S-TEN

Intelligent Self-describing Technical and Environmental Networks

A research project partially funded by the European Commission
month 12 of 30

S-TEN Partner

FGH - Forschungsgemeinschaft für Elektrische Anlagen und Stromwirtschaft e.V.	🇩🇪
Caesar - Caesar Systems Limited	🇬🇧
Cygnus - Cygnus Engineering AG	🇨🇭
HEVS – Haute Ecole Valaisanne	🇨🇭
Labein - Fundacion Labein	🇪🇸
LKBaltic - UAB LKSoft Baltic	🇱🇹
LKSoft – LKSoftWare GmbH	🇩🇪
Racos - Racos Technische Informationssysteme	🇩🇪

One of the Scope items: Linking the Semantic Web (OWL) with design knowledge (STEP)

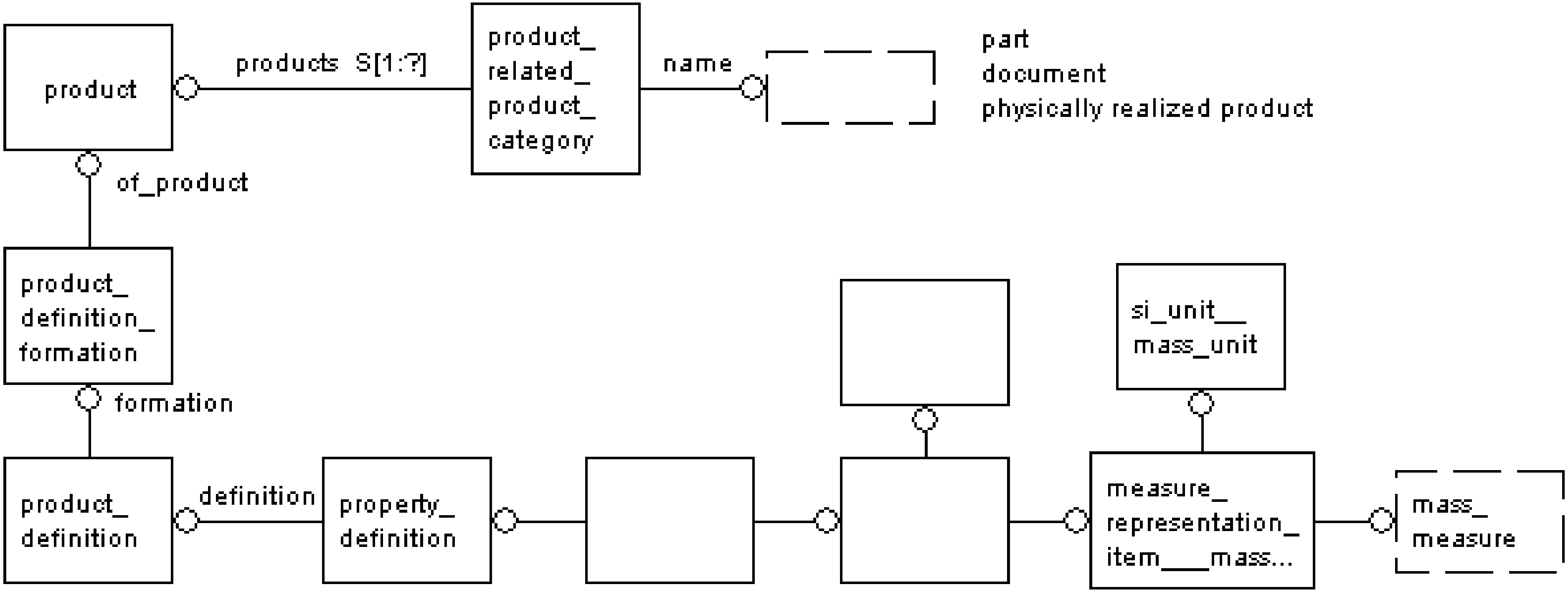
- The development of two way translators between data represented according to STEP and data represented according to the OWL ontologies. The translators will be used within the demonstrators to provide design data that can be accessed by inference engines, and to visualise a self-describing network.

Standardisation

- Definition of standard ontologies for the publication of network information on the Web
  - ISO 15926 (Life cycle data for process plant, including oil and gas production), committee: TC184/SC4 WG3 T25

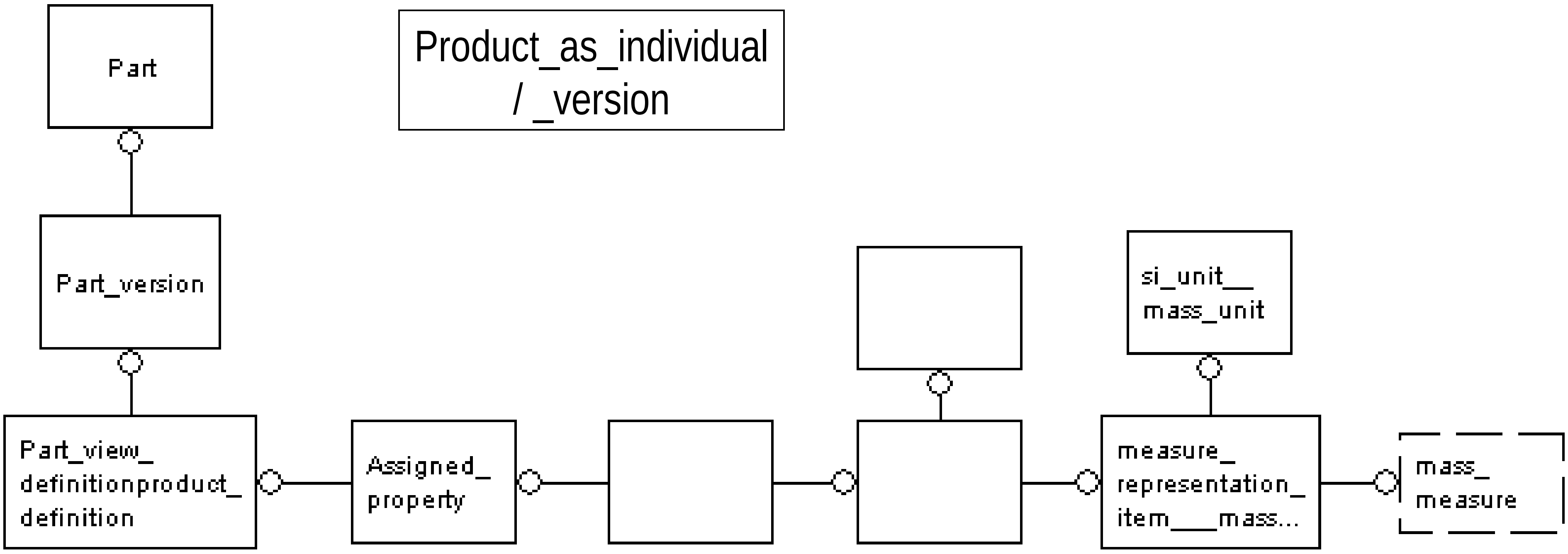
- Definition of a methodology for the extraction of ontologies from existing international standards.
  - STEP (ISO 10303 Product data representation and exchange), committee ISO TC184/SC4 WG12
  - IEC 61970 (derive APIs from the Common Information Model (CIM) for electrical distribution and transmission networks), committee IEC TC57 WG13

Some core STEP entities (AIM/MIM)

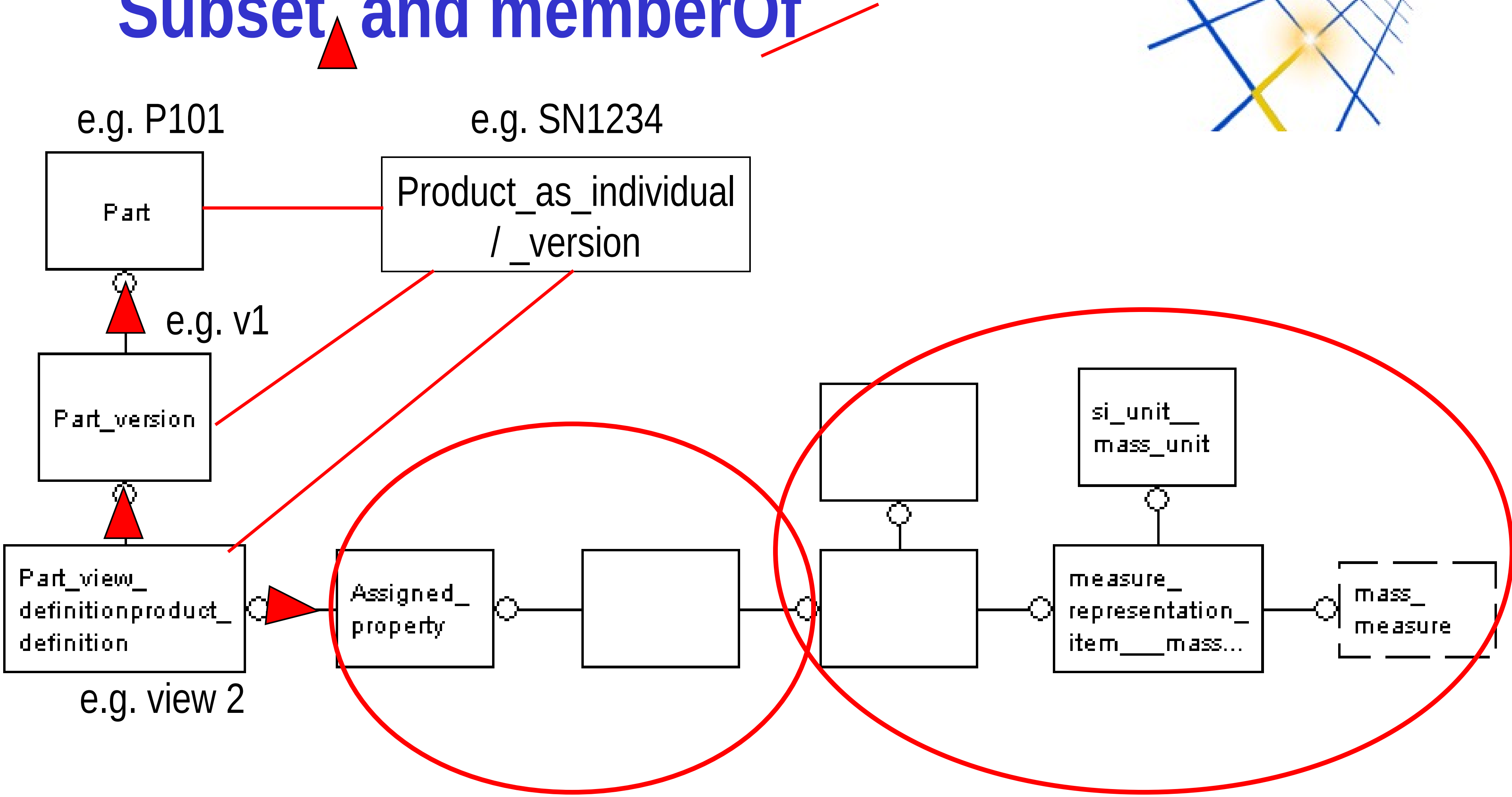


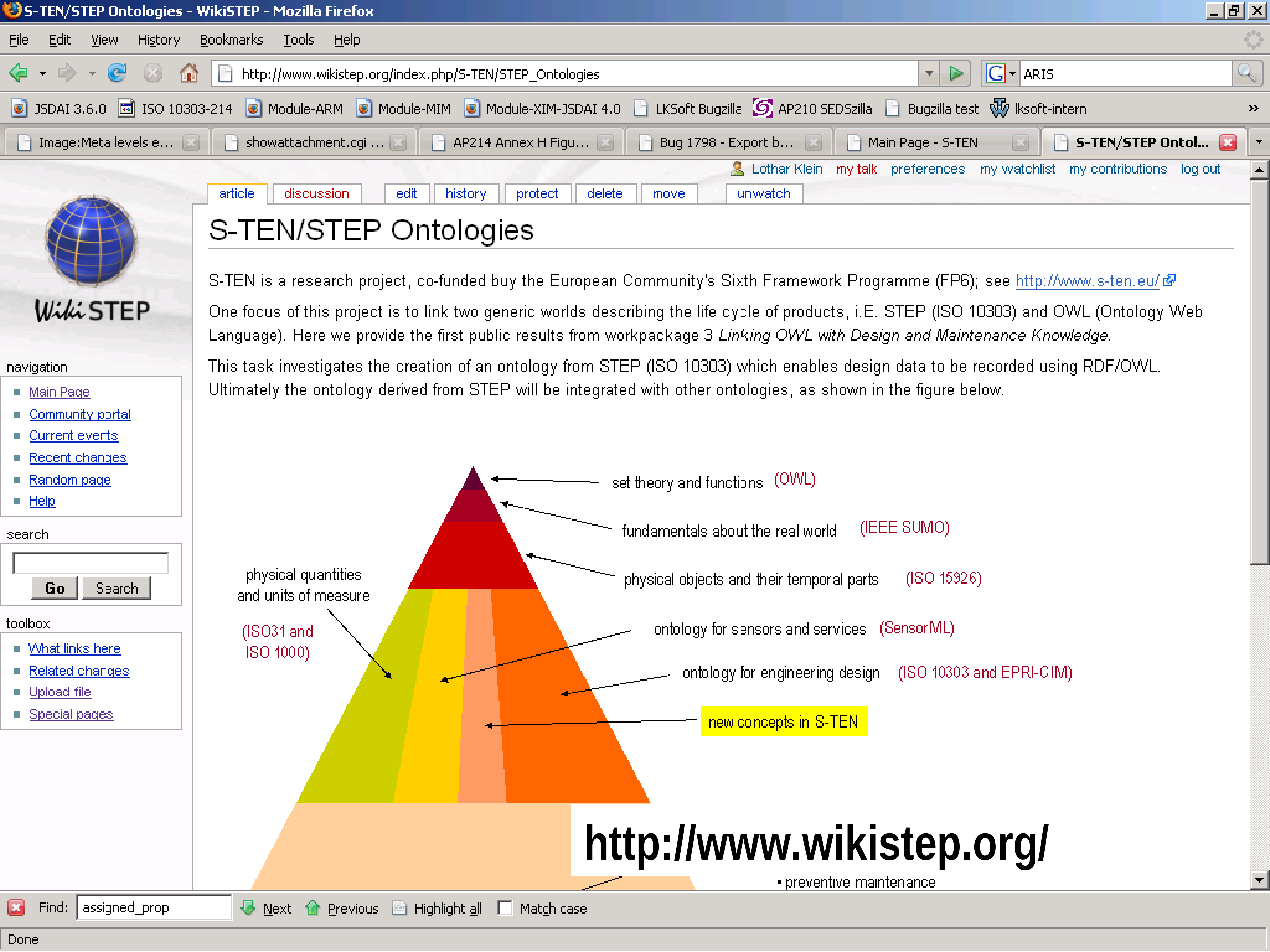
Information Society Technologies

The ARM has more semantics ...


Part
Product_as_individual / _version
Part_version
si_unit__ mass_unit
Part_view_ definitionproduct_ definition
Assigned_ property
measure_ representation_ item___mass...
mass_ measure

Set theory relations: Subset and memberOf


e.g. P101
e.g. SN1234
Part
Product_as_individual / _version
e.g. v1
Part_version
Part_view_ definitionproduct_ definition
e.g. view 2
Assigned_ property
si_unit_ mass_unit
measure_ representation_ item___mass...
mass_ measure



S-TEN/STEP Ontologies

S-TEN is a research project, co-funded buy the European Community's Sixth Framework Programme (FP6); see http://www.s-ten.eu/

One focus of this project is to link two generic worlds describing the life cycle of products, i.E. STEP (ISO 10303) and OWL (Ontology Web Language). Here we provide the first public results from workpackage 3 *Linking OWL with Design and Maintenance Knowledge*.

This task investigates the creation of an ontology from STEP (ISO 10303) which enables design data to be recorded using RDF/OWL. Ultimately the ontology derived from STEP will be integrated with other ontologies, as shown in the figure below.

set theory and functions (OWL)
fundamentals about the real world (IEEE SUMO)
physical objects and their temporal parts (ISO 15926)
physical quantities and units of measure
(ISO31 and ISO 1000)
ontology for sensors and services (SensorML)
ontology for engineering design (ISO 10303 and EPRI-CIM)
new concepts in S-TEN
• preventive maintenance

http://www.wikistep.org/

The question

We know information about “part version” (or “produce definition formation”) XYZ.

- Information can be relationships (such as assembly structure) or physical properties.

We know individual item 98/1234 is of type XYZ

What more information do we know about 98/1234?

- What relationships does it have?
- What physical properties does it have?

A more difficult question

We know information about “part version” (or “produce definition formation”) XYZ.

- Information can be relationships (such as assembly structure) or physical properties.

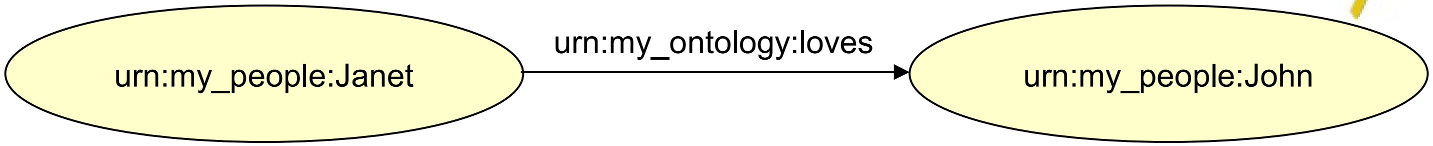
We know information about individual item 98/1234.

- What relationships it has.
- What physical properties it has.

Is individual item 98/1234 of type XYZ?

RDF – XML and N3

a statement: Janet loves John.

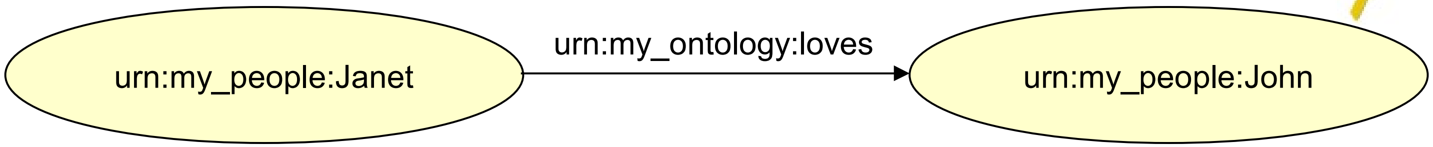


RDF is a graphical language

Each object and each relationship has a URI.

RDF – XML and N3

a statement: Janet loves John.



urn:my_people:Janet

urn:my_ontology:loves

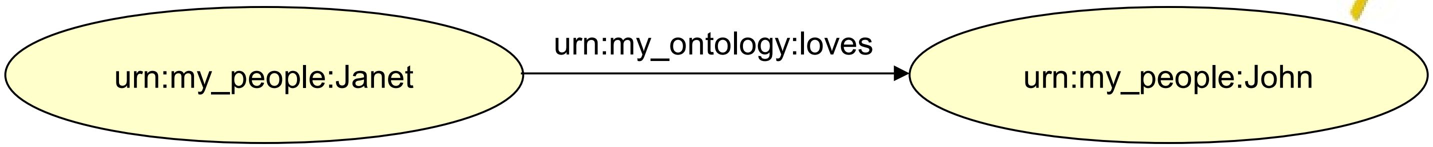
urn:my_people:John

serialised in XML:

```
<owl:Thing rdf:about=“urn:my_people:Janet”>
  <urn:my_ontology:loves rdf:resource=“urn:my_people:John”/>
</owl:Thing>
```

RDF – XML and N3

a statement: Janet loves John.

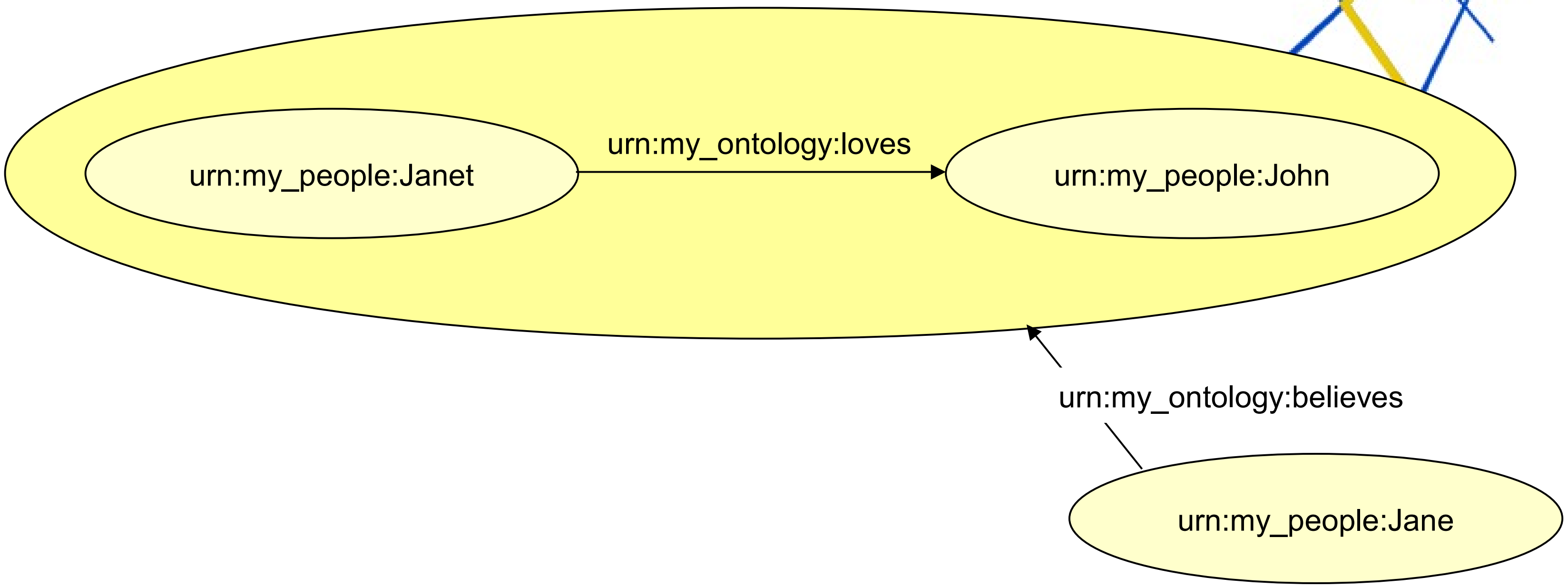


serialised in N3:

```
urn:my_people:Janet   urn:my_ontology:loves    urn:my_people:John .
```

Provenance and trust

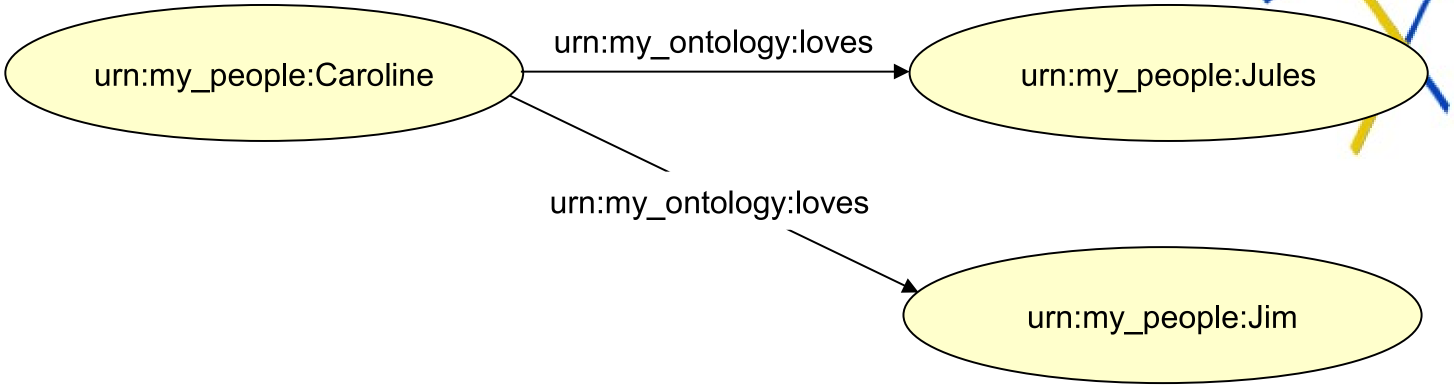
a statement: Janet loves John.



```
urn:my_people:Jane   urn:my_ontology:believes
  { urn:my_people:Janet   urn:my_ontology:loves urn:my_people:John } .
```

RDF – XML and N3

a statement: Caroline loves Jules and Jim.



urn:my_people:Caroline

urn:my_ontology:loves

urn:my_people:Jules

urn:my_ontology:loves

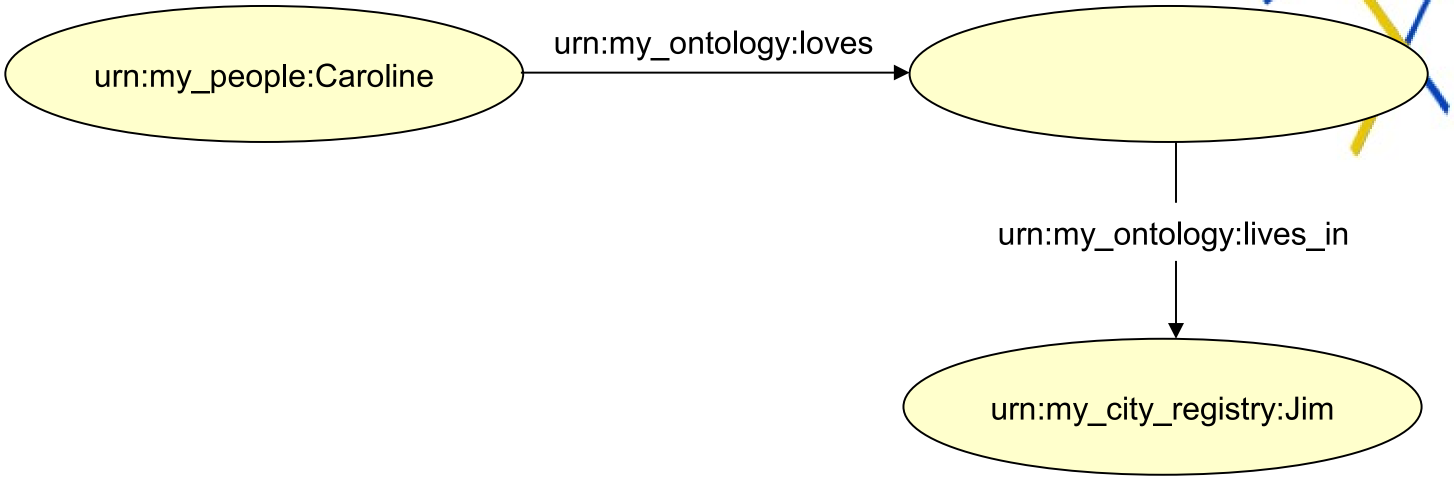
urn:my_people:Jim

serialised in N3:

```
urn:my_people:Caroline  urn:my_ontology:loves    urn:my_people:Jules ;
                        urn:my_ontology:loves    urn:my_people:Jim   .
```

RDF – XML and N3

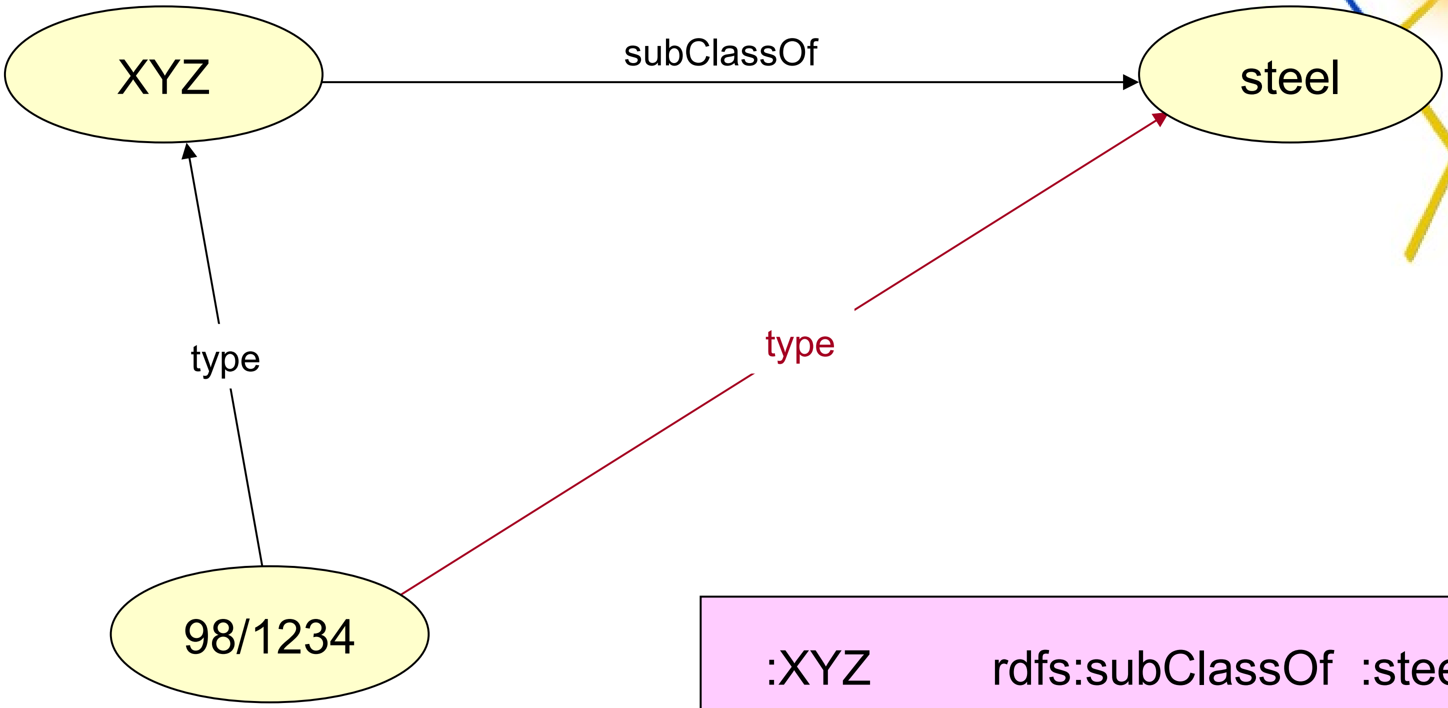
a statement: Caroline loves somebody who lives in London.



serialised in N3:

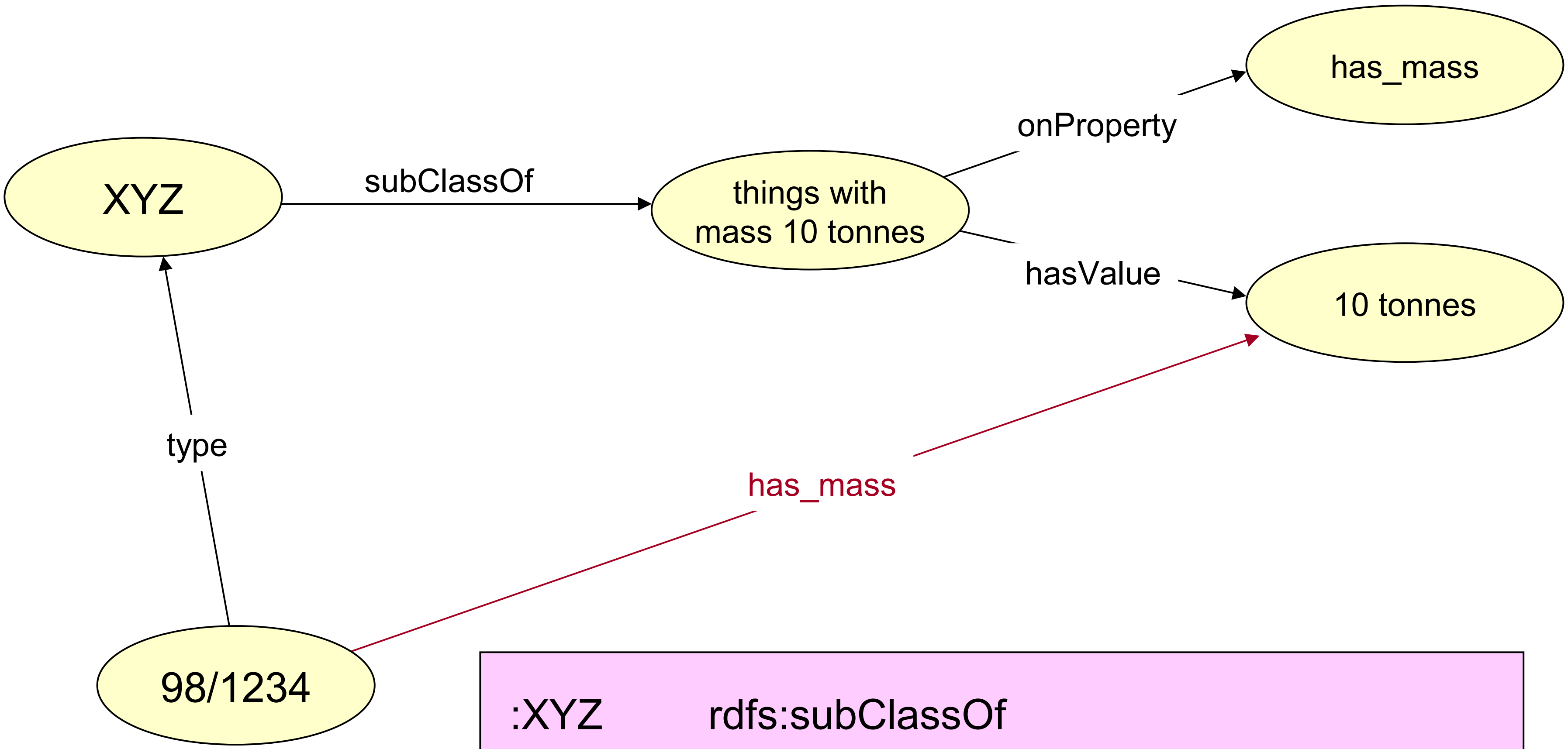
```
urn:my_people:Caroline   urn:my_ontology:loves
            [ urn:my_ontology:lives_in    urn:my_cities:London ]    .
```

Simple deduction



```
:XYZ        rdfs:subClassOf  :steel .

:98/1234   a                 :XYZ  ;
           a                 :steel .
```

Deduction of property



```
:XYZ        rdfs:subClassOf
                [   owl:onProperty    :has_mass ;
                    owl:hasValue    [  :tonnes  “10”  ]  ] .

:98/1234    a                   :XYZ  ;
            :has_mass           [  :tonnes  “10”  ] .
```

Physical Quantity := Value * Unit

```
#500=(MASS_UNIT()NAMED_UNIT(*)SI_UNIT($,.TONNE.));
#514=REPRESENTATION_CONTEXT(' ',' ');
#515=REPRESENTATION(' ',(#516),#514);
#516=(LENGTH_MEASURE_WITH_UNIT()MEASURE_REPRESENTATION_ITEM()
MEASURE_WITH_UNIT(MASS_MEASURE(4.5),#500)REPRESENTATION_ITEM(' '));
```

```
<iso_31:Mass rdf:about="#_516">
  <iso_1000:tonne_scale>
    <iso:Real>
      <value rdf:datatype="&xsd;double">10</value>
    </iso:Real>
  </iso_1000:tonne_scale>
</iso_31:Mass>
```

A property establishes a restriction class ...

```
#515=REPRESENTATION(' ',(#516),#514);
…
#512=PROPERTY_DEFINITION(Mass when empty',$,#314);
#513=PROPERTY_DEFINITION_REPRESENTATION(#512,#515);
```

```
<owl:FunctionalProperty rdf:about="#_510">
  <step:id rdf:datatype="&xsd;string">Mass when empty</step:id>
</owl:FunctionalProperty>

<step:Property_definition rdf:about="#_512">
  <owl:equivalentClass>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#_510"/>
      <owl:hasValue rdf:resource="#_516"/>
    </owl:Restriction>
  </owl:equivalentClass>
</step:Property_definition>
```

A Part_view_definition is the intersection of its property (and assembly component) classes.

```
#314=PRODUCT_DEFINITION('2',$,#311,#303);
…
#512=PROPERTY_DEFINITION('Overall length',$,#314);
#513=PROPERTY_DEFINITION_REPRESENTATION(#512,#515);
…
#522=PROPERTY_DEFINITION('mass when empty',$,#314);
#523=PROPERTY_DEFINITION_REPRESENTATION(#522,#525);
```

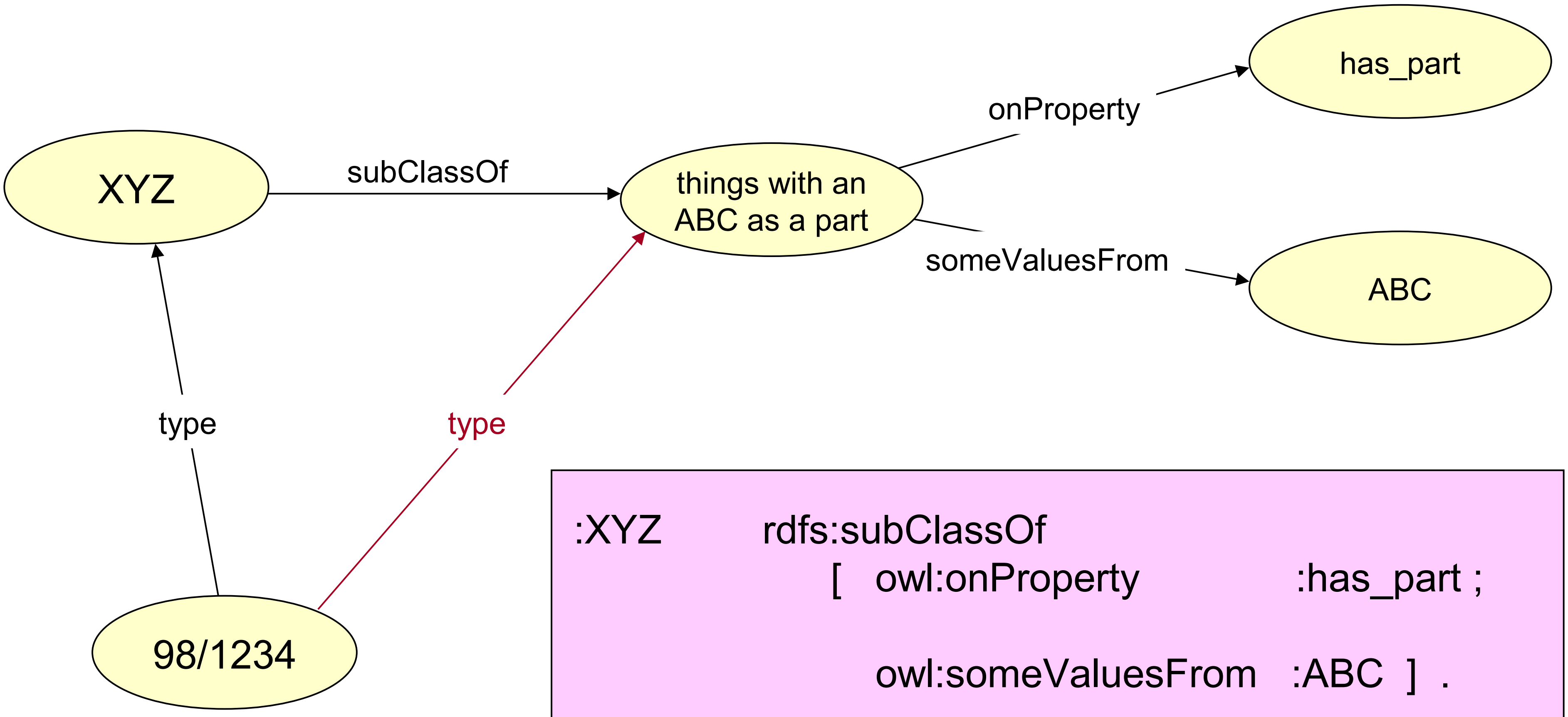
```
<step:Part_view_definition rdf:about="#_314">
  <owl:equivalentClass>
    <owl:Class>
       <owl:intersectionOf rdf:parseType=“Collection”>
         <owl:Class rdf:about="#_512"/>
         <owl:Class rdf:about="#_522"/>
       </owl:intersectionOf>
    </owl:Class>
  </owl:equivalentClass>
</step:Part_view_definition>
```

Mapping of Product / Part & _version

```
#308=PRODUCT_RELATED_PRODUCT_CATEGORY('part',$,(#310,...));
#310=PRODUCT('A0001','VW Beetle \X2\2013\X0\ P_101',' ',(#302));
#311=PRODUCT_DEFINITION_FORMATION('1',$,#310);
#312=PRODUCT_DEFINITION('1',$,#311,#303);
#314=PRODUCT_DEFINITION('2',$,#311,#303);
```

```
<step:Part rdf:ID="_310">
  <step:id rdf:datatype="&xsd;string">A0001</step:id>
  <step:name rdf:datatype="&xsd;string">VW Beetle / P_101</step:name>
</step:Part>

<step:Part_version rdf:ID="_311">
  <rdfs:subClassOf rdf:resource="#_310"/>
  <rdfs:subClassOf rdf:resource="#_312"/>
  <rdfs:subClassOf rdf:resource="#_314"/>
  <step:id rdf:datatype="&xsd;string">1</step:id>
</step:Part_version>
```

Deduction of relationship

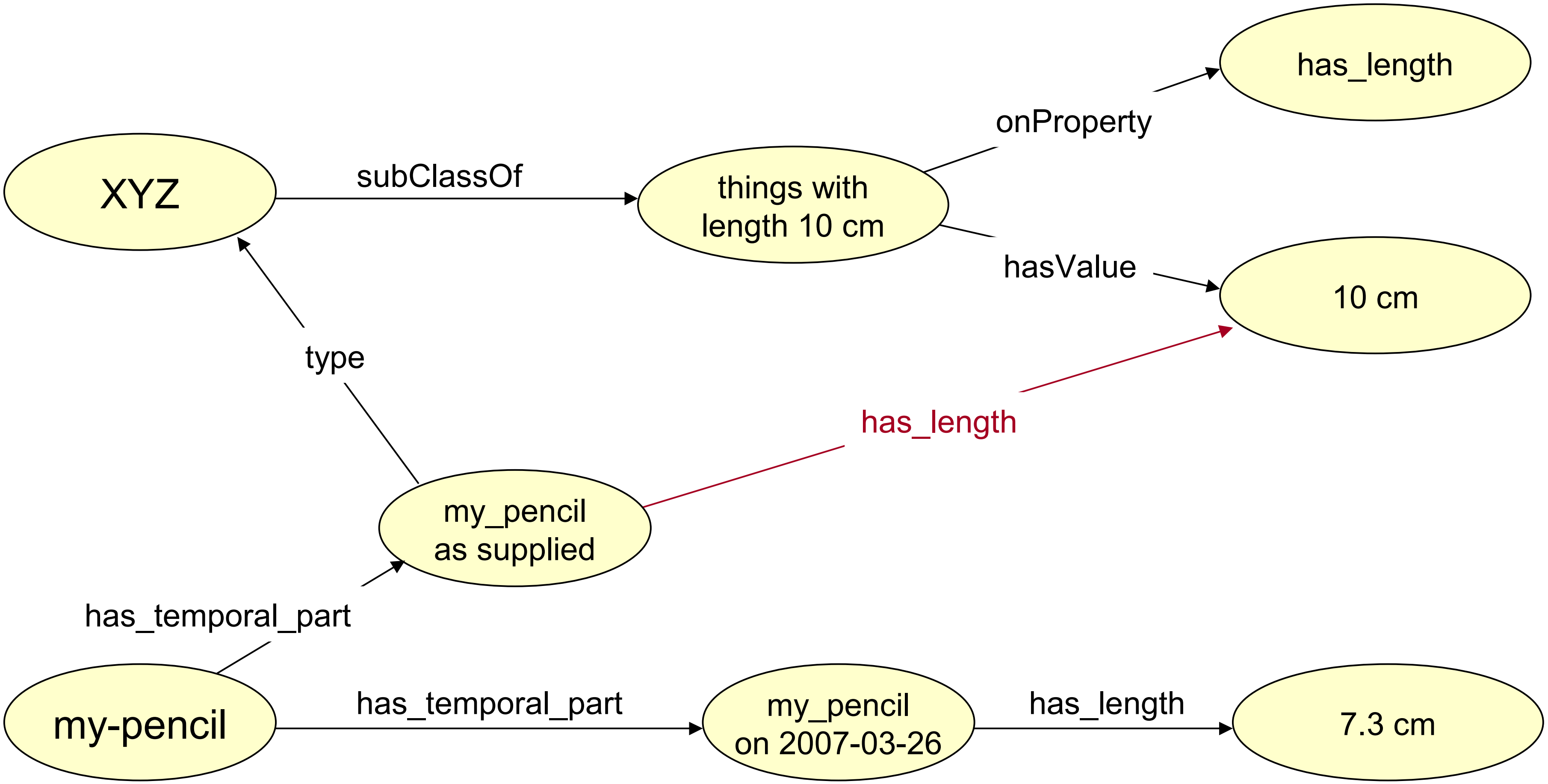


```
:XYZ        rdfs:subClassOf
                 [   owl:onProperty            :has_part ;

                     owl:someValuesFrom   :ABC  ]  .

:98/1234   a                           :XYZ  ;
            rdfs:subClassOf
                 [   owl:onProperty             :has_part  ;

                     owl:someValuesFrom   :ABC  ]  .
```

A temporal part

A pencil of type XYZ has a length of 10 cm as supplied.

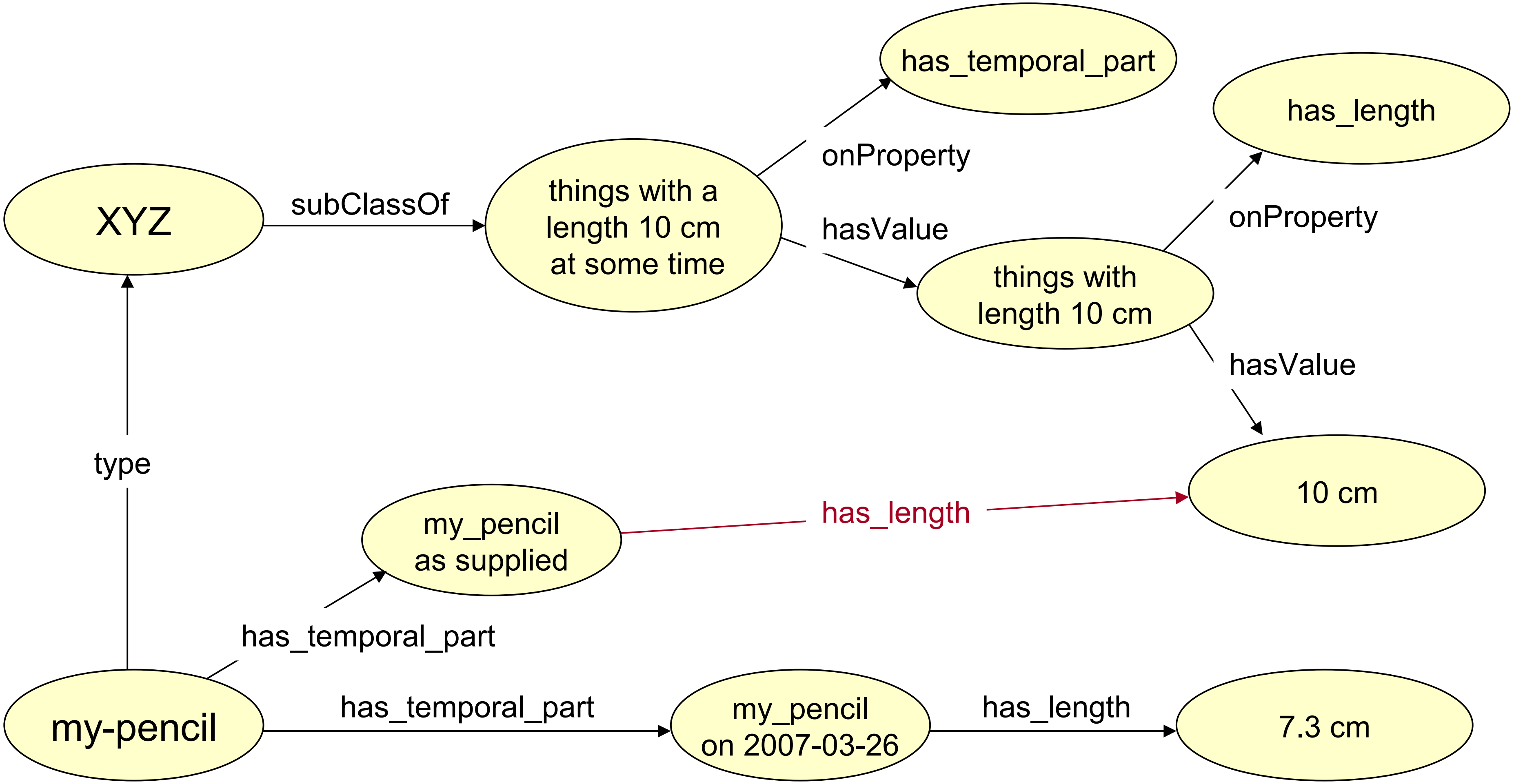
My_pencil is of type XYZ. It had a length of 10 cm when supplied.



A temporal part

A pencil of type XYZ has a length of 10 cm as supplied.

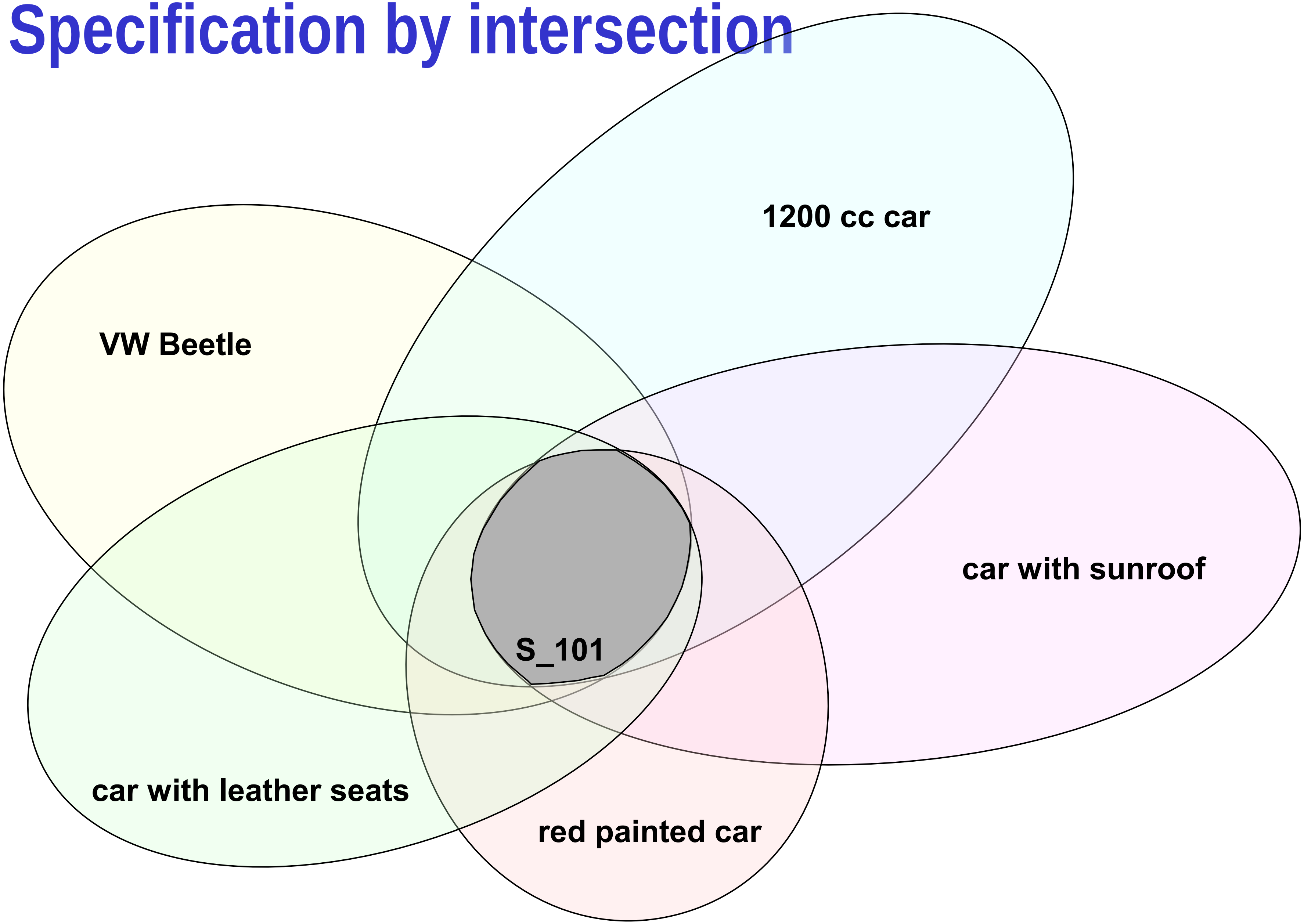
My_pencil is of type XYZ. It had a length of 10 cm when supplied.



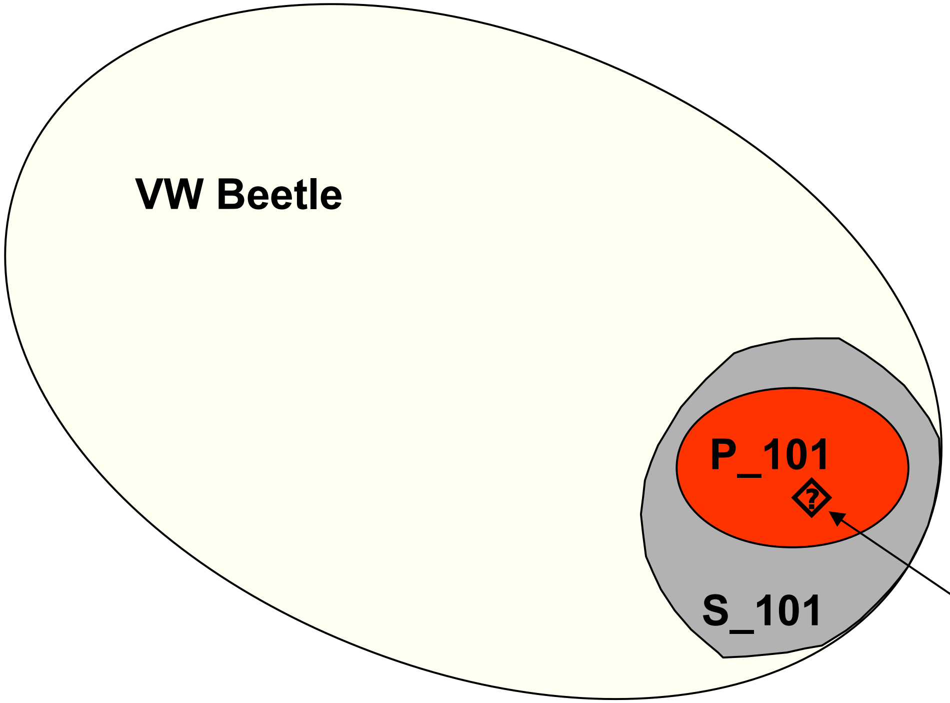
Entity and instance

entity	instance
product class	VW Beetle
product specification	VW Beetle – S_101 (1200cc, sunroof, red paint, leather seats)
part version	VW Beetle – P_101 (what VW produces in 2007 to meet the specification S_101)
product_as_individual	car with serial VW 07/12345678
class (of physical object, (defined by a business activity other than production)	Bighorn car rental – B_101 (compact, 2 door, 4 person, 2 luggage items)

Specification by intersection

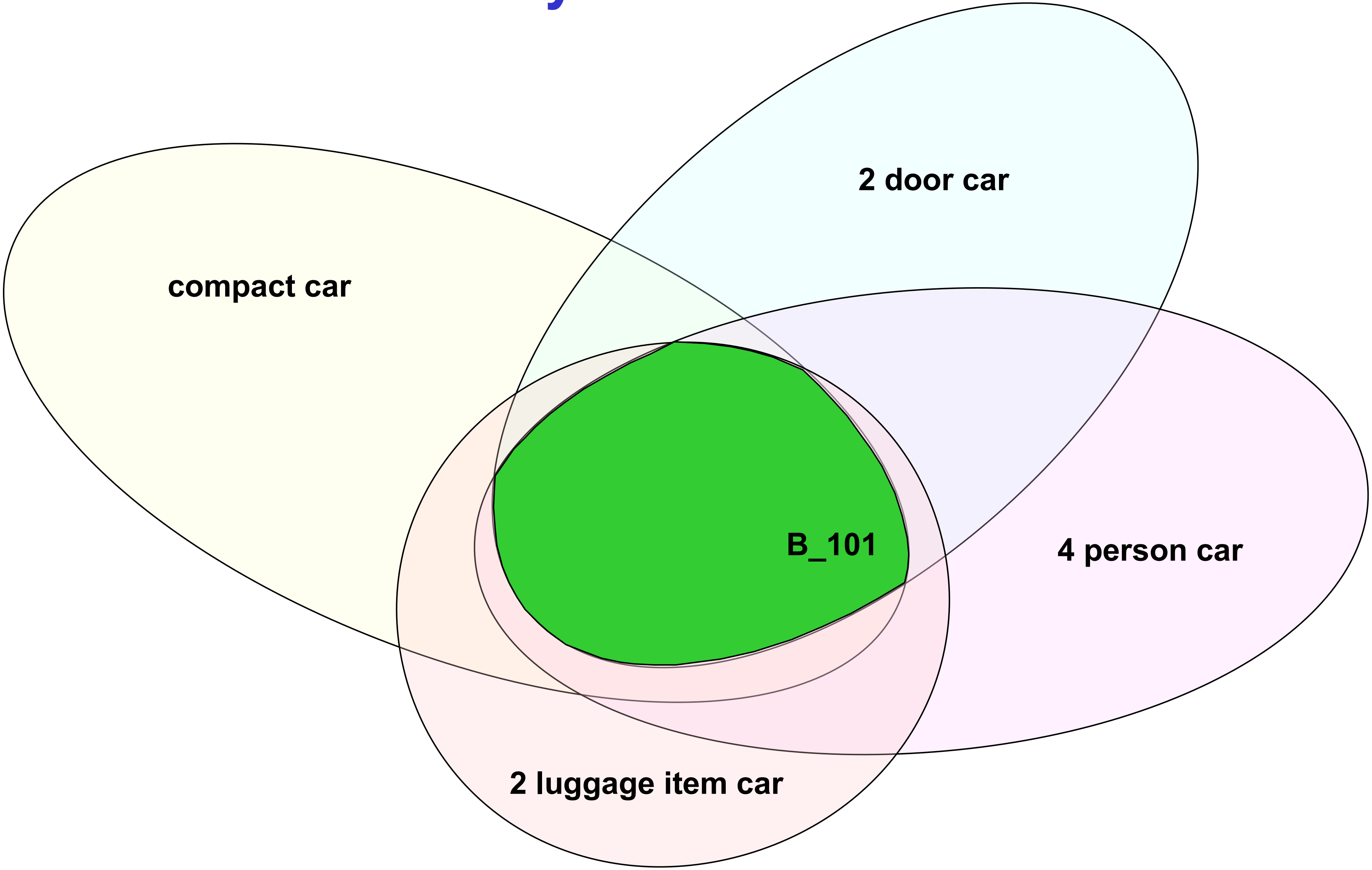


Part version satisfies specification

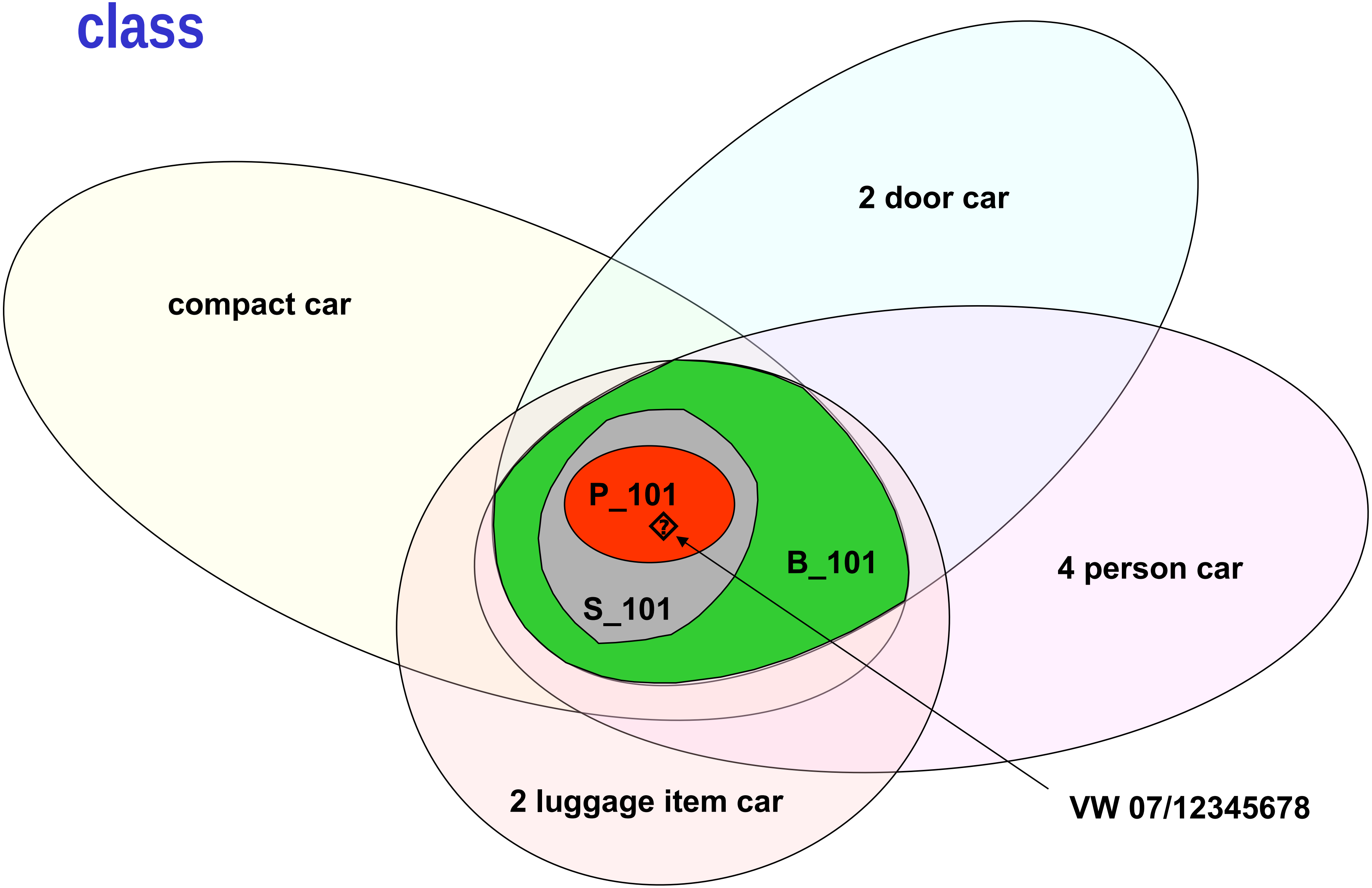


VW 07/12345678

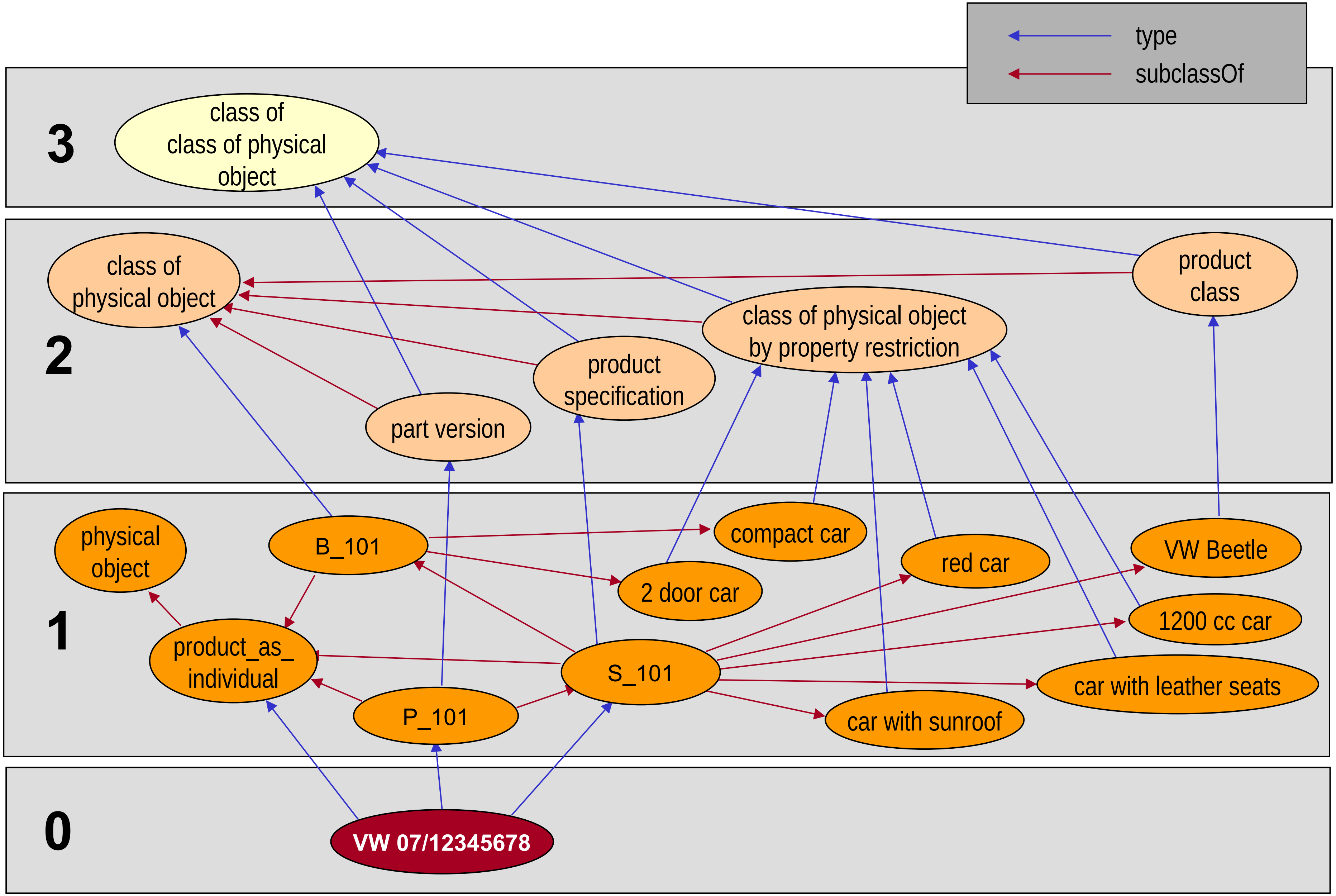
Business class by intersection



Specification satisfies business class



Meta-levels – motor car


type
subclassOf
3
class of class of physical object
2
class of physical object
product class
class of physical object by property restriction
product specification
part version
1
physical object
B_101
compact car
red car
VW Beetle
2 door car
1200 cc car
product_as_ individual
S_101
car with leather seats
P_101
car with sunroof
0
VW 07/12345678