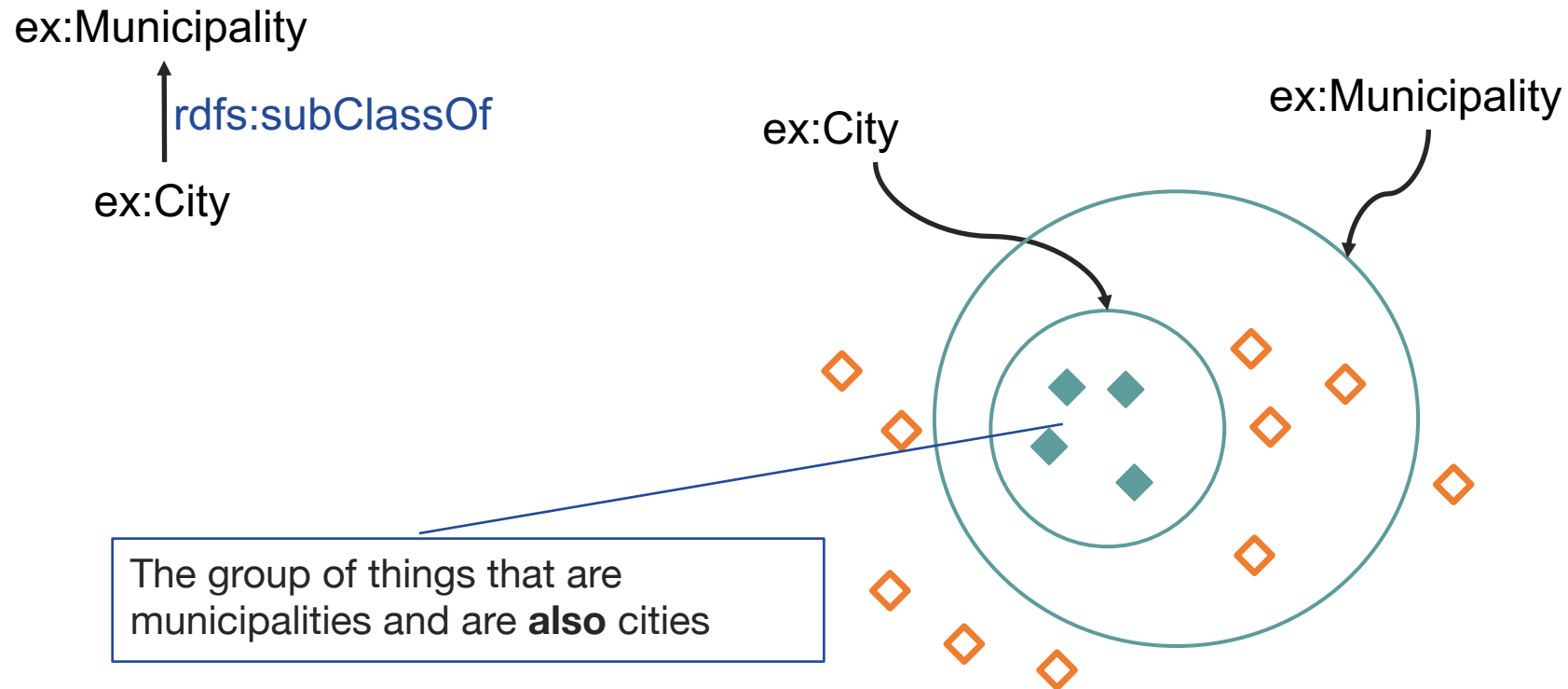
owl:Class

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



- **rdfs:subClassOf**

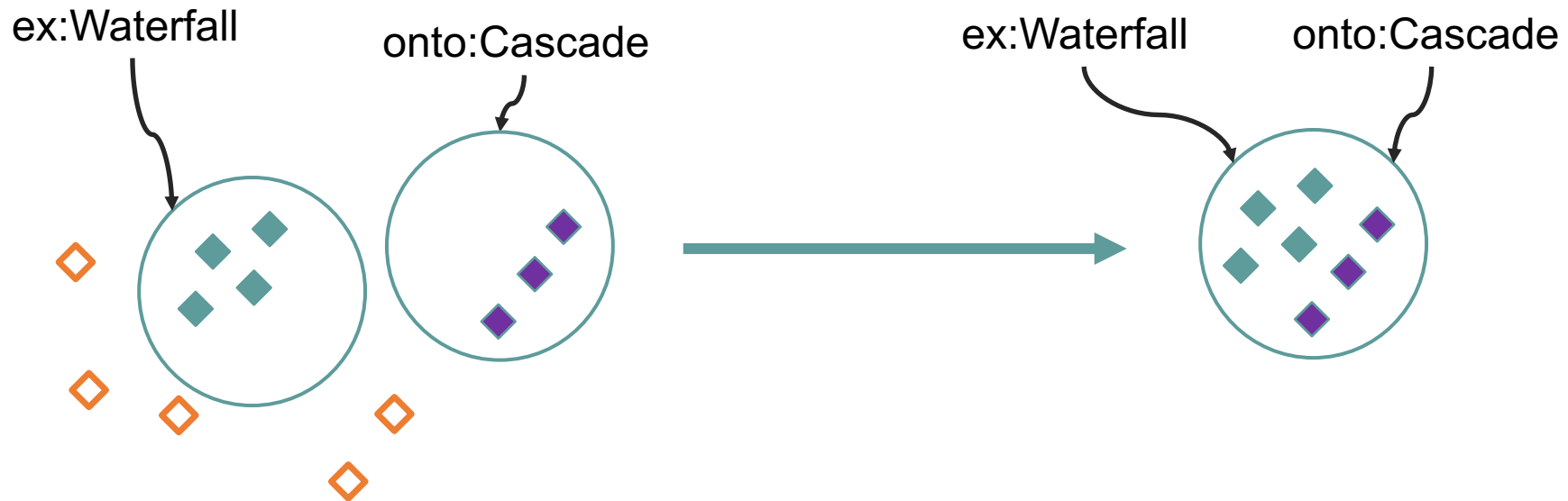
- The individuals belonging to a class also belong to the parent classes in the hierarchy



- **owl:equivalentClass**

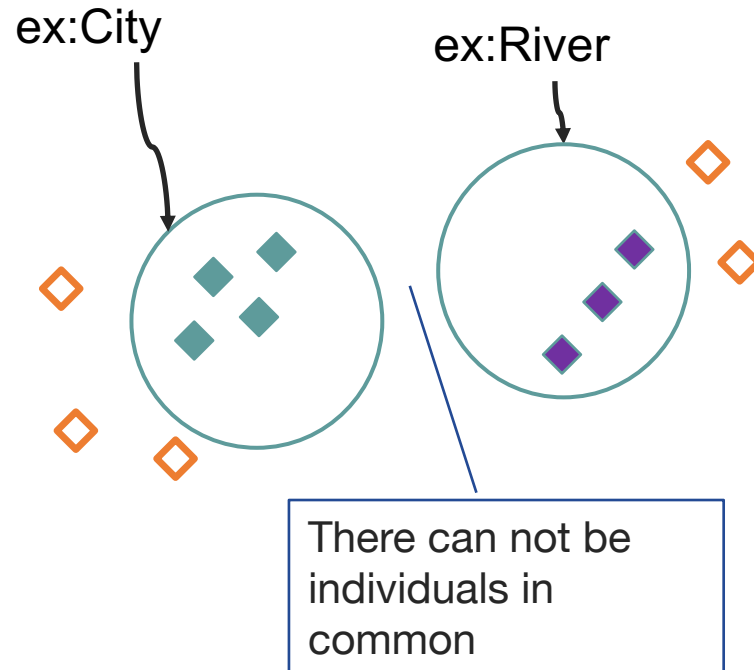
- The two classes contain exactly the same individuals
- If an individual belongs to a class it also belongs to the other

ex:Waterfall $\xrightarrow{\text{owl:equivalentClass}}$ onto:Cascade



- **owl:disjointWith**

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

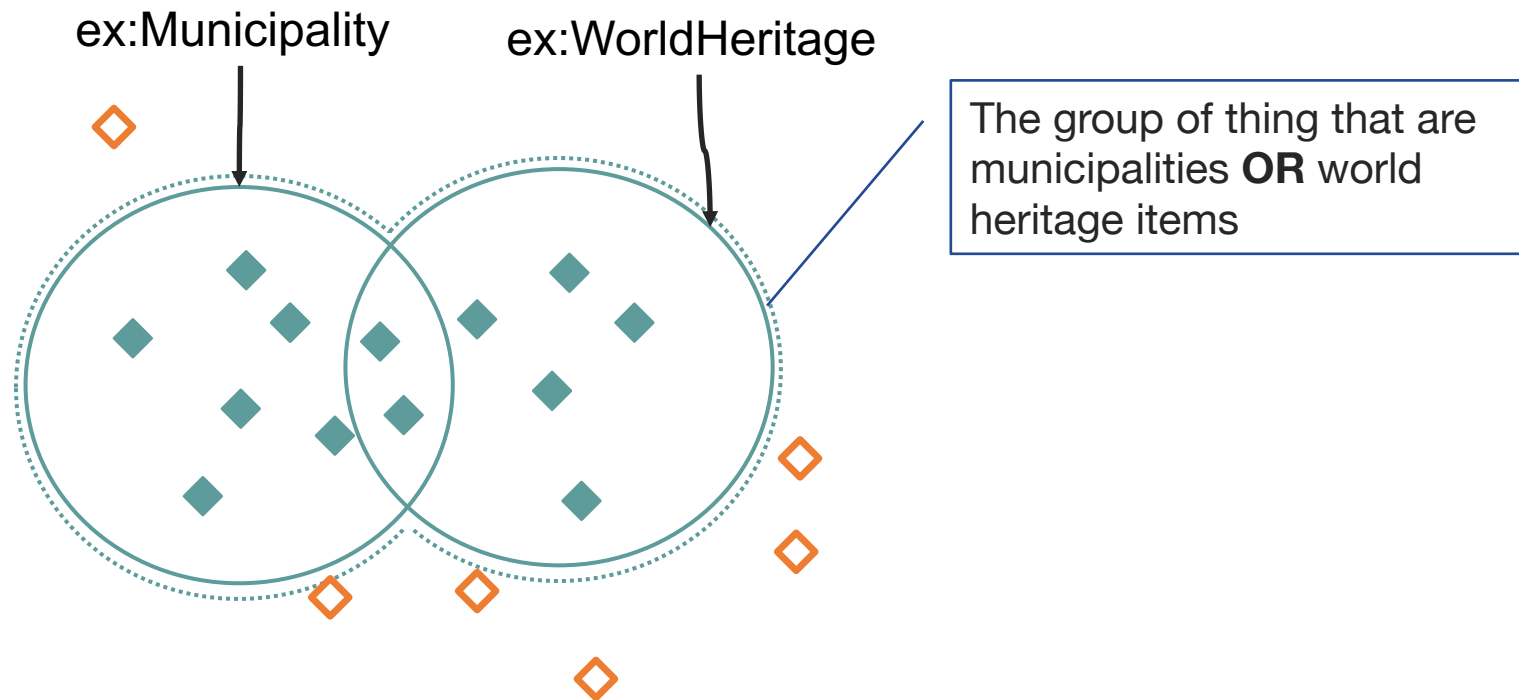


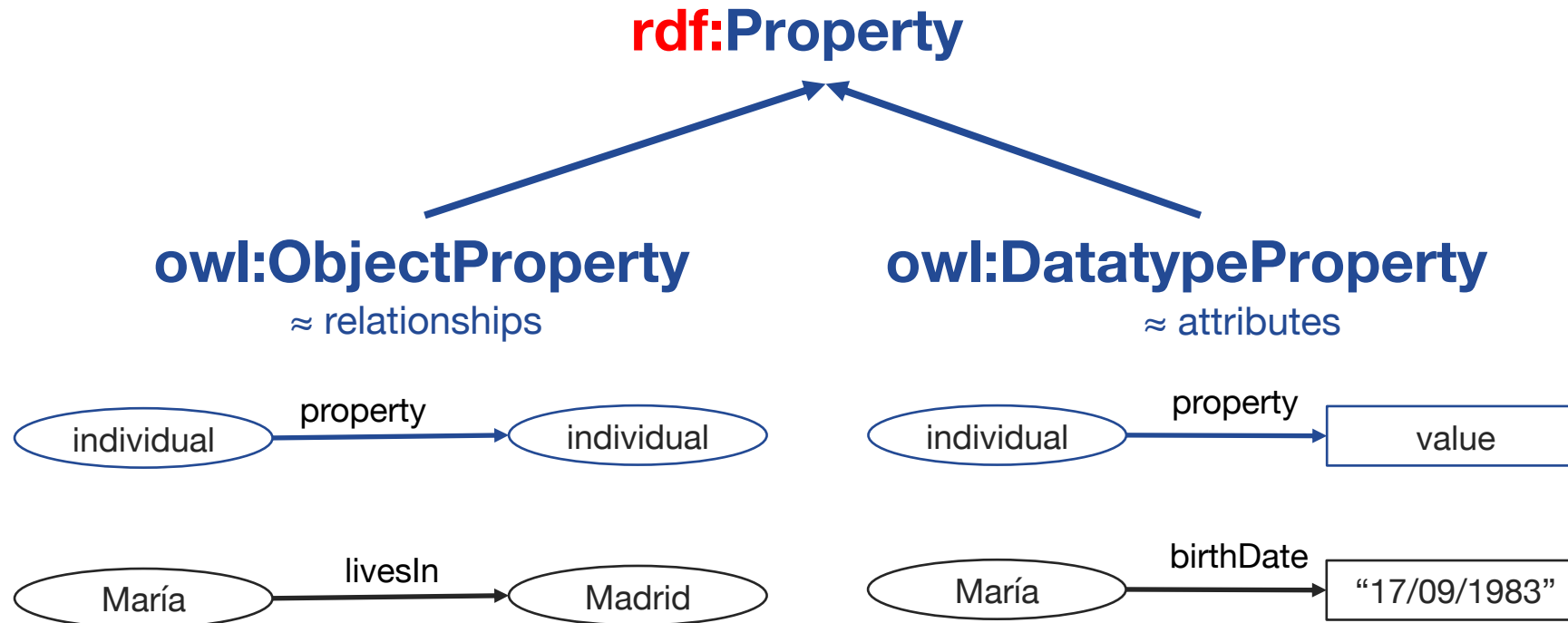
- **owl:unionOf**
- **owl:intersectionOf**
- **owl:complementOf**
- **owl :Thing**
- **owl :Nothing**

- OWL
 - Classes
 - Axioms
 - **Properties**
 - Characteristics
 - Individuals
 - Restrictions

- **owl:unionOf**

- An individual could belong to one or more of the classes involved
- **OR** logic





▪ **rdfs:domain**

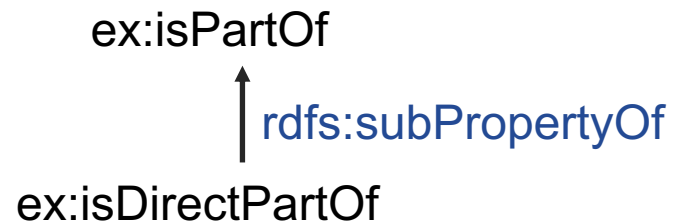
- All individuals that appear as **subject** in a triple with the given property **belongs to** the **class** defined as **domain** of the property
- It is valid for relations and attributes

▪ **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 classes (applies to relations) or datatypes (applies to attributes)

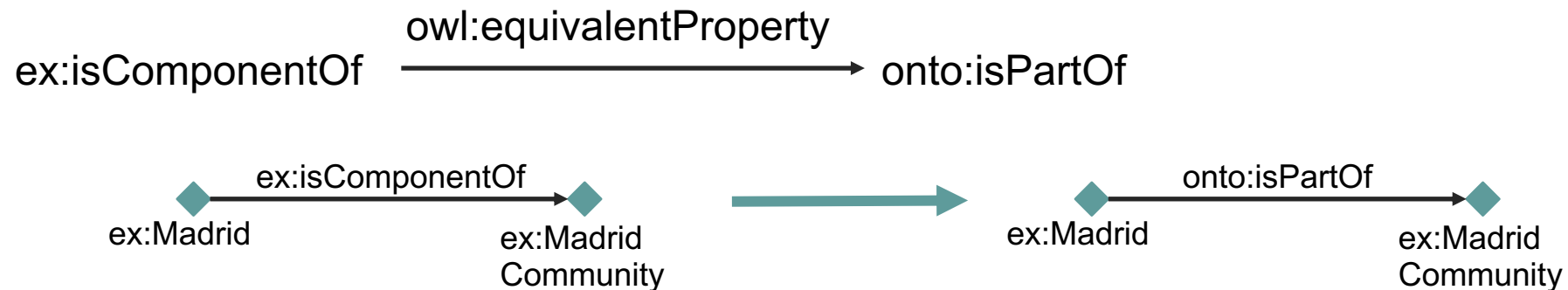
- **rdfs:subPropertyOf**

- 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
- Applicable 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
- Applicable 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*.

ex:isPartOf
 ↑ owl:inverseOf
 ex:hasPart

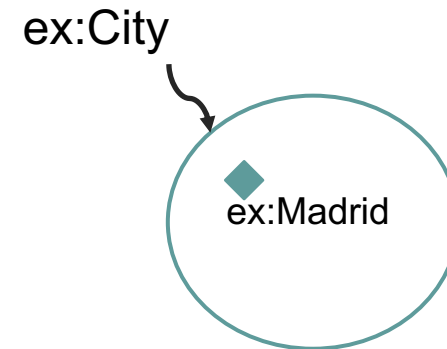
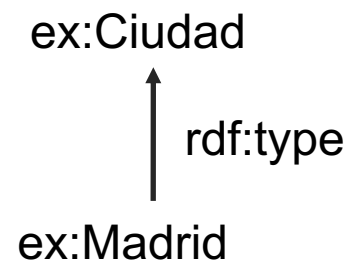


- **owl:FunctionalProperty**
- **owl:InverseFunctionalProperty**
- **owl:TransitiveProperty**
- **owl:SymmetricProperty**

- OWL
 - Classes
 - Axioms
 - Properties
 - Characteristics
 - **Individuals**
 - Restrictions

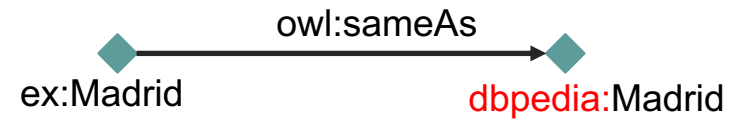
- **rdf:type**

- Indicates the class or classes the individual belongs to

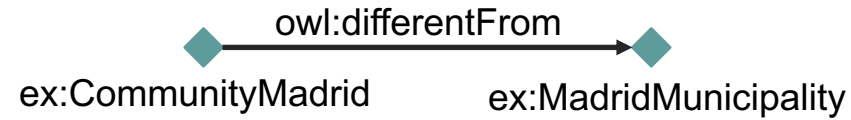


■ owl:sameAs

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

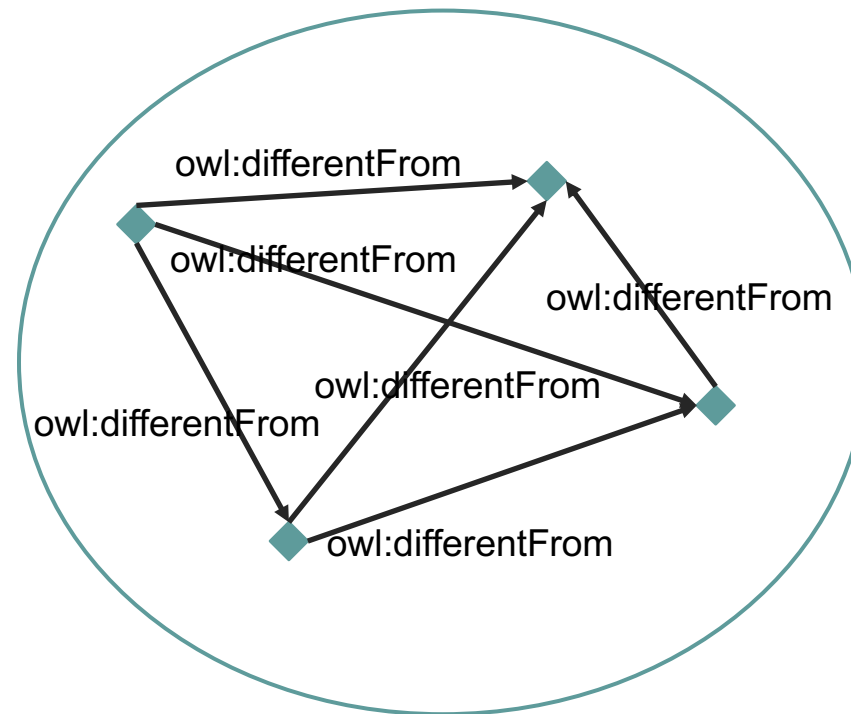


- **owl:differentFrom**
 - Declare that two URIs identify different individuals



■ owl:AllDifferent

- Declare that all URIs indicated identify different individuals
- Normally used to force the *unique name assumption*



- *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



Advanced ontologies and reasoning

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



✉ [mpoveda@fi.upm.es]

🐦 @MariaPovedaV

📍 SSoLDAC23