Enriching Raw Data

John Beverley

Assistant Professor, *University at Buffalo*
Co-Director, *National Center for Ontological Research*
Affiliate Faculty, *Institute of Artificial Intelligence and Data Science*

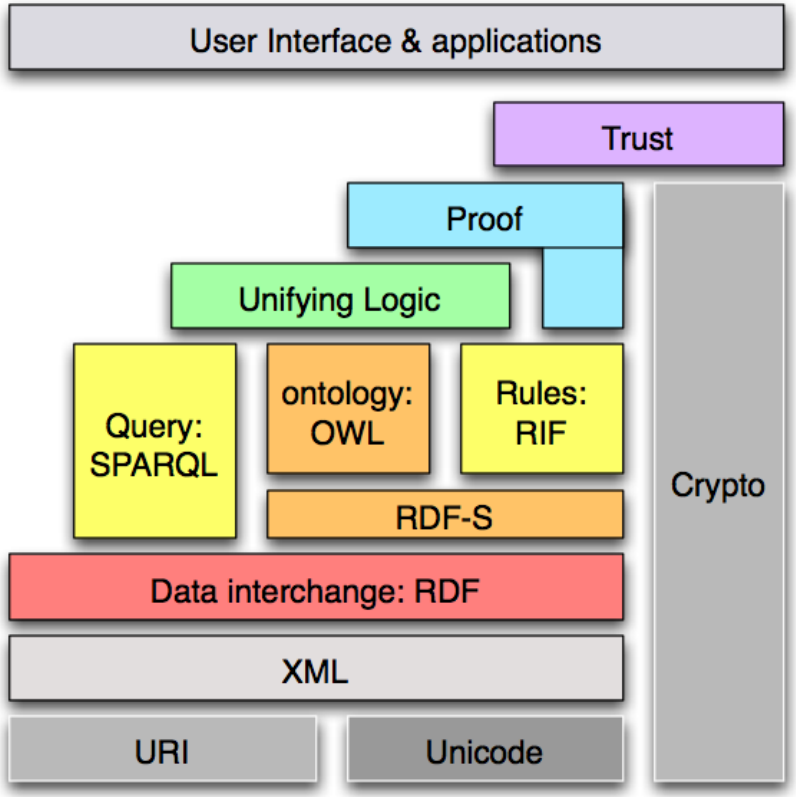
Outline

- Resource Description Framework (RDF)
- RDF Schema (RDFs)
- Modeling with Basic Formal Ontology
- Exercises

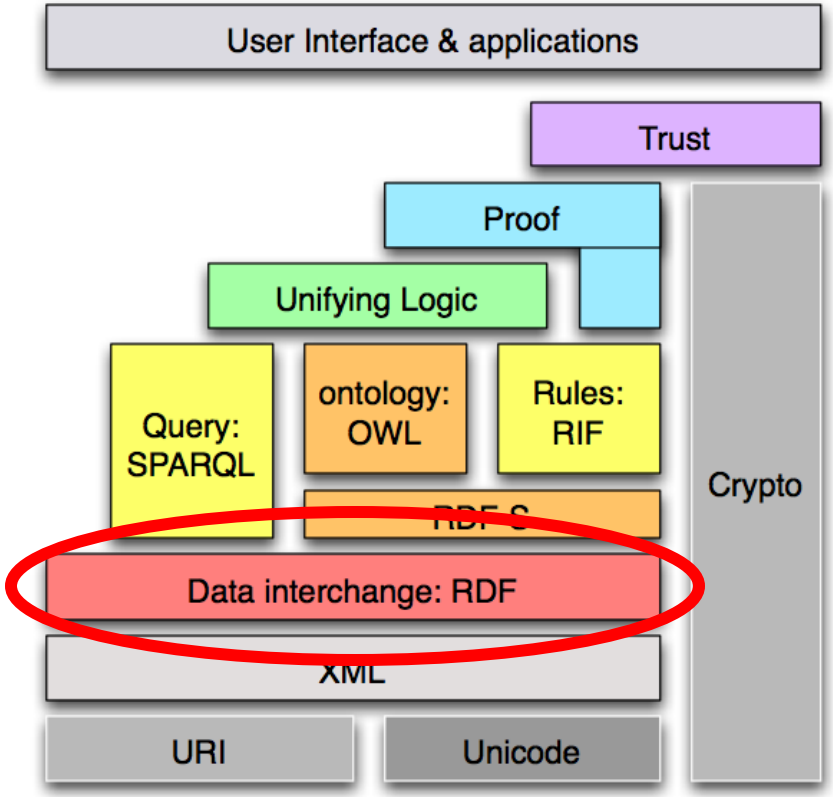
Outline

- Resource Description Framework (RDF)
- RDF Schema (RDFs)
- Modeling with Basic Formal Ontology
- Exercises

Semantic Web Stack

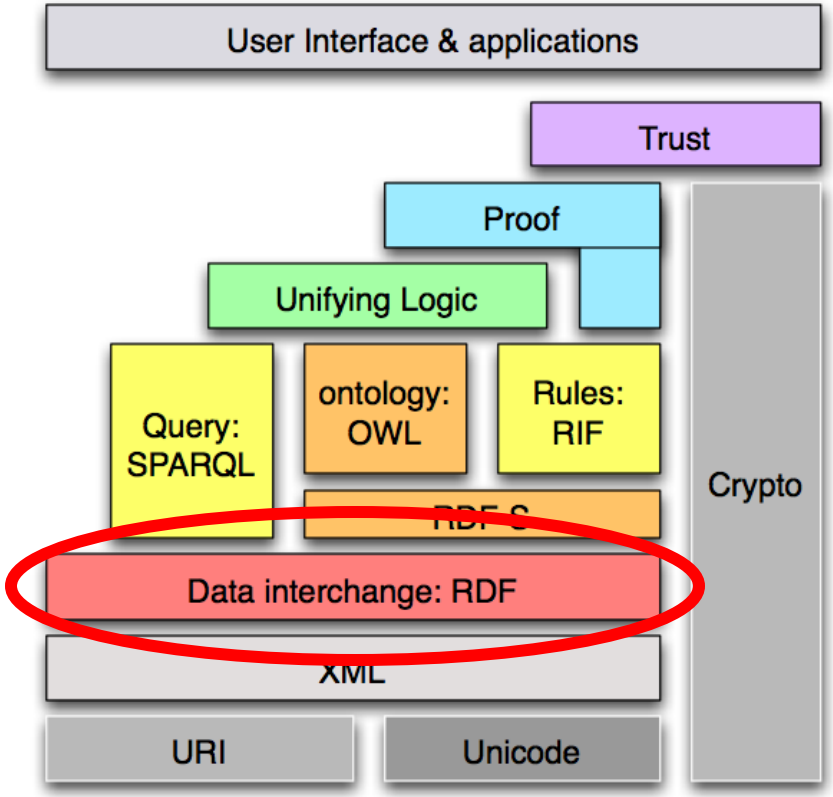


Semantic Web Stack



Semantic Web Stack

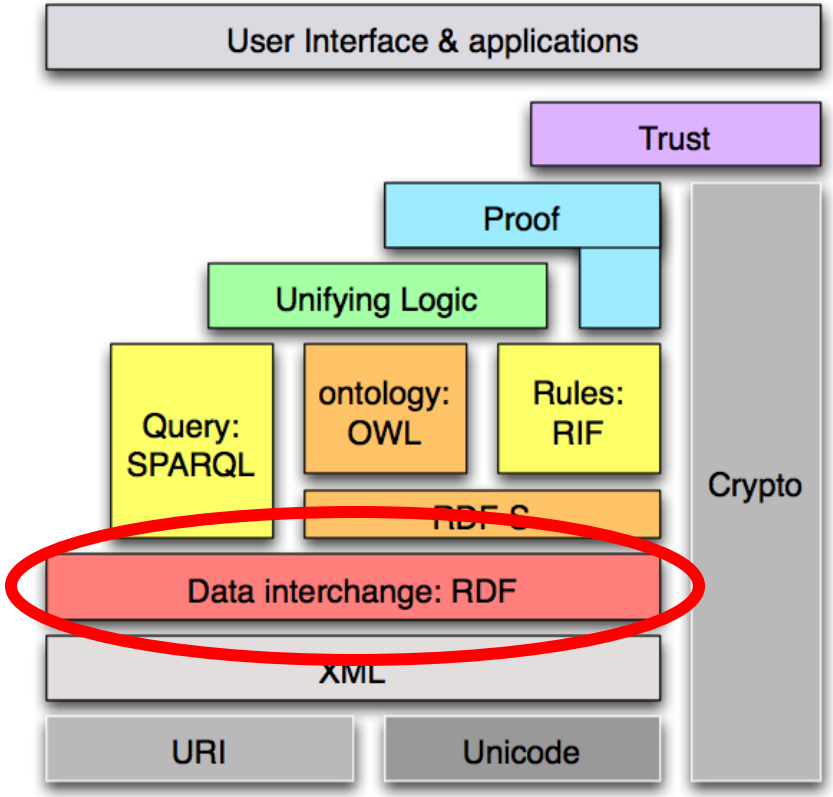
- "RDF" stands for:
  - **Resource**: Everything that can have a unique identifier, e.g. pages, places, people, dogs, products...
  - **Description**: attributes, features, and relations of among resources
  - **Framework**: model, languages and syntaxes for these descriptions



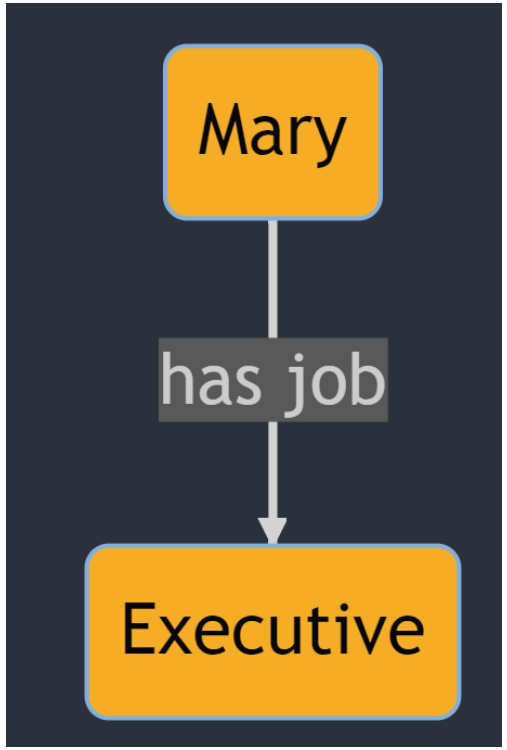
Semantic Web Stack

- "RDF" stands for:
  - **Resource**: Everything that can have a unique identifier, e.g. pages, places, people, dogs, products...
  - **Description**: attributes, features, and relations of among resources
  - **Framework**: model, languages and syntaxes for these descriptions

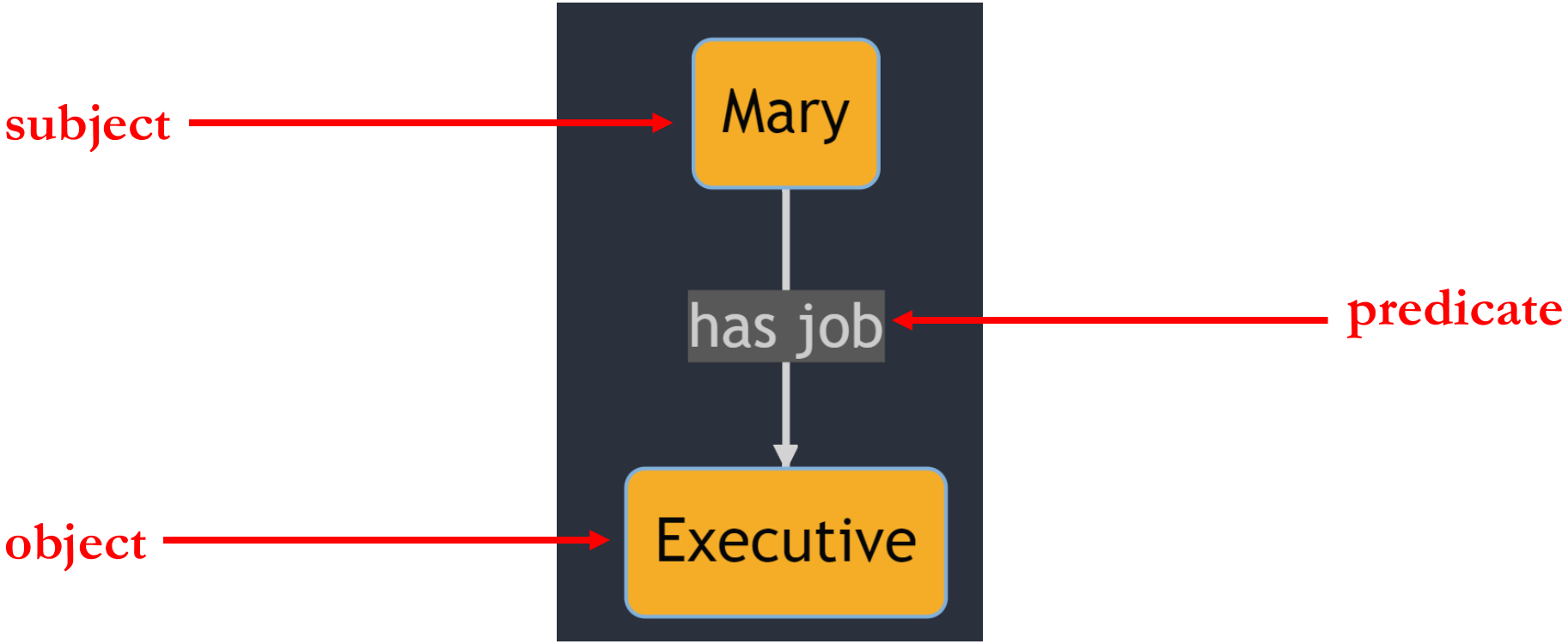
- RDF is:
  - A *data model*
  - That is based on *triples*
  - Which provide semantics for *distributed web data*



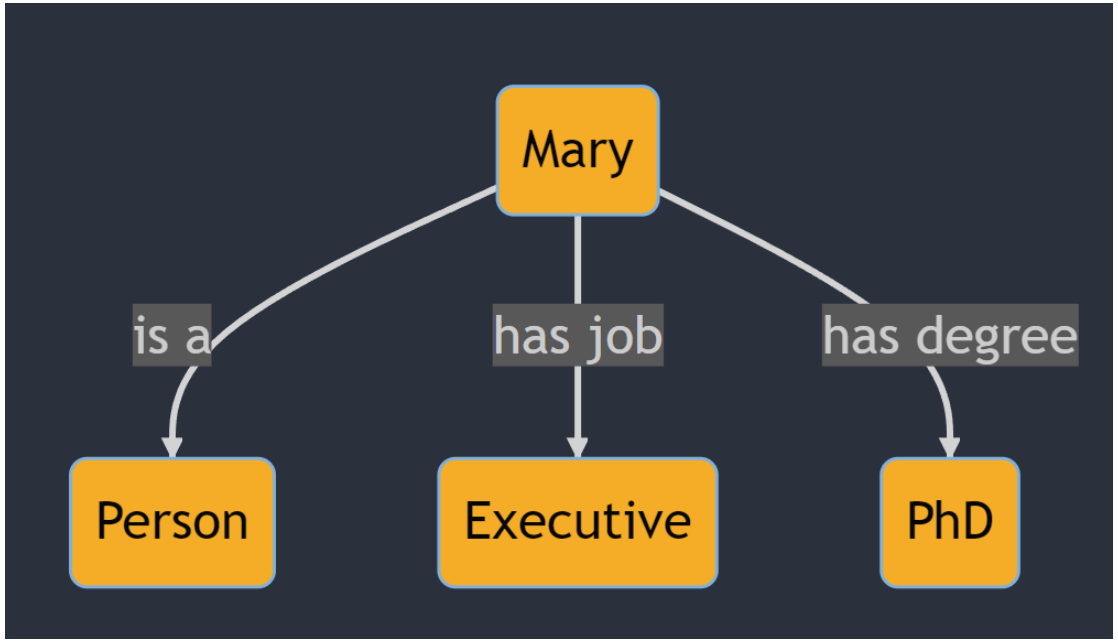
Triple Example



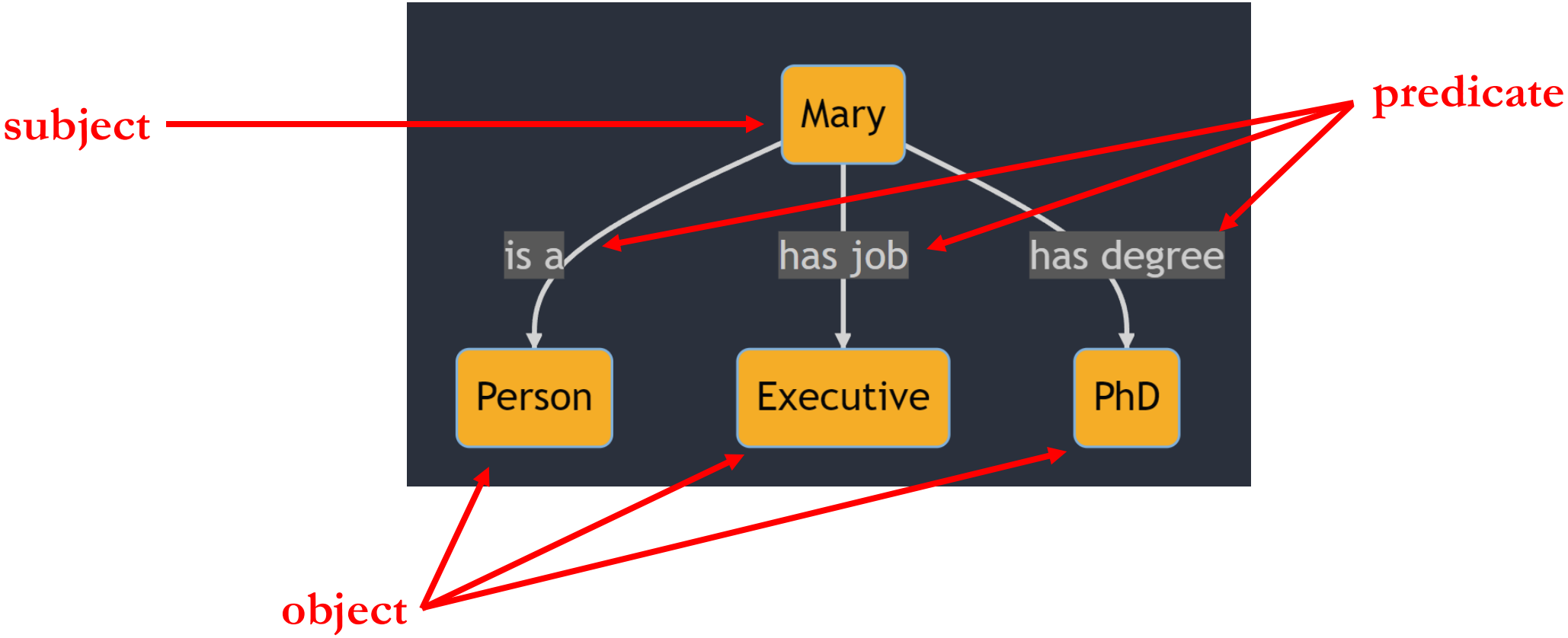
Triple Example



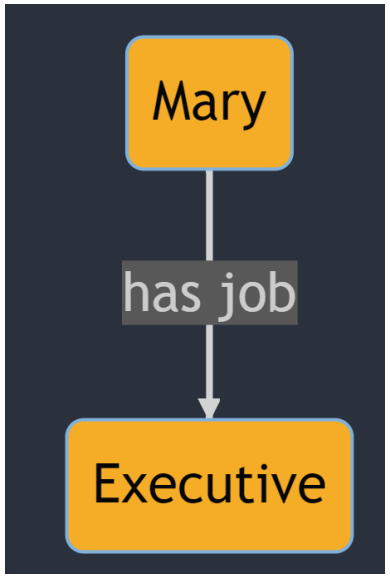
Triple Example



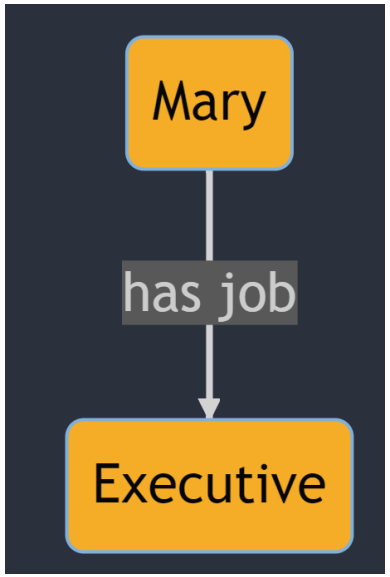
Triple Example



RDF Triple Structure



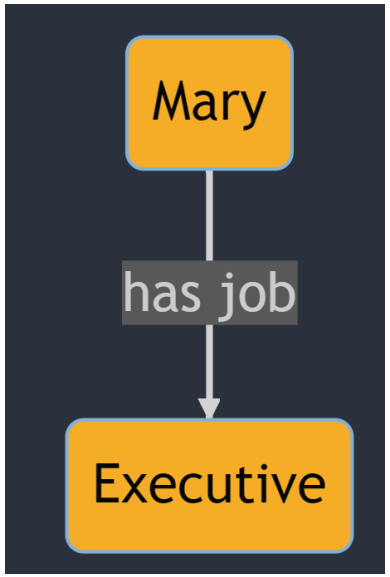
RDF Triple Structure



<https://www.example.com/occupation/Mary> <https://www.example.com/occupation/has_job> <https://www.example.com/occupation/Executive>

Subject Predicate Object

RDF Triple Structure



Uniform Resource Identifier (URI)

<https://www.example.com/occupation/Mary> <https://www.example.com/occupation/has_job> <https://www.example.com/occupation/Executive>

Subject

Predicate

Object

RDF Serializations

- Some RDF **serializations** are easier for humans to read than others, but many have been developed owing to many translational needs of users

- Common serializations you will encounter include:
  - RDF/XML
  - JSON-LD
  - Turtle
  - N-Triples

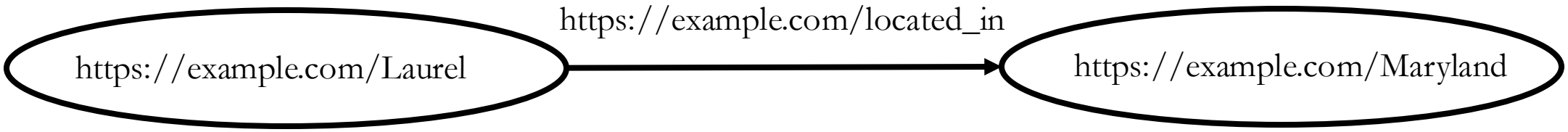
RDF Serializations

- Some RDF **serializations** are easier for humans to read than others, but many have been developed owing to many translational needs of users

- Common serializations you will encounter include:
  - RDF/XML
  - JSON-LD
  - Turtle
  - N-Triples

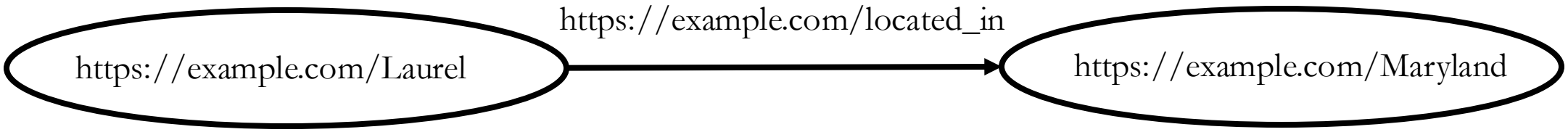
RDF/XML

- RDF/XML **serializations** express RDF as Extensible Markup Language (XML) documents, which is used widely across the web



RDF/XML

- RDF/XML **serializations** express RDF as Extensible Markup Language (XML) documents, which is used widely across the web




```
<?xml version="1.0"?>
<rdf:RDF   xmlsns:ex="https://example.com/"
           xmlsns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
<rdf:Description rdf:about="https://example.com/Laurel">
    <rdfs:label>Laurel</rdfs:label>
    <ex:located_in>Maryland</ex:located_in>
</rdf:Description>
</rdf:RDF>
```

RDF Serializations

- Some RDF **serializations** are easier for humans to read than others, but many have been developed owing to many translational needs of users

- Common serializations you will encounter include:
  - RDF/XML
  - JSON-LD
  - Turtle
  - N-Triples

JSON-LD

- JSON-LD is a recently developed serialization, but is now the preferred way to structure data in the Google knowledge graph, so...

JSON-LD

- JSON-LD is a recently developed serialization, but is now the preferred way to structure data in the Google knowledge graph, so...

[{"@id":"https://example.com/Laurel",
https://example.com/located_in
:[{"@id":"https://example.com/Maryland"}],
http://www.w3.org/2000/01/rdf-schema#label
:[{"@value":"Laurel"}]},{"@id":"https://example.com/Maryland"}]

RDF Serializations

- Some RDF **serializations** are easier for humans to read than others, but many have been developed owing to many translational needs of users

- Common serializations you will encounter include:
  - RDF/XML
  - JSON-LD
  - Turtle
  - N-Triples

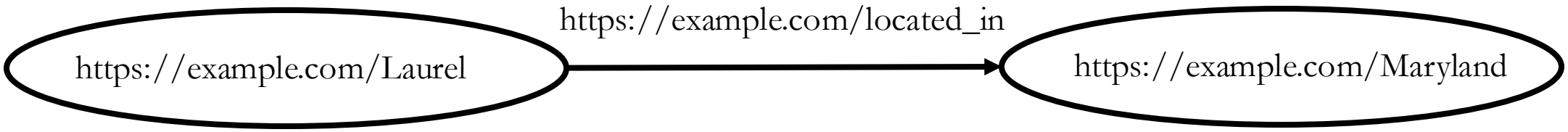
RDF Serializations

- Some RDF **serializations** are easier for humans to read than others, but many have been developed owing to many translational needs of users

- Common serializations you will encounter include:
  - RDF/XML
  - JSON-LD
  - Turtle
  - N-Triples

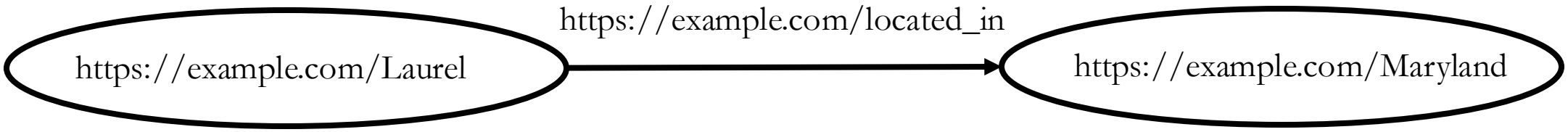
Turtle

- The Terse RDF Triple Language (Turtle) is a compact serialization of RDF that is, in my opinion, much easier to read



Turtle

- The Terse RDF Triple Language (Turtle) is a compact serialization of RDF that is, in my opinion, much easier to read




```
@prefix ex: <https://example.com/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

ex:Laurel rdfs:label "Laurel" ;
          ex:located_in ex:Maryland .
```

RDF Serializations

- Some RDF **serializations** are easier for humans to read than others, but many have been developed owing to many translational needs of users

- Common serializations you will encounter include:
  - RDF/XML
  - JSON-LD
  - Turtle
  - N-Triples

N-Triples

- The N-Triple serialization standard represents triples as unabbreviated URIs enclosed in angle brackets and strings in quotes

N-Triples

- The N-Triple serialization standard represents triples as unabbreviated URIs enclosed in angle brackets and strings in quotes

- What you see in a text file:

<https://example.com/Laurel> <https://example.com/located_in> <https://example.com/Maryland> .

<https://example.com/Laurel> <http://www.w3.org/2000/01/rdf-schema#label> “Laurel” .

Rules of RDF

- Every fact must be **expressed as a triple**
- Subjects, predicates, and objects **are names** for entities represented as URIs
- **Objects** may be literal values, but **subjects and predicates cannot**

RDF Vocabulary

- RDF is a very simple data model, with a very simple vocabulary:

rdf:Property	rdf:Bag
rdf:type	rdf:List
rdf:Statement	rdf:first
rdf:subject	rdf:rest
rdf:predicate	rdf:nil
rdf:object	

RDF Vocabulary

- RDF is a very simple data model, with a very simple vocabulary:

rdf:Property	rdf:Bag
rdf:type	rdf:List
rdf:Statement	rdf:first
rdf:subject	rdf:rest
rdf:predicate	rdf:nil
rdf:object	

The class of rdf properties, i.e. relationships among rdf resources

RDF Vocabulary

- RDF is a very simple data model, with a very simple vocabulary:

rdf:Property	rdf:Bag
rdf:type	rdf:List
rdf:Statement	rdf:first
rdf:subject	rdf:rest
rdf:predicate	rdf:nil
rdf:object	

Instance of rdf:Property used to state that a resource is an instance of another resource

RDF Vocabulary

- RDF is a very simple data model, with a very simple vocabulary:

rdf:Property
rdf:type
rdf:Statement
rdf:subject
rdf:predicate
rdf:object

rdf:Bag
rdf:List
rdf:first
rdf:rest
rdf:nil

The class of statements made about rdf triples

RDF Vocabulary

- RDF is a very simple data model, with a very simple vocabulary:

rdf:Property	rdf:Bag
rdf:type	rdf:List
rdf:Statement	rdf:first
rdf:subject	rdf:rest
rdf:predicate	rdf:nil
rdf:object	

Used often for *reification*

Reification

- To reify in RDF is to make a statement about a statement

- For example, Sam might introduce a triple to an ontology:

  ex:butter_potato_1 rdf:type ex:Potato

Reification

- To reify in RDF is to make a statement about a statement

- For example, Sam might introduce a triple to an ontology:

  ex:butter_potato_1 rdf:type ex:Potato

- But wants to make sure this is because Gordan Ramsey says so...

- Sam wants to represent:

  ex:Ramsey ex:believes {ex:butter_potato_1 rdf:type ex:Potato}

Reification

- To reify in RDF is to make a statement about a statement

  ex:Ramsey ex:believes {ex:butter_potato_1 rdf:type ex:Potato}

- Represented in RDF as:

```
ex:statement_1 [ rdf:type rdf:Statement;
                 rdf:subject ex:butter_potato_1 ;
                 rdf:predicate rdf:type ;
                 rdf:object ex:Potato ] .
```

Reification

- To reify in RDF is to make a statement about a statement

  ex:Ramsey ex:believes {ex:butter_potato_1 rdf:type ex:Potato}

- Represented in RDF as:

```
ex:statement_1 [ rdf:type rdf:Statement;
                 rdf:subject ex:butter_potato_1 ;
                 rdf:predicate rdf:type ;
                 rdf:object ex:Potato ] .
```

Reification

- To reify in RDF is to make a statement about a statement

  ex:Ramsey ex:believes {ex:butter_potato_1 rdf:type ex:Potato}

- Represented in RDF as:

```
ex:statement_1 [ rdf:type rdf:Statement;
                 rdf:subject ex:butter_potato_1 ;
                 rdf:predicate rdf:type ;
                 rdf:object ex:Potato ] .

ex:Ramsey ex:believes ex:statement_1
```

Reification

- To reify in RDF is to make a statement about a statement

  ex:Ramsey ex:believes {ex:butter_potato_1 rdf:type ex:Potato}

- Represented in RDF as:

```
ex:statement_1 [ rdf:type rdf:Statement;
                 rdf:subject ex:butter_potato_1 ;
                 rdf:predicate rdf:type ;
                 rdf:object ex:Potato ] .
```

ex:Ramsey ex:believes ex:statement_1

RDF Vocabulary

- RDF is a very simple data model, with a very simple vocabulary:

rdf:Property	rdf:Bag
rdf:type	rdf:List
rdf:Statement	rdf:first
rdf:subject	rdf:rest
rdf:predicate	rdf:nil
rdf:object	

rdf resource used to conventionally indicate to readers that collected members are unordered, e.g. John teaches group of students

RDF Vocabulary

- RDF is a very simple data model, with a very simple vocabulary:

rdf:Property	rdf:Bag
rdf:type	rdf:List
rdf:Statement	rdf:first
rdf:subject	rdf:rest
rdf:predicate	rdf:nil
rdf:object	

class of rdf lists

RDF Vocabulary

- RDF is a very simple data model, with a very simple vocabulary:

rdf:Property	rdf:Bag
rdf:type	rdf:List
rdf:Statement	rdf:first
rdf:subject	rdf:rest
rdf:predicate	rdf:nil
rdf:object	

Distinguishes first element of list from rest, and from empty list, e.g.

RDF Vocabulary

- RDF is a very simple data model, with a very simple vocabulary:

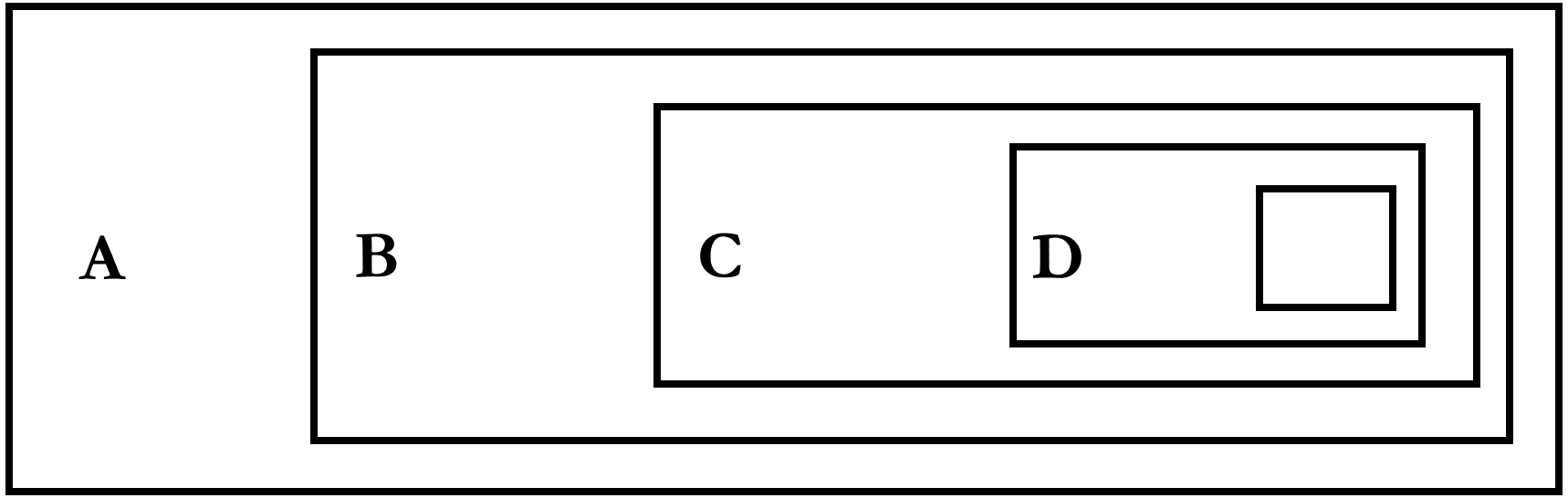
rdf:Property	rdf:Bag
rdf:type	rdf:List
rdf:Statement	rdf:first
rdf:subject	rdf:rest
rdf:predicate	rdf:nil
rdf:object	

Distinguishes first element of list from rest, and from empty list

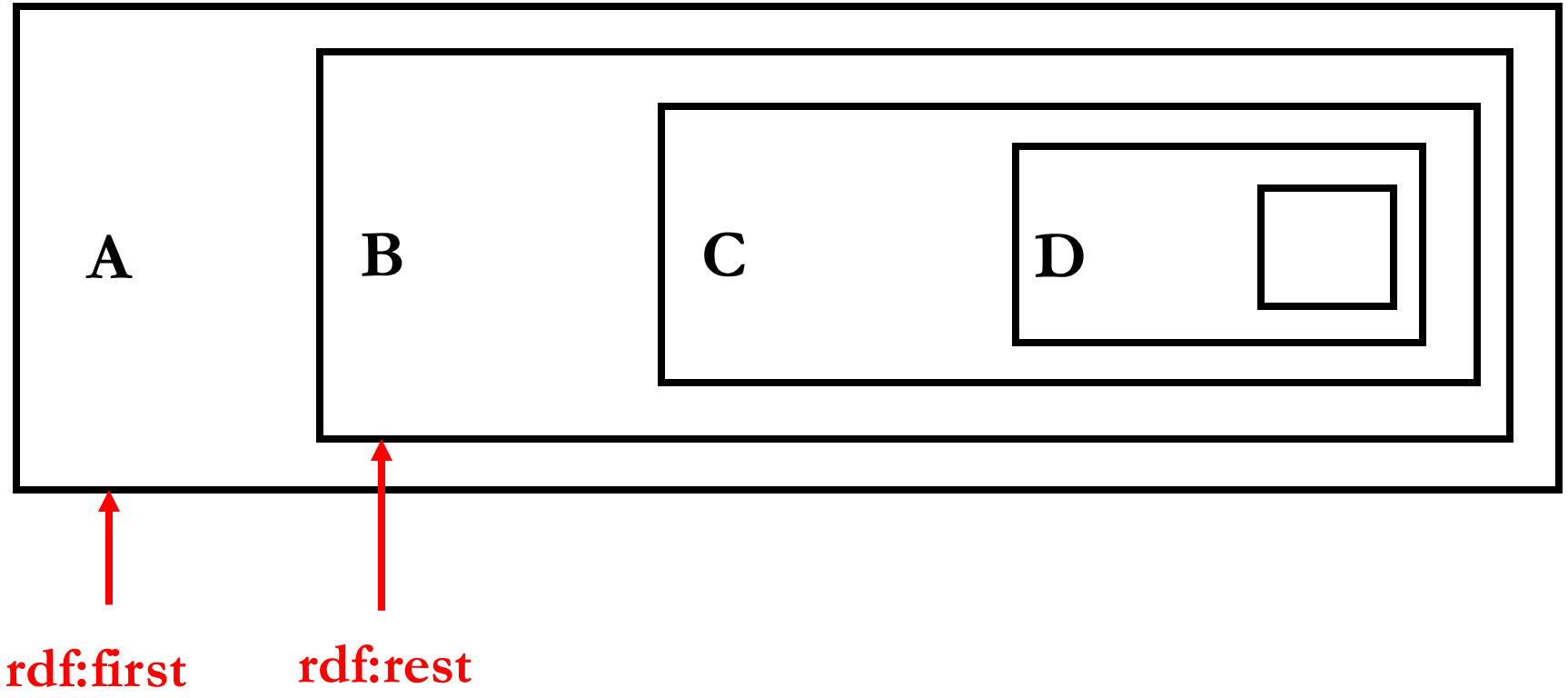
List = A,B,C,D

A **B** **C** **D**

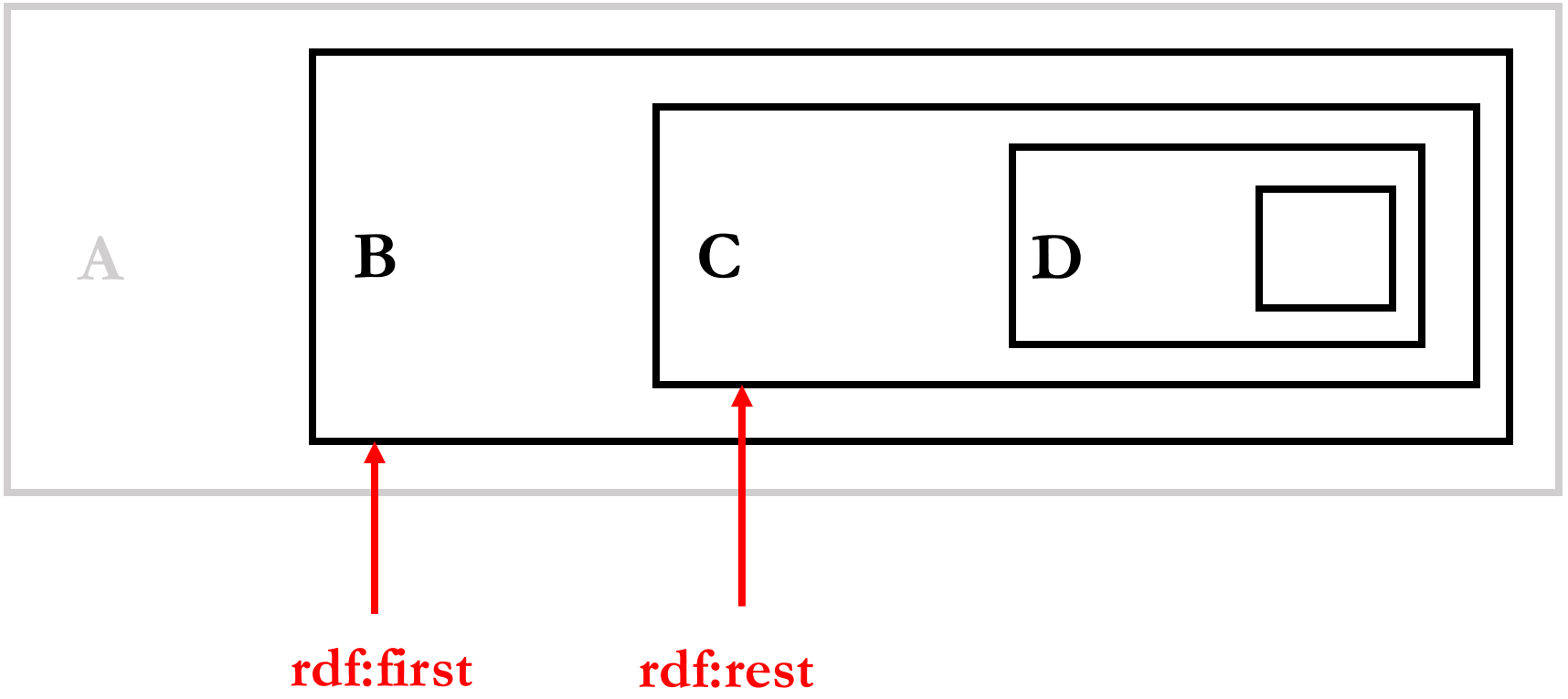
List = A,B,C,D



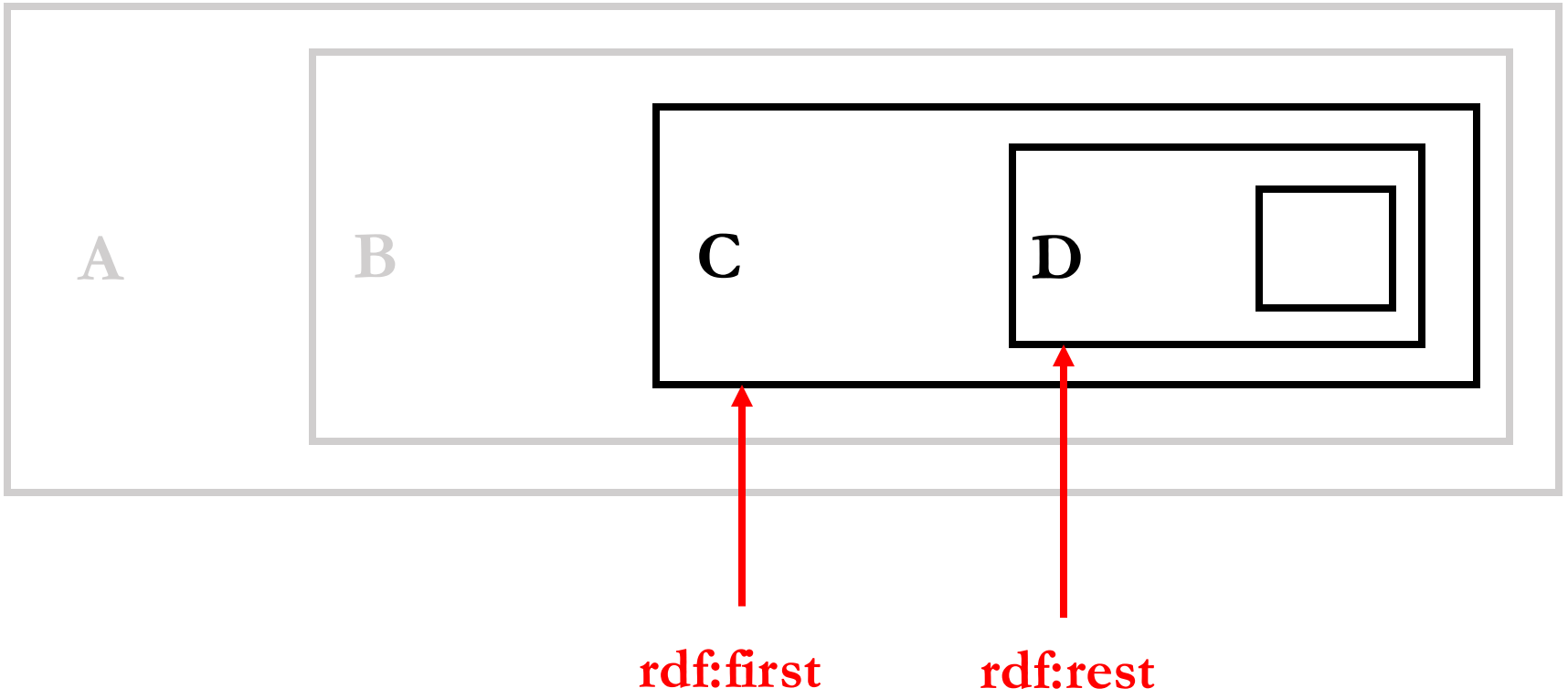
List = A,B,C,D



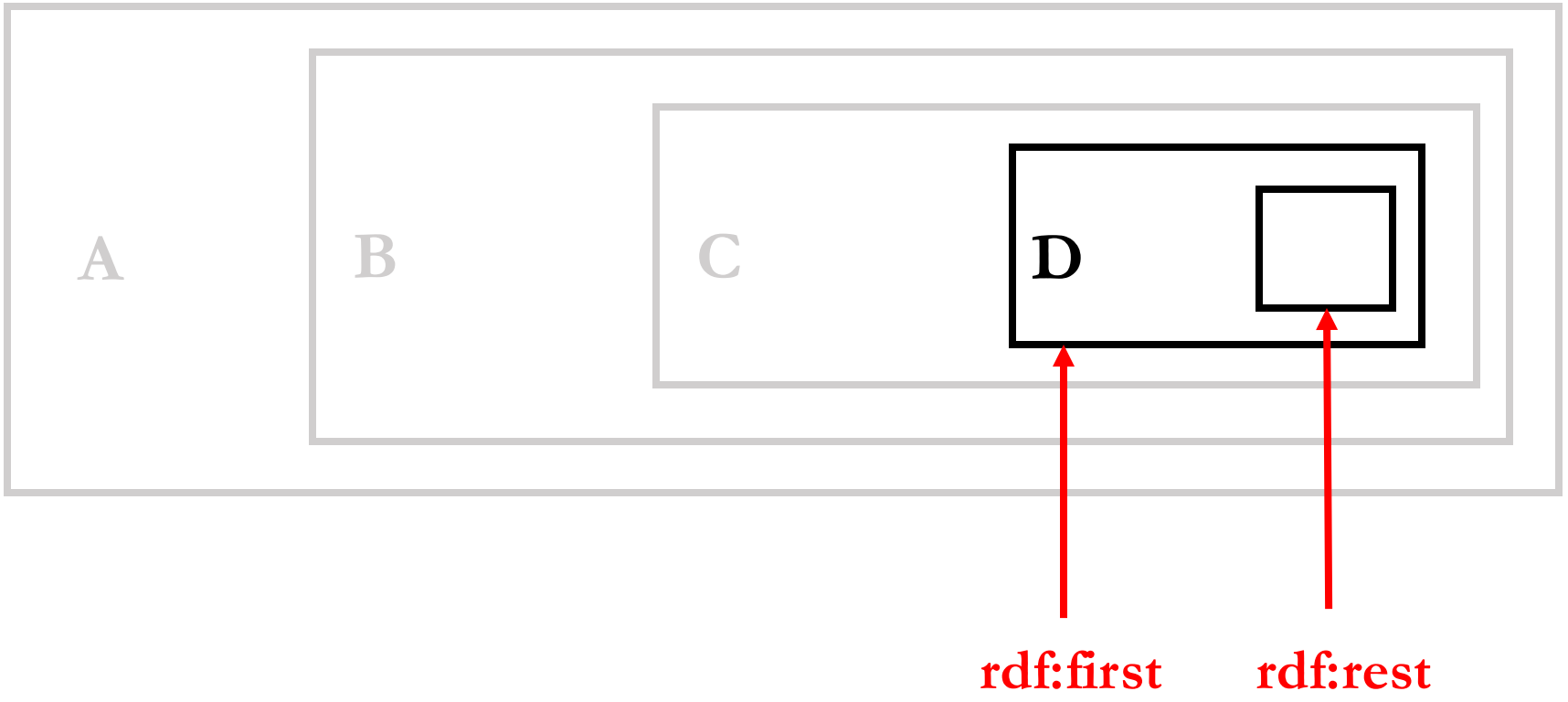
List = A,B,C,D



List = A,B,C,D



List = A,B,C,D



List = A,B,C,D

List Example

- New England Patriots are a better football team than the Premier League football team Arsenal, which is better than the Atlanta Braves...

- The ordering involved can be represented in rdf as:

List Example

- New England Patriots are a better football team than the Premier League football team Arsenal, which is better than the Atlanta Braves...

- The ordering involved can be represented in rdf as:

```
ex:list ex:football_rank [ rdf:first ex:Patriots ;
                           rdf:rest [ rdf:first ex:Arsenal ;
                                      rdf:rest [ rdf:first ex:Braves ;
                                                 rdf:rest rdf:nil ]]].
```

Subject is a list

List Example

- New England Patriots are a better football team than the Premier League football team Arsenal, which is better than the Atlanta Braves...

- The ordering involved can be represented in rdf as:

```
ex:list ex:football_rank [ rdf:first ex:Patriots ;
                           rdf:rest [ rdf:first ex:Arsenal ;
                                      rdf:rest [ rdf:first ex:Braves ;
                                                 rdf:rest rdf:nil ]]].
```

Predicate implies ordering

List Example

- New England Patriots are a better football team than the Premier League football team Arsenal, which is better than the Atlanta Braves...

- The ordering involved can be represented in rdf as:

```
ex:list ex:football_rank [ rdf:first ex:Patriots ;
                           rdf:rest [ rdf:first ex:Arsenal ;
                                      rdf:rest [ rdf:first ex:Braves ;
                                                 rdf:rest rdf:nil ]]].
```

First element of object list is Patriots

List Example

- New England Patriots are a better football team than the Premier League football team Arsenal, which is better than the Atlanta Braves...

- The ordering involved can be represented in rdf as:

```
ex:list ex:football_rank [ rdf:first ex:Patriots ;
                           rdf:rest [ rdf:first ex:Arsenal ;
                                      rdf:rest [ rdf:first ex:Braves ;
                                                 rdf:rest rdf:nil ]]].
```

Remainder of list is second

List Example

- New England Patriots are a better football team than the Premier League football team Arsenal, which is better than the Atlanta Braves...

- The ordering involved can be represented in rdf as:

```
ex:list ex:football_rank [ rdf:first ex:Patriots ;
                           rdf:rest [ rdf:first ex:Arsenal ;
                                      rdf:rest [ rdf:first ex:Braves ;
                                                 rdf:rest rdf:nil ]]].
```

First element of the second list is Arsenal

List Example

- New England Patriots are a better football team than the Premier League football team Arsenal, which is better than the Atlanta Braves...

- The ordering involved can be represented in rdf as:

```
ex:list ex:football_rank [ rdf:first ex:Patriots ;
                           rdf:rest [ rdf:first ex:Arsenal ;
                                      rdf:rest [ rdf:first ex:Braves ;
                                                 rdf:rest rdf:nil ]]].
```

Remainder of list is third

List Example

- New England Patriots are a better football team than the Premier League football team Arsenal, which is better than the Atlanta Braves...

- The ordering involved can be represented in rdf as:

```
ex:list ex:football_rank [ rdf:first ex:Patriots ;
                           rdf:rest [ rdf:first ex:Arsenal ;
                                      rdf:rest [ rdf:first ex:Braves ;
                                                 rdf:rest rdf:nil ]]].
```

First element of this list is Braves

List Example

- New England Patriots are a better football team than the Premier League football team Arsenal, which is better than the Atlanta Braves...

- The ordering involved can be represented in rdf as:

```
ex:list ex:football_rank [ rdf:first ex:Patriots ;
                           rdf:rest [ rdf:first ex:Arsenal ;
                                      rdf:rest [ rdf:first ex:Braves ;
                                                 rdf:rest rdf:nil ]]].
```

Remainder of list is fourth → rdf:rest rdf:nil

List Example

- New England Patriots are a better football team than the Premier League football team Arsenal, which is better than the Atlanta Braves...

- The ordering involved can be represented in rdf as:

```
ex:list ex:football_rank [ rdf:first ex:Patriots ;
                           rdf:rest [ rdf:first ex:Arsenal ;
                                      rdf:rest [ rdf:first ex:Braves ;
                                                 rdf:rest rdf:nil ]]].
```

Only element of fourth list is empty

RDF Vocabulary

- RDF is a very simple data model, with a very simple vocabulary:

rdf:Property	rdf:Bag
rdf:type	rdf:List
rdf:Statement	rdf:first
rdf:subject	rdf:rest
rdf:predicate	rdf:nil
rdf:object	

There are more terms in the RDF vocabulary, some of which can be leveraged for rather expressive representations; those mentioned here are what you'll most often encounted though

RDF Vocabulary

- RDF is a very simple data model, with a very simple vocabulary:

rdf:Property	rdf:Bag
rdf:type	rdf:List
rdf:Statement	rdf:first
rdf:subject	rdf:rest
rdf:predicate	rdf:nil
rdf:object	

But before turning to extensions of RDF, one further topic deserves our attention…

Blank Nodes

- Thus far, we've seen how RDF allows one to identify resources across the Web with unique URIs...

- But RDF also allows for the use of resources that *don't have any specific unique identifier*

- This is useful because there are many cases in which one wants to represent that, say, some x is related to a specific entity, without being able to identify that x

Blank Nodes

- Thus far, we've seen how RDF allows one to identify resources across the Web with unique URIs...

- But RDF also allows for the use of resources that *don't have any specific unique identifier*

That is, sometimes we want to represent *existentially quantified* relata in our ontologies

Blank Nodes Example

- Suppose you know that Barry plays the violin and you'd like to represent this with an object property...but don't know which violin he plays...

ex:Barry ex:plays ??? .

Blank Nodes Example

- Suppose you know that Barry plays the violin and you'd like to represent this with an object property...but don't know which violin he plays...

ex:Barry ex:plays **[rdf:type ex:Violin]** .

Blank Nodes Example

- Suppose you know that Barry plays the violin and you'd like to represent this with an object property...but don't know which violin he plays...

ex:Barry ex:plays **[rdf:type ex:Violin]** .

- Which simply says Barry plays something that is of the type Violin

Blank Nodes Example

- Suppose you know that Barry plays the violin and you'd like to represent this with an object property...but don't know which violin he plays...

ex:Barry ex:plays **[rdf:type ex:Violin]** .

Note, there is an implicit numerical id being used here; this implicit id allows other triples in the graph to refer to the same violin Barry plays

Blank Nodes Example

- Suppose you know that Barry plays the violin and you'd like to represent this with an object property...but don't know which violin he plays...

ex:Barry ex:plays **_:0001** .

_0001 rdf:type ex:Violin .

Blank Nodes Example

- Suppose you know that Barry plays the violin and you'd like to represent this with an object property...but don't know which violin he plays...

ex:Barry ex:plays **_:0001** .

_0001 rdf:type ex:Violin .

- Importantly, this numerical id is only *locally* unique, not globally unique like a URI

Blank Nodes Example

- Suppose you know that Barry plays the violin and you'd like to represent this with an object property...but don't know which violin he plays...

ex:Barry ex:plays **_:0001** .

_:0001 rdf:type ex:Violin .

- And any time you change the serialization of an RDF file – for example, from XML to Turtle – the blank node identifiers *change*

Blank Nodes across Serializations

```
<rdf:RDF xmlns:ex="https://example.com/"
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
 <rdf:Description rdf:about="https://example.com/Barry">
  <ex:plays rdf:nodeID="b1" />
 </rdf:Description>
 <rdf:Description rdf:nodeID="b1">
  <rdf:type rdf:resource="https://example.com/Violin" />
 </rdf:Description>
</rdf:RDF>
```

RDF/XML

```
{"@context": {
   "ex": "https://example.com/",
   "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#" },
  "@id": "https://example.com/Barry",
  "ex:plays": {
   "rdf:type": "https://example.com/Violin" }}
```

JSON-LD

```
<https://example.com/Barry> <https://example.com/plays> _:b1 .
_:b1 <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<https://example.com/Violin> .
```

N-TRIPLES

```
ex:Barry ex:plays [ rdf:type ex:Violin ] .
```

TURTLE

Blank Nodes across Serializations

```
<rdf:RDF xmlns:ex="https://example.com/"
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
 <rdf:Description rdf:about="https://example.com/Barry">
  <ex:plays rdf:nodeID="b1" />
 </rdf:Description>
 <rdf:Description rdf:nodeID="b1">
  <rdf:type rdf:resource="https://example.com/Violin" />
 </rdf:Description>
</rdf:RDF>
```

RDF/XML

```
{"@context": {
   "ex": "https://example.com/",
   "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#" },
  "@id": "https://example.com/Barry",
  "ex:plays": {
   "rdf:type": "https://example.com/Violin" }}
```

JSON-LD

```
<https://example.com/Barry> <https://example.com/plays> _:b1 .
_:b1 <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<https://example.com/Violin> .
```

N-TRIPLES

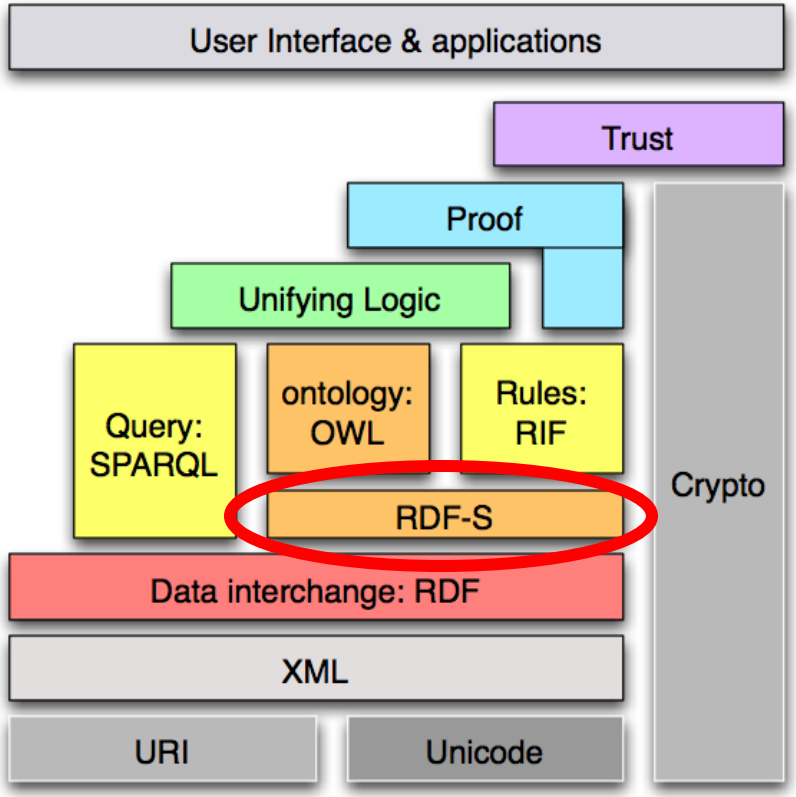
```
ex:Barry ex:plays [ rdf:type ex:Violin ] .
```

TURTLE

Outline

- Resource Description Framework (RDF)
- RDF Schema (RDFs)
- Modeling with Basic Formal Ontology
- Exercises

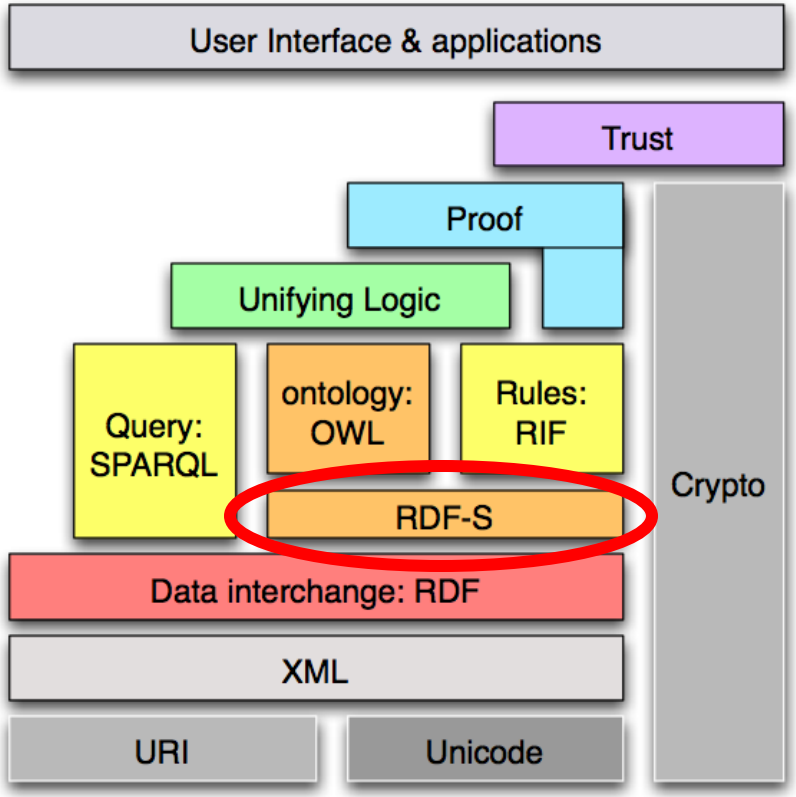
Semantic Web Stack



Semantic Web Stack

- The "S" in RDFS stands for:
  Schema – A syntax and semantics for an intended extension of the RDF syntax

- RDFS is:
  A *vocabulary*
  That is based on the *RDF data model*
  That extends the *RDF vocabulary*
  And provide semantics for *connecting RDF resources*



Resource Description Framework Schema

- RDFs is an extension of RDF facilitating representation of:

rdfs:Class
rdfs:subClassOf
rdfs:domain
rdfs:range
rdfs:subPropertyOf
rdfs:Resource
rdfs:Datatype
rdfs:Literal
rdfs:label
rdfs:comment
rdfs:isDefinedBy

Resource Description Framework Schema

- RDFs is an extension of RDF facilitating representation of:

rdfs:Class	rdfs:Datatype
rdfs:subClassOf	rdfs:Literal
rdfs:domain	rdfs:label
rdfs:range	rdfs:comment
rdfs:subPropertyOf	rdfs:isDefinedBy
rdfs:Resource	

We often say X rdf:type Y and intend Y to be a class, but strictly speaking there is no notion of "class" in rdf
rdfs:Class introduces the syntax needed for such assertions

Resource Description Framework Schema

- RDFs is an extension of RDF facilitating representation of:

rdfs:Class
rdfs:subClassOf
rdfs:domain
rdfs:range
rdfs:subPropertyOf
rdfs:Resource
rdfs:Datatype
rdfs:Literal
rdfs:label
rdfs:comment
rdfs:isDefinedBy

rdfs:subClassOf facilitates representing the "is a" hierarchy
e.g. bfo:object rdfs:subClassOf bfo:MaterialEntity

Resource Description Framework Schema

- RDFs is an extension of RDF facilitating representation of:

rdfs:Class
rdfs:subClassOf
rdfs:domain
rdfs:range
rdfs:subPropertyOf
rdfs:Resource
rdfs:Datatype
rdfs:Literal
rdfs:label
rdfs:comment
rdfs:isDefinedBy

For a binary relation R(x,y) the domain of R is whatever can occupy x and the range is whatever can occupy y

Resource Description Framework Schema

- RDFs is an extension of RDF facilitating representation of:

rdfs:Class
rdfs:subClassOf
rdfs:domain
rdfs:range
rdfs:subPropertyOf
rdfs:Resource
rdfs:Datatype
rdfs:Literal
rdfs:label
rdfs:comment
rdfs:isDefinedBy

Suppose we want to define a binary relation "has leaf".
Plausibly, the domain will be "plant" and the range will be "leaf"

Resource Description Framework Schema

- RDFs is an extension of RDF facilitating representation of:

rdfs:Class
rdfs:subClassOf
rdfs:domain
rdfs:range
rdfs:subPropertyOf
rdfs:Resource
rdfs:Datatype
rdfs:Literal
rdfs:label
rdfs:comment
rdfs:isDefinedBy

Much like we often find need to describe "is a" relationships between classes, so too we need to describe "is a" relationships between relations, e.g. x devours y is a sub-property of x eats y.

Resource Description Framework Schema

- RDFs is an extension of RDF facilitating representation of:

rdfs:Class	rdfs:Datatype
rdfs:subClassOf	rdfs:Literal
rdfs:domain	rdfs:label
rdfs:range	rdfs:comment
rdfs:subPropertyOf	rdfs:isDefinedBy
rdfs:Resource	

The class of everything

Any instance represented in RDF is an instance of rdfs:Resource

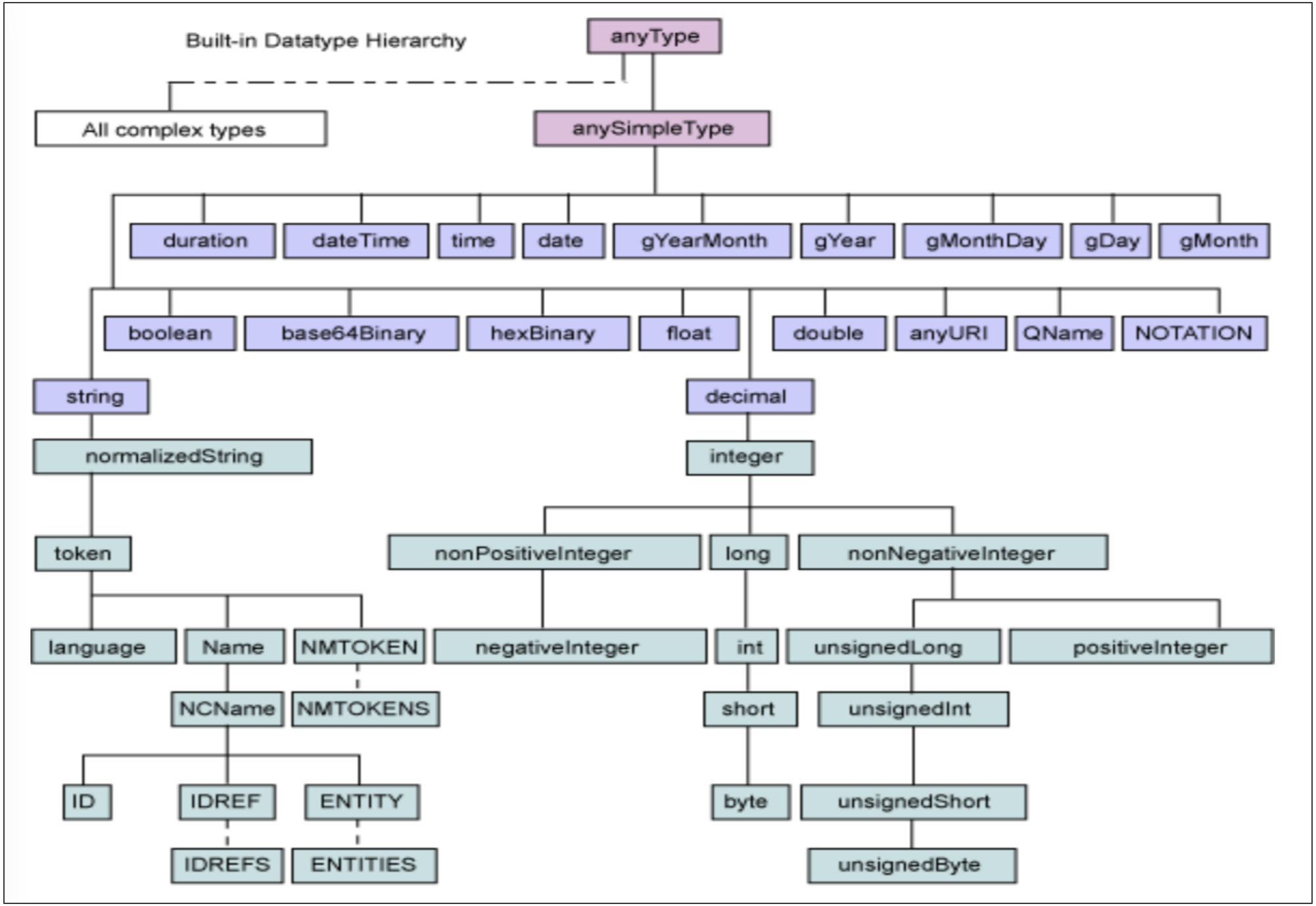
Any class is a subclass of rdfs:Resource

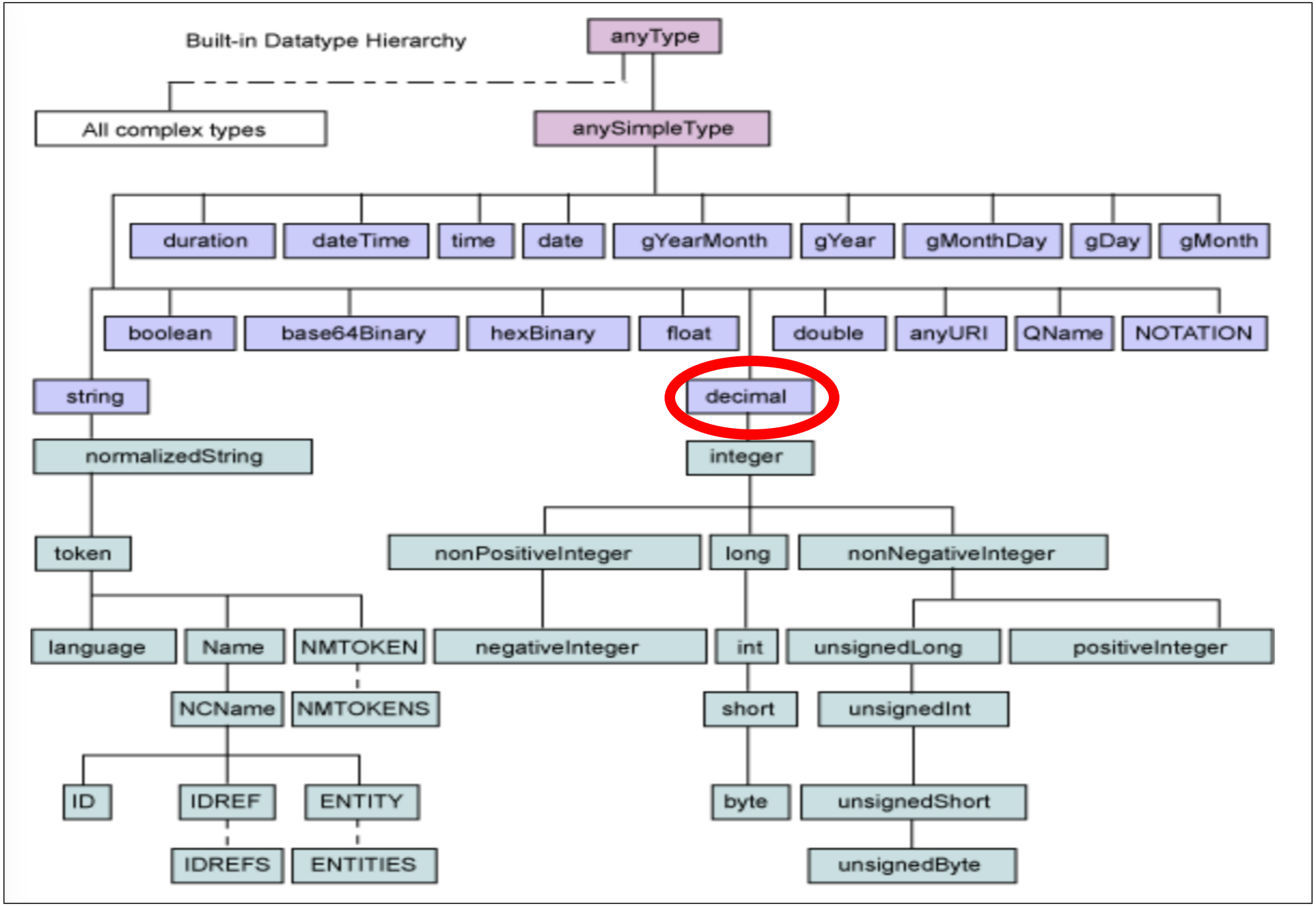
Resource Description Framework Schema

- RDFs is an extension of RDF facilitating representation of:

rdfs:Class	rdfs:Datatype
rdfs:subClassOf	rdfs:Literal
rdfs:domain	rdfs:label
rdfs:range	rdfs:comment
rdfs:subPropertyOf	rdfs:isDefinedBy
rdfs:Resource	

rdfs:Datatype is a subclass of rdfs:Literal and is the class of all datatypes, such as "integer" or "decimal"


Built-in Datatype Hierarchy
anyType
All complex types
anySimpleType
duration
dateTime
time
date
gYearMonth
gYear
gMonthDay
gDay
gMonth
boolean
base64Binary
hexBinary
float
double
anyURI
QName
NOTATION
string
decimal
normalizedString
integer
token
nonPositiveInteger
long
nonNegativeInteger
language
Name
NMTOKEN
negativeInteger
int
unsignedLong
positiveInteger
NCName
NMTOKENS
short
unsignedInt
ID
IDREF
ENTITY
byte
unsignedShort
IDREFS
ENTITIES
unsignedByte


Built-in Datatype Hierarchy
anyType
All complex types
anySimpleType
duration
dateTime
time
date
gYearMonth
gYear
gMonthDay
gDay
gMonth
boolean
base64Binary
hexBinary
float
double
anyURI
QName
NOTATION
string
decimal
normalizedString
integer
token
nonPositiveInteger
long
nonNegativeInteger
language
Name
NMTOKEN
negativeInteger
int
unsignedLong
positiveInteger
NCName
NMTOKENS
short
unsignedInt
ID
IDREF
ENTITY
byte
unsignedShort
IDREFS
ENTITIES
unsignedByte

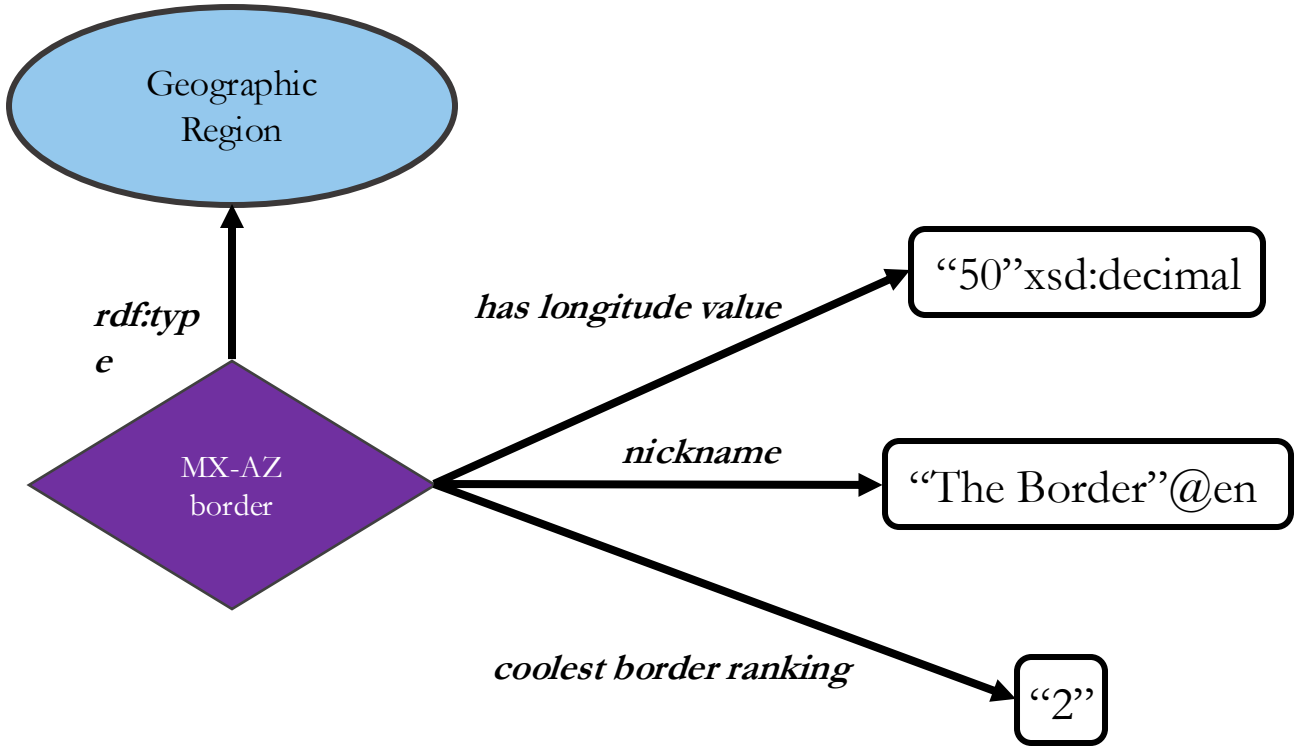
Resource Description Framework Schema

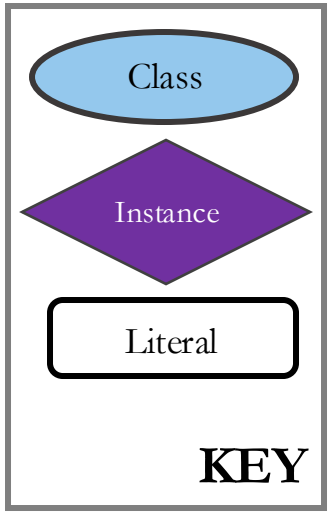
- RDFs is an extension of RDF facilitating representation of:

rdfs:Class
rdfs:subClassOf
rdfs:domain
rdfs:range
rdfs:subPropertyOf
rdfs:Resource
rdfs:Datatype
rdfs:Literal
rdfs:label
rdfs:comment
rdfs:isDefinedBy

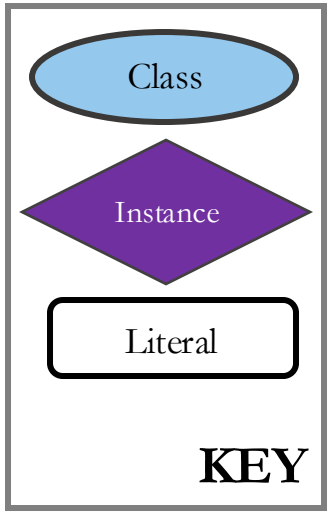
rdfs:Literal relates resources to strings, which can be "plain" or "typed"
Typed literals are strings with an associated datatype tag
Plain literals are strings without such datatype tags

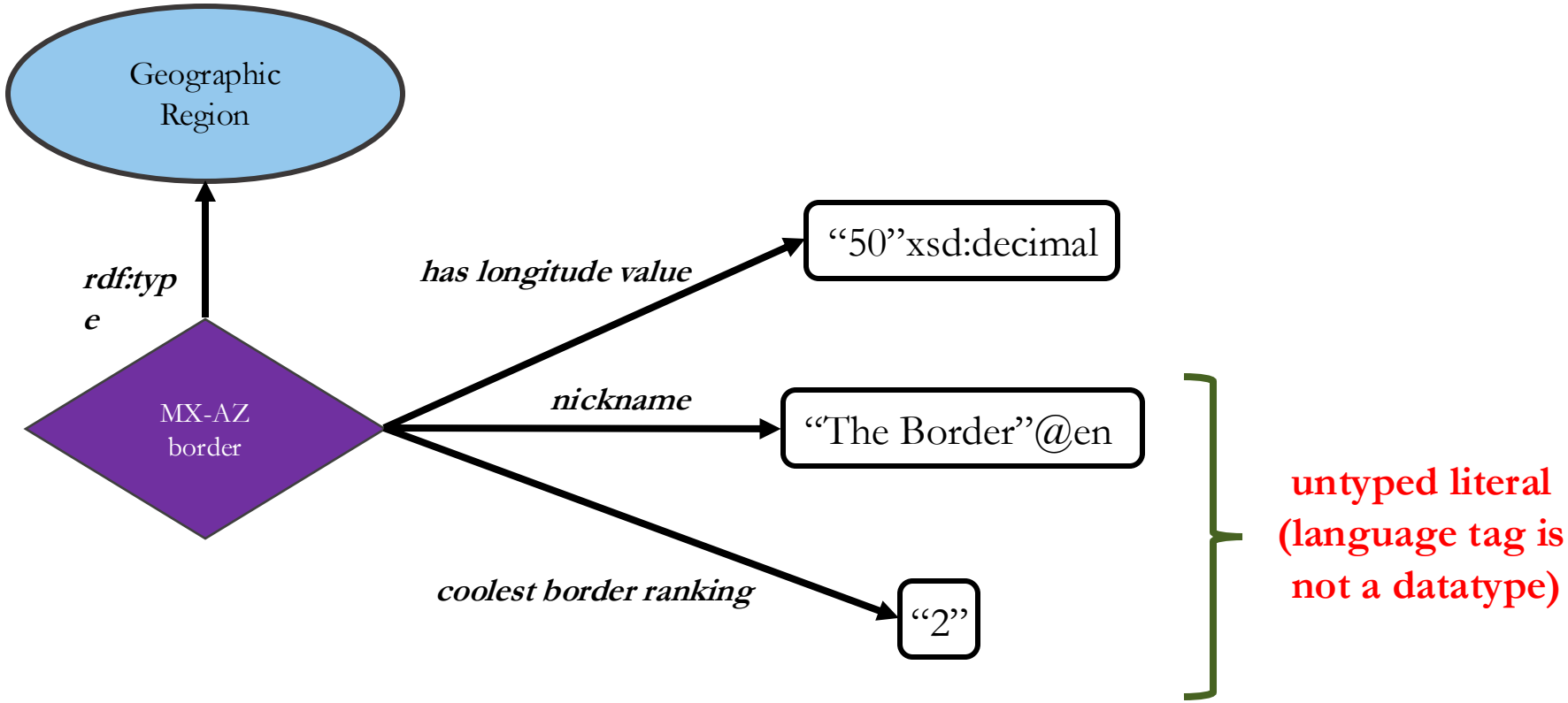
Literals


Geographic Region
rdf:type
MX-AZ border
has longitude value
"50"xsd:decimal
nickname
"The Border"@en
coolest border ranking
"2"

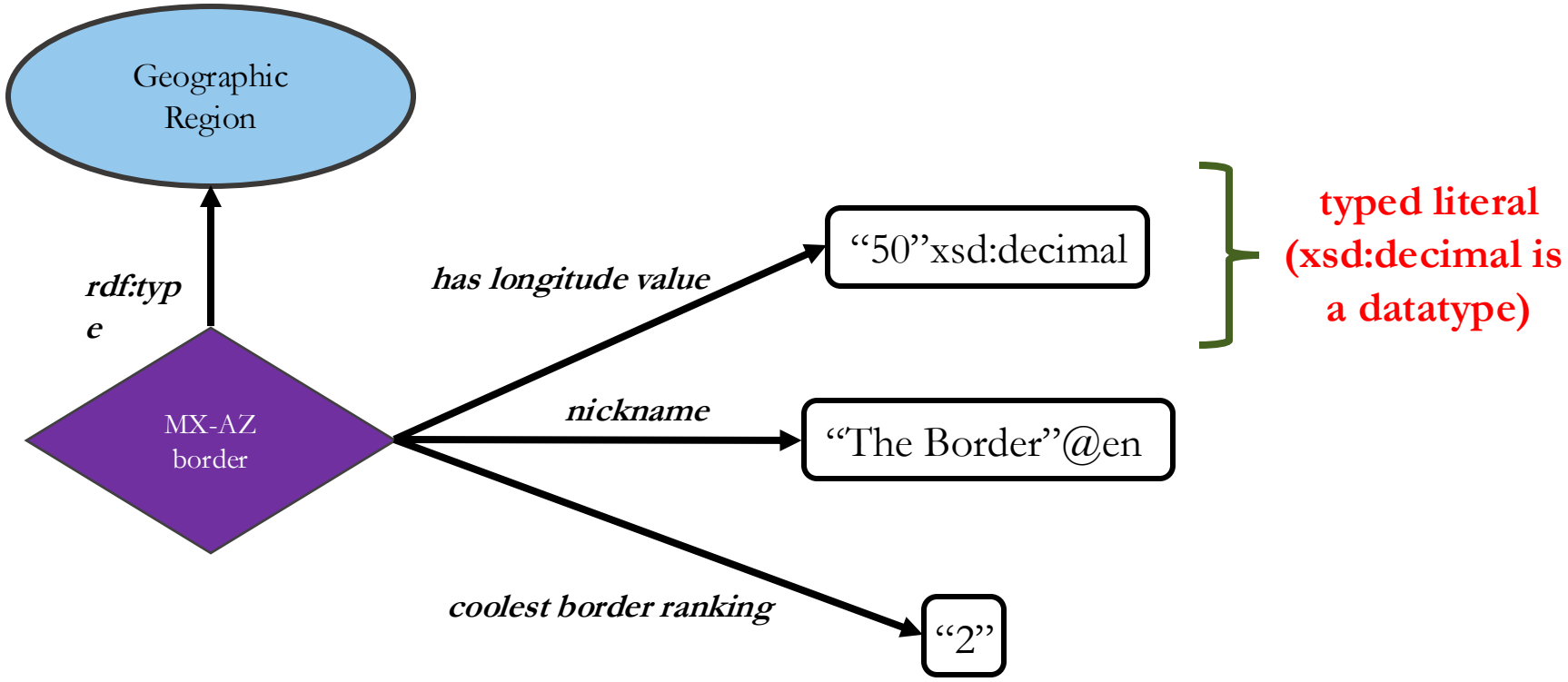

Class
Instance
Literal
KEY

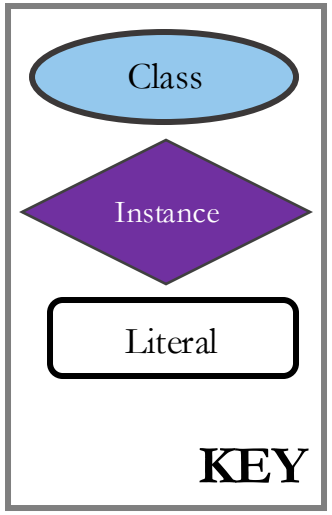
Literals


Class
Instance
Literal
KEY


Geographic Region
rdf:type
MX-AZ border
has longitude value
"50"xsd:decimal
nickname
"The Border"@en
coolest border ranking
"2"
untyped literal (language tag is not a datatype)

Literals


Geographic Region
rdf:type
MX-AZ border
has longitude value
"50"xsd:decimal
nickname
"The Border"@en
coolest border ranking
"2"
typed literal (xsd:decimal is a datatype)


Class
Instance
Literal
KEY

Logic of Datatypes

- xsd datatype semantics has been integrated in RDF, allowing for logical comparisons, equality, and calculations

Logic of Datatypes

- xsd datatype semantics has been integrated in RDF, allowing for logical comparisons, equality, and calculations

VALUE SPACE

Values datatype can represent.

E.g. xsd:integer has value space
all integers

Logic of Datatypes

- xsd datatype semantics has been integrated in RDF, allowing for logical comparisons, equality, and calculations

VALUE SPACE

Values datatype can represent.

E.g. xsd:integer has value space all integers

LEXICAL SPACE

Strings representing valid values for datatype

E.g. xsd:integer has a lexical space including strings such as “1” and “-4”

Logic of Datatypes

- xsd datatype semantics has been integrated in RDF, allowing for logical comparisons, equality, and calculations

VALUE SPACE

Values datatype can represent.

E.g. xsd:integer has value space all integers

LEXICAL SPACE

Strings representing valid values for datatype

E.g. xsd:integer has a lexical space including strings such as “1” and “-4”

Both value **and** lexical space are needed because **some values have multiple lexical representations**, e.g. “01” and “1”

Logic of Datatypes

- When comparisons are conducted, for example by using an OWL reasoner, xsd datatype definitions are used to ensure:

  - Integer comparions are conducted **numerically**, e.g. “3”^^xsd:integer is less than “30”^^xsd:integer
  - Strings are compared **lexically**, e.g. “friend”^^xsd:string is identical to “friend”^^xsd:string, but not “Friend”^^xsd:string
  - Dates are ordered **chronologically**, e.g. “12-2-2023”^^xsd:datatime is earlier than “12-2-2024”^^xsd:datatime

Resource Description Framework Schema

- RDFs is an extension of RDF facilitating representation of:

rdfs:Class	rdfs:Literal
rdfs:subClassOf	rdfs:Datatype
rdfs:domain	rdfs:label
rdfs:range	rdfs:comment
rdfs:subPropertyOf	rdfs:isDefinedBy
rdfs:Resource	

rdfs:label is an annotation property that allows you to provide a human-readable name for resources, so you don't need to look at the ugly URI all the time…

Classes | Object properties | Data properties | Annotation properties | Datatypes | Individuals

Class hierarchy: continuant

Asserted

- owl:Thing
  - entity
    - **continuant**
      - generically dependent continuant
      - independent continuant
        - immaterial entity
          - continuant fiat boundary
            - fiat line
            - fiat point
            - fiat surface
          - site
          - spatial region
            - one-dimensional spatial region
            - three-dimensional spatial region
            - two-dimensional spatial region
            - zero-dimensional spatial region
        - material entity
          - fiat object part
          - object
          - object aggregate
      - specifically dependent continuant
        - quality
          - relational quality
        - realizable entity
          - disposition
            - function
          - role
    - occurrent
      - process

continuant — BFO:0000002 — http://purl.obolibrary.org/obo/BFO_0000002

Annotations | Usage

Annotations: continuant

Annotations

rdfs:label [language: en]

continuant

skos:prefLabel [language: en]

continuant

skos:definition [language: en]

(Elucidation) A continuant is an entity that persists, endures, or continues to exist through time while maintaining its identity

dc:identifier

008-BFO

skos:example [language: en]

A human being, a tennis ball, a cave, a region of space, someone's temperature.

Description: continuant

Equivalent To

SubClass Of

- 'continuant part of at some time' only continuant
- entity

General class axioms

Classes | Object properties | Data properties | Annotation properties | Datatypes | Individuals

continuant — BFO:0000002 — http://purl.obolibrary.org/obo/BFO_0000002

Annotations | Usage

Class hierarchy: continuant

Asserted

- owl:Thing
  - entity
    - continuant
      - generically dependent continuant
      - independent continuant
        - immaterial entity
          - continuant fiat boundary
            - fiat line
            - fiat point
            - fiat surface
          - site
          - spatial region
            - one-dimensional spatial region
            - three-dimensional spatial region
            - two-dimensional spatial region
            - zero-dimensional spatial region
        - material entity
          - fiat object part
          - object
          - object aggregate
      - specifically dependent continuant
        - quality
          - relational quality
        - realizable entity
          - disposition
            - function
          - role
    - occurrent
      - process

Annotations: continuant

Annotations

rdfs:label [language: en]

continuant

skos:prefLabel [language: en]

continuant

skos:definition [language: en]

(Elucidation) A continuant is an entity that persists, endures, or continues to exist through time while maintaining its identity

dc:identifier

008-BFO

skos:example [language: en]

A human being, a tennis ball, a cave, a region of space, someone's temperature.

Description: continuant

Equivalent To

SubClass Of

- 'continuant part of at some time' only continuant
- entity

General class axioms

Resource Description Framework Schema

- RDFs is an extension of RDF facilitating representation of:

rdfs:Class
rdfs:subClassOf
rdfs:domain
rdfs:range
rdfs:subPropertyOf
rdfs:Resource
rdfs:Literal
rdfs:Datatype
rdfs:label
rdfs:comment
rdfs:isDefinedBy

rdfs:comment is an annotation property that allows to you create comments on resources for, say, suggested changes, updates, disagreements, etc.

Classes | Object properties | Data properties | Annotation properties | Datatypes | Individuals

Class hierarchy: continuant

Asserted

- owl:Thing
  - entity
    - continuant
      - generically dependent continuant
      - independent continuant
        - immaterial entity
          - continuant fiat boundary
            - fiat line
            - fiat point
            - fiat surface
          - site
          - spatial region
            - one-dimensional spatial region
            - three-dimensional spatial region
            - two-dimensional spatial region
            - zero-dimensional spatial region
        - material entity
      - specifically dependent continuant
        - quality
          - relational quality
        - realizable entity
          - disposition
            - function
          - role
    - occurrent
      - process

continuant — BFO:0000002 — http://purl.obolibrary.org/obo/BFO_0000002

Annotations | Usage

Annotations: continuant

Annotations

rdfs:label [language: en]

continuant

skos:prefLabel [language: en]

continuant

skos:definition [language: en]

(Elucidation) A continuant is an entity that persists, endures, or continues to exist through time while maintaining its identity

rdfs:comment

Hi there, I'm a comment! Here is my content:

Help me. It's so dark and I haven't eaten for - I don't know - maybe days; I've lost count. The wraiths will be back soon. I can feel them, watching, waiting for me to sleep...waiting for the world to fall away...like tears in rain.

dc:identifier

008-BFO

skos:example [language: en]

A human being, a tennis ball, a cave, a region of space, someone's temperature.

Classes | Object properties | Data properties | Annotation properties | Datatypes | Individuals

Class hierarchy: continuant

Asserted

- owl:Thing
  - entity
    - continuant
      - generically dependent continuant
      - independent continuant
        - immaterial entity
          - continuant fiat boundary
            - fiat line
            - fiat point
            - fiat surface
          - site
          - spatial region
            - one-dimensional spatial region
            - three-dimensional spatial region
            - two-dimensional spatial region
            - zero-dimensional spatial region
        - material entity
      - specifically dependent continuant
        - quality
          - relational quality
        - realizable entity
          - disposition
            - function
          - role
    - occurrent
      - process

continuant — BFO:0000002 — http://purl.obolibrary.org/obo/BFO_0000002

Annotations | Usage

Annotations: continuant

Annotations

rdfs:label [language: en]

continuant

skos:prefLabel [language: en]

continuant

skos:definition [language: en]

(Elucidation) A continuant is an entity that persists, endures, or continues to exist through time while maintaining its identity

rdfs:comment

Hi there, I'm a comment! Here is my content:

Help me. It's so dark and I haven't eaten for - I don't know - maybe days; I've lost count. The wraiths will be back soon. I can feel them, watching, waiting for me to sleep...waiting for the world to fall away...like tears in rain.

dc:identifier

008-BFO

skos:example [language: en]

A human being, a tennis ball, a cave, a region of space, someone's temperature.

Resource Description Framework Schema

- RDFs is an extension of RDF facilitating representation of:

rdfs:Class	rdfs:Literal
rdfs:subClassOf	rdfs:Datatype
rdfs:domain	rdfs:label
rdfs:range	rdfs:comment
rdfs:subPropertyOf	rdfs:isDefinedBy
rdfs:Resource	

rdfs:isDefinedBy is an annotation property indicating the primary location of a resource's definition

- **entity**
  - continuant
    - generically dependent continuant
    - independent continuant
    - specifically dependent continuant
  - occurrent
- obsolete class
- Version

From the Cell Line Ontology, which imports BFO as its top-level architecture.

BFO is the primary location of the definition for “entity”

Preferred Name	entity
Synonyms	
ID	http://purl.obolibrary.org/obo/BFO_0000001
BFO CLIF specification label	Entity
BFO OWL specification label	entity
editor note	BFO 2 Reference: In all areas of empirical inquiry we encounter general terms of two sorts. First are general terms which refer to types:animaltuberculosissurgical procedurediseaseSecond, are general terms used to refer to groups of entities which instantiate but do not correspond to the extension of any subuniversal of that universal because there is nothing intrinsic to the entities in which they – and only they – are counted as belonging to the given group. Examples are: animal purchased by the Emperortuberc a Wednesdaysurgical procedure performed on a patient from Stockholmperson identified as candidate for clinical trial #2056-55 signatory of Form 656-PPVpainting by Leonardo da VinciSuch terms, which represent what are called ‘specializations’ in [81 Entity doesn't have a closure axiom because the subclasses don't necessarily exhaust all possibilites. For example Werner Ceuste reality' include 4 sorts, entities (as BFO construes them), universals, configurations, and relations. It is an open question as to wh construed in BFO will at some point also include these other portions of reality. See, for example, 'How to track absolutely everyt http://www.referent-tracking.com/_RTU/papers/CeustersICbookRevised.pdf
editor preferred label	entity
elucidation	An entity is anything that exists or has existed or will exist. (axiom label in BFO2 Reference: [001-001])
example of usage	the Second World War Julius Caesar Verdi’s Requiem your body mass index
imported from	http://purl.obolibrary.org/obo/uberon.owl http://purl.obolibrary.org/obo/caro.owl http://purl.obolibrary.org/obo/obi.owl http://purl.obolibrary.org/obo/BFO
label	entity
prefixIRI	BFO:0000001
prefLabel	entity
rdfs:isDefinedBy	http://purl.obolibrary.org/obo/bfo.owl
subClassOf	Thing

- entity
  - continuant
    - generically dependent continuant
    - independent continuant
    - specifically dependent continuant
  - occurrent
- obsolete class
- Version

From the Cell Line Ontology, which imports BFO as its top-level architecture.

BFO is the primary location of the definition for “entity”

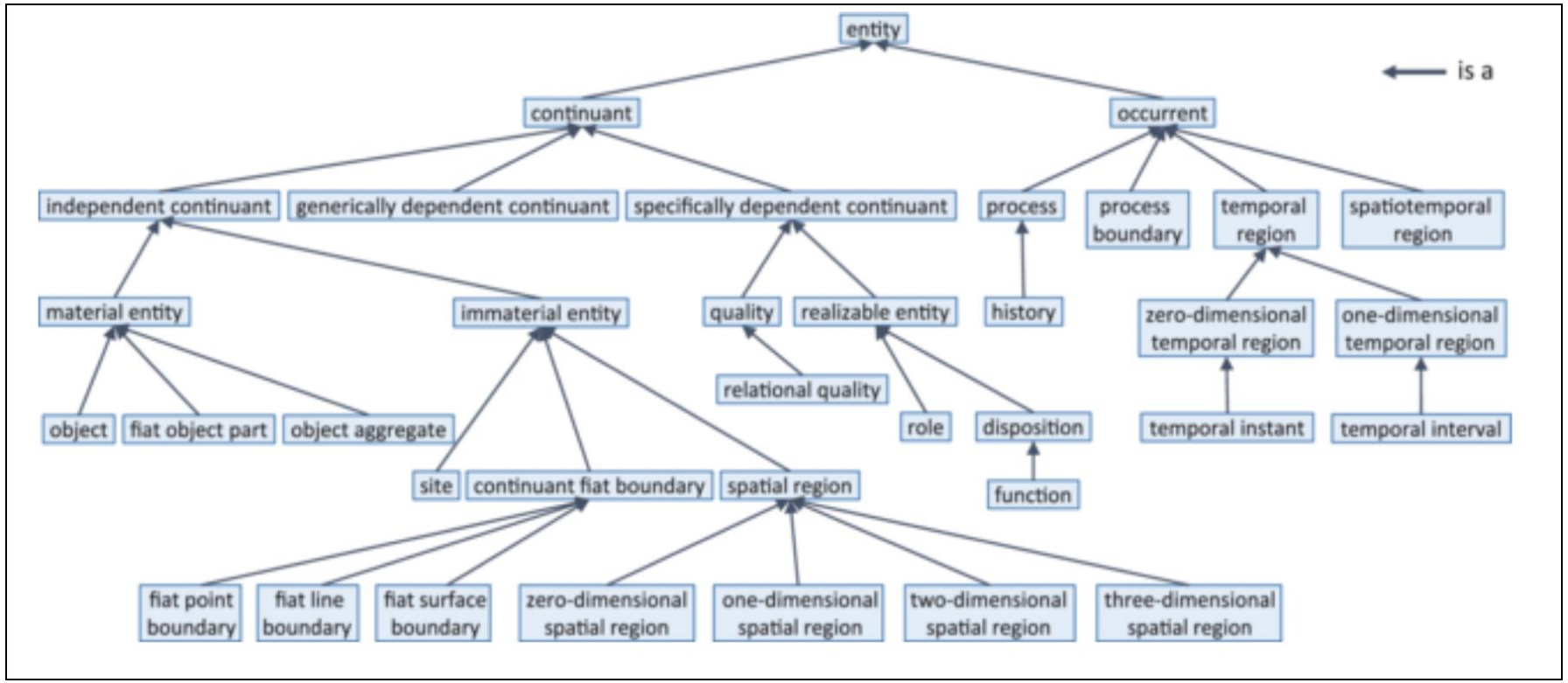
Preferred Name	entity
Synonyms	
ID	http://purl.obolibrary.org/obo/BFO_0000001
BFO CLIF specification label	Entity
BFO OWL specification label	entity
editor note	BFO 2 Reference: In all areas of empirical inquiry we encounter general terms of two sorts. First are general terms which refer to types:animaltuberculosissurgical procedurediseaseSecond, are general terms used to refer to groups of entities which instantiate but do not correspond to the extension of any subuniversal of that universal because there is nothing intrinsic to the entities in which they – and only they – are counted as belonging to the given group. Examples are: animal purchased by the Emperortuberc a Wednesdaysurgical procedure performed on a patient from Stockholmperson identified as candidate for clinical trial #2056-55 signatory of Form 656-PPVpainting by Leonardo da VinciSuch terms, which represent what are called ‘specializations’ in [81 Entity doesn't have a closure axiom because the subclasses don't necessarily exhaust all possibilites. For example Werner Ceuste reality' include 4 sorts, entities (as BFO construes them), universals, configurations, and relations. It is an open question as to wh construed in BFO will at some point also include these other portions of reality. See, for example, 'How to track absolutely everyt http://www.referent-tracking.com/_RTU/papers/CeustersICbookRevised.pdf
editor preferred label	entity
elucidation	An entity is anything that exists or has existed or will exist. (axiom label in BFO2 Reference: [001-001])
example of usage	the Second World War Julius Caesar Verdi’s Requiem your body mass index
imported from	http://purl.obolibrary.org/obo/uberon.owl http://purl.obolibrary.org/obo/caro.owl http://purl.obolibrary.org/obo/obi.owl http://purl.obolibrary.org/obo/BFO
label	entity
prefixIRI	BFO:0000001
prefLabel	entity
rdfs:isDefinedBy	http://purl.obolibrary.org/obo/bfo.owl
subClassOf	Thing

Outline

- Resource Description Framework (RDF)
- RDF Schema (RDFs)
- Modeling with Basic Formal Ontology
- Exercises

Basic Formal Ontology

BFO is such a standard, used by over 700 open-source groups, the first ISO/IEC top-level ontology standard, and a "baseline standard" for DOD-IC ontology development



Disambiguation

- **Information** vs what that information **is about**, e.g. occupation code vs a holder of an occupation
- **Material** vs **immaterial** things, e.g. a given river vs the site where the river used to flow
- **Bearing properties** vs **bearers of properties**, e.g. apple's redness vs the apple
- **Processes** vs **product**, e.g. ontology engineering vs ontology produced

Analogy

- BFO is analogous to the **Python programming language**; extensions of BFO are analogous to **Python libraries**

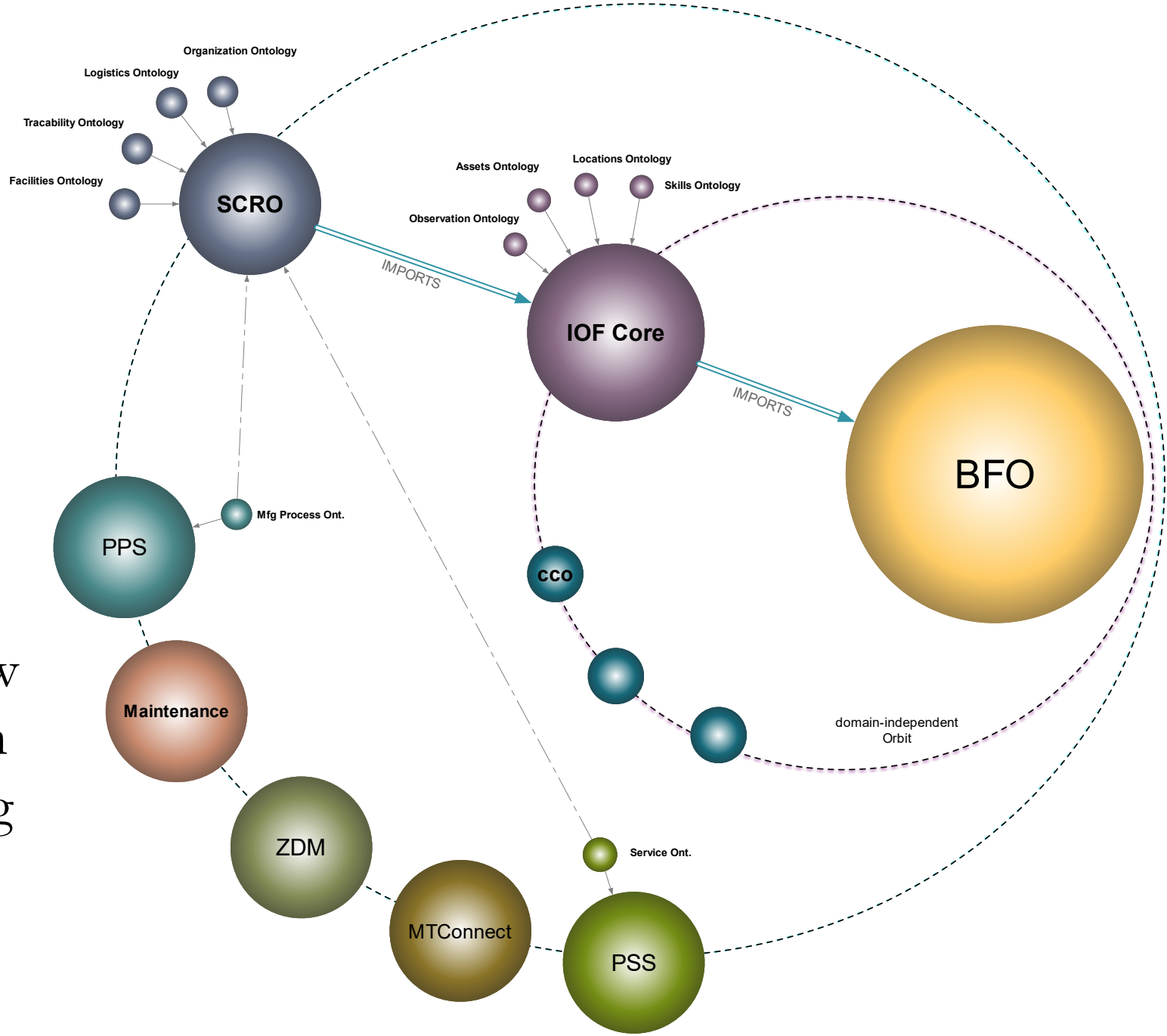
- You **could** create code that allows you to interact with, say, dataframes or you could **instead** start with Python and import a library like Pandas

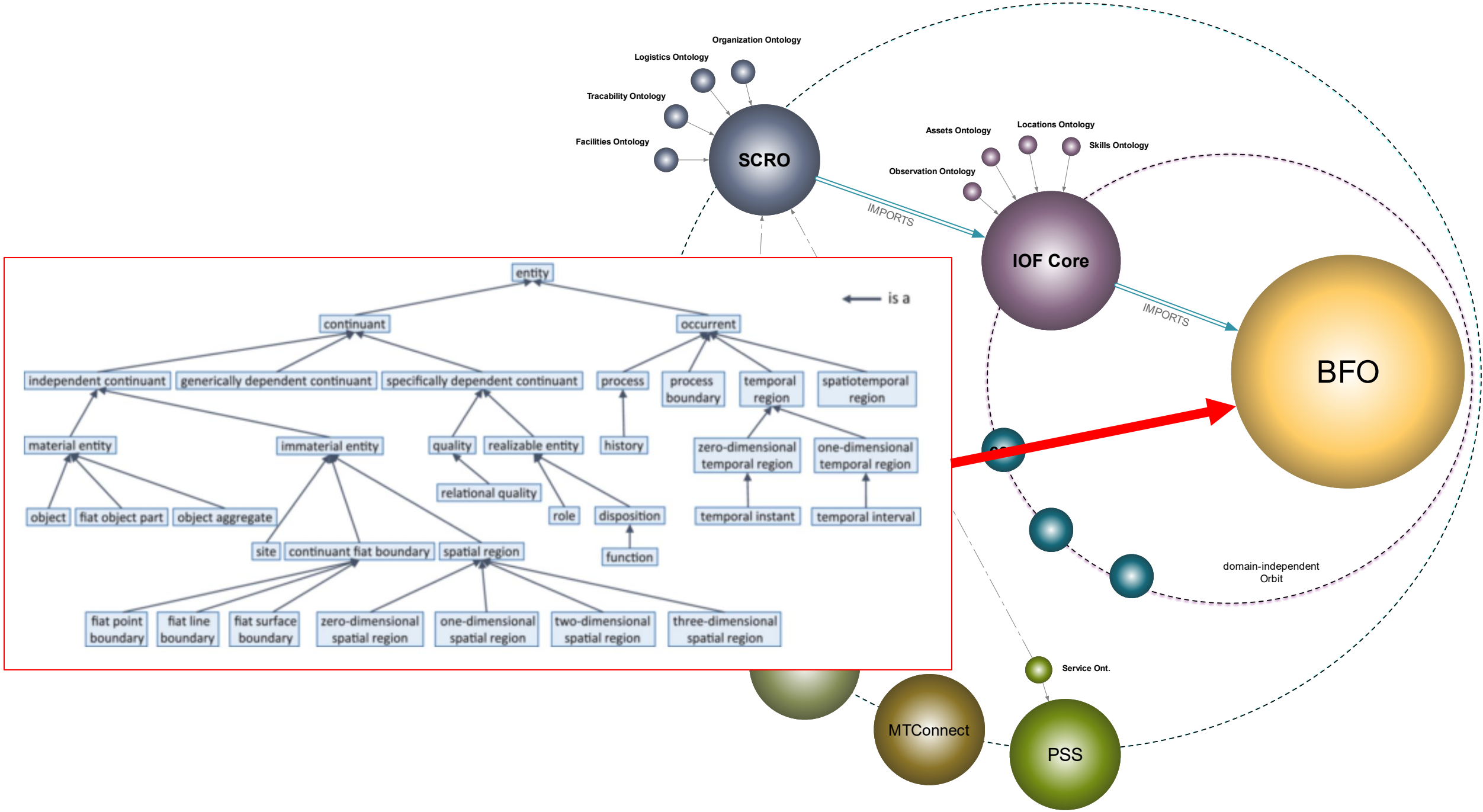
- You **could** create ontology elements that allow you to model artifacts and processes or you **could** instead start with BFO and import an extension

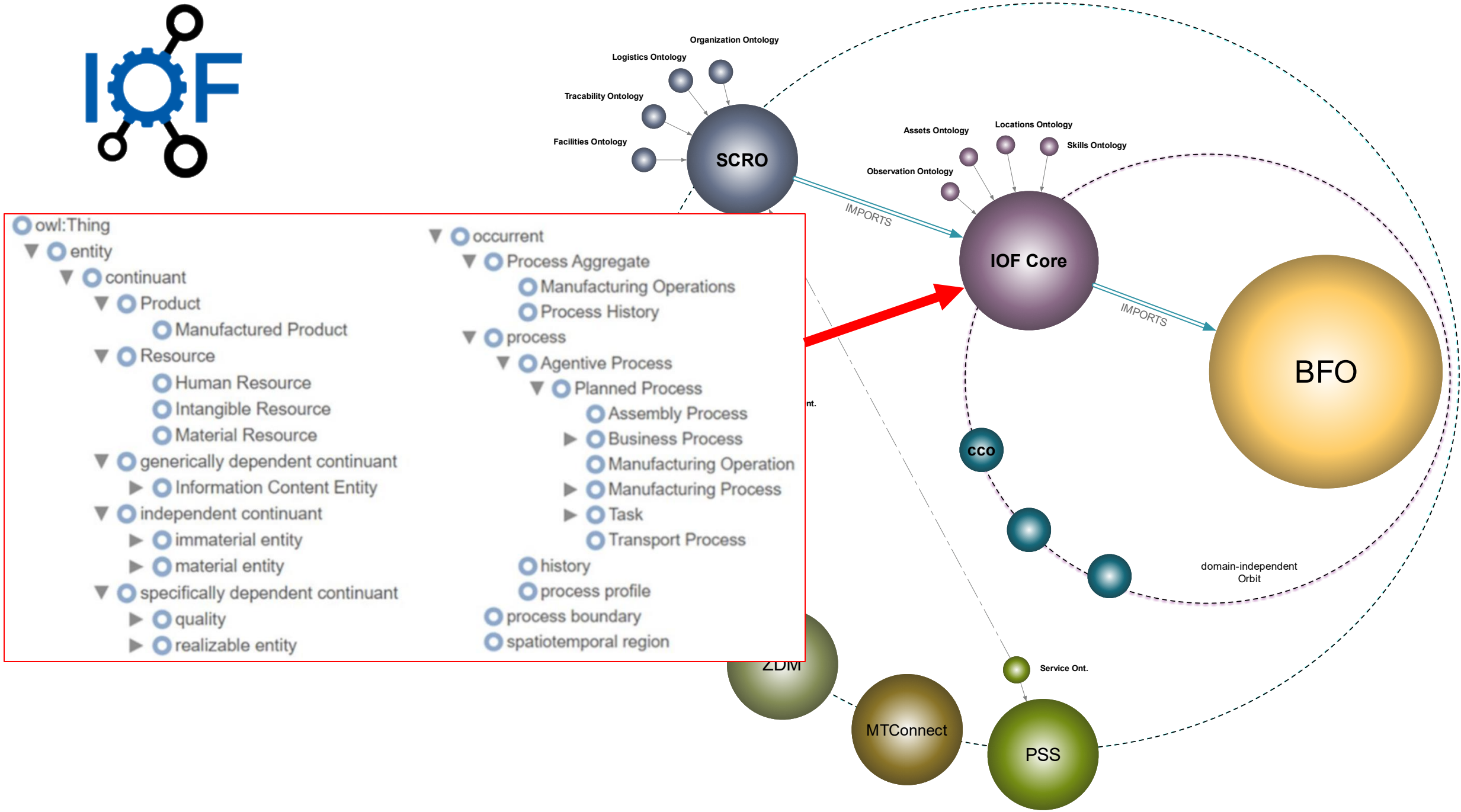
Hub & Spoke

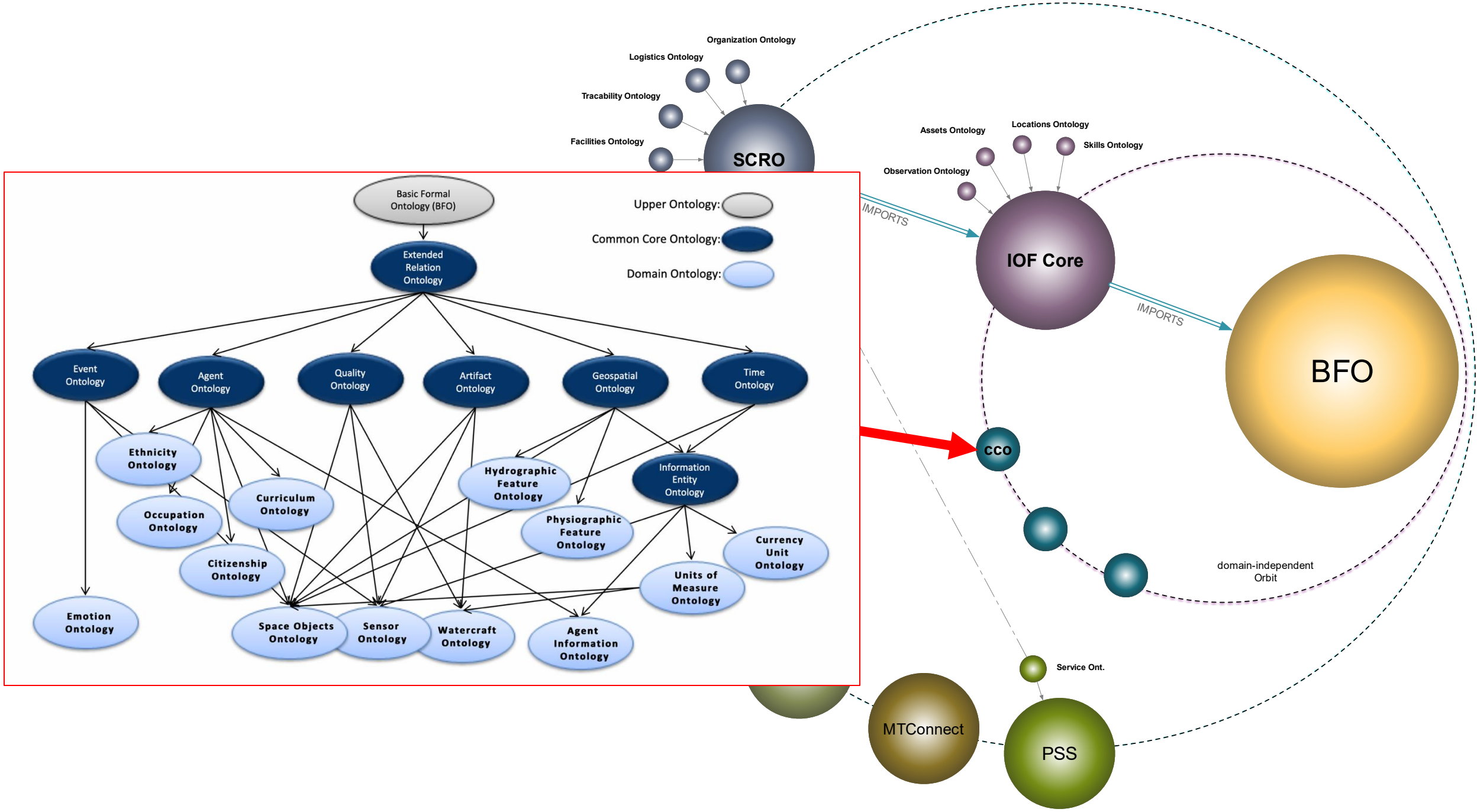
Ontologies extending from BFO are modules in a larger hub & spoke structure

Ontologies are extended by **downward population**, new classes have parent classes in a hierarchy ultimately leading to a BFO class

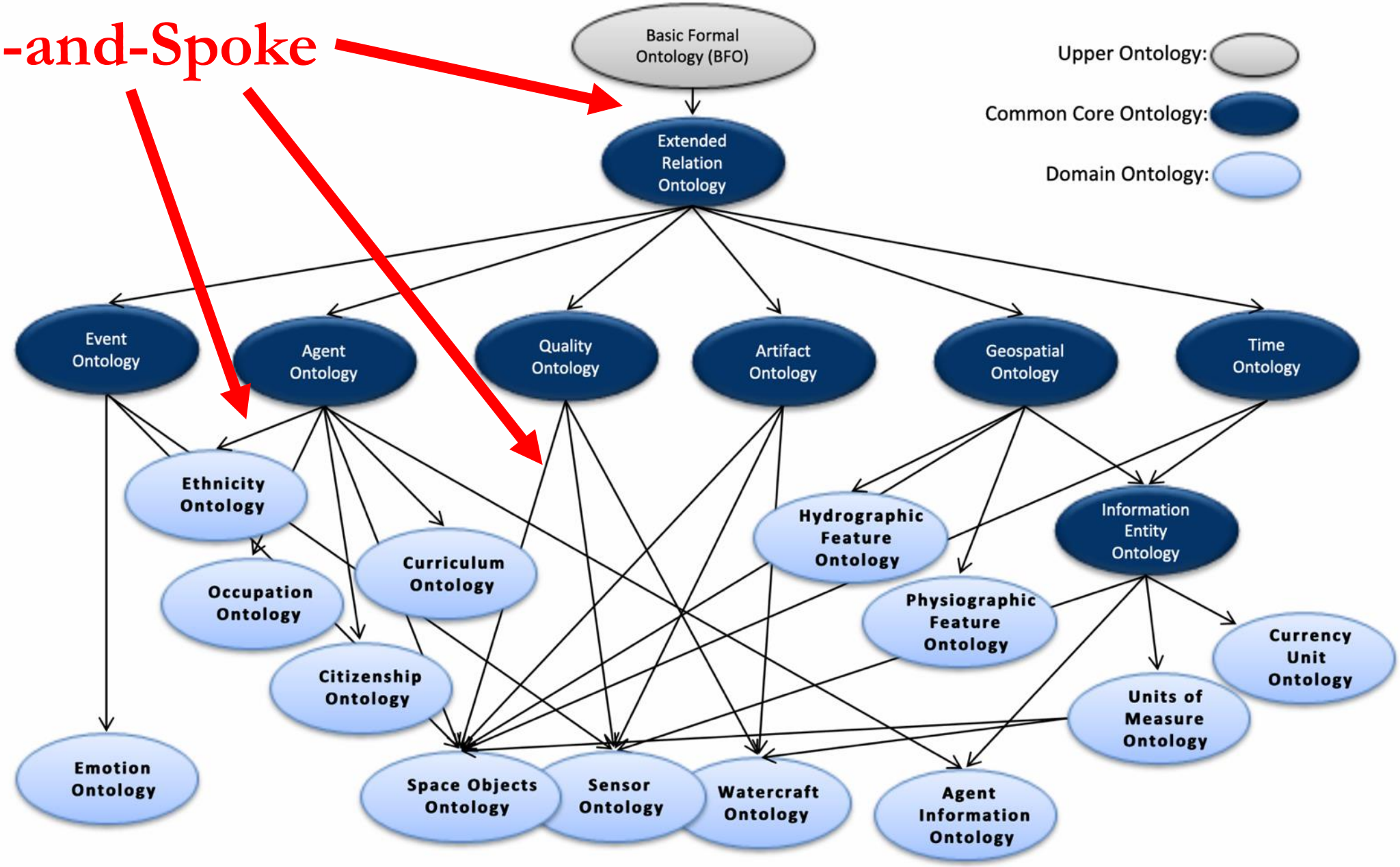



Organization Ontology
Logistics Ontology
Tracability Ontology
Facilities Ontology
SCRO
IMPORTS
Assets Ontology
Locations Ontology
Skills Ontology
Observation Ontology
IOF Core
IMPORTS
BFO
domain-independent Orbit
Service Ont.
MTConnect
PSS
entity
is a
continuant
occurrent
independent continuant
generically dependent continuant
specifically dependent continuant
process
process boundary
temporal region
spatiotemporal region
material entity
immaterial entity
quality
realizable entity
history
zero-dimensional temporal region
one-dimensional temporal region
relational quality
object
fiat object part
object aggregate
role
disposition
temporal instant
temporal interval
site
continuant fiat boundary
spatial region
function
fiat point boundary
fiat line boundary
fiat surface boundary
zero-dimensional spatial region
one-dimensional spatial region
two-dimensional spatial region
three-dimensional spatial region

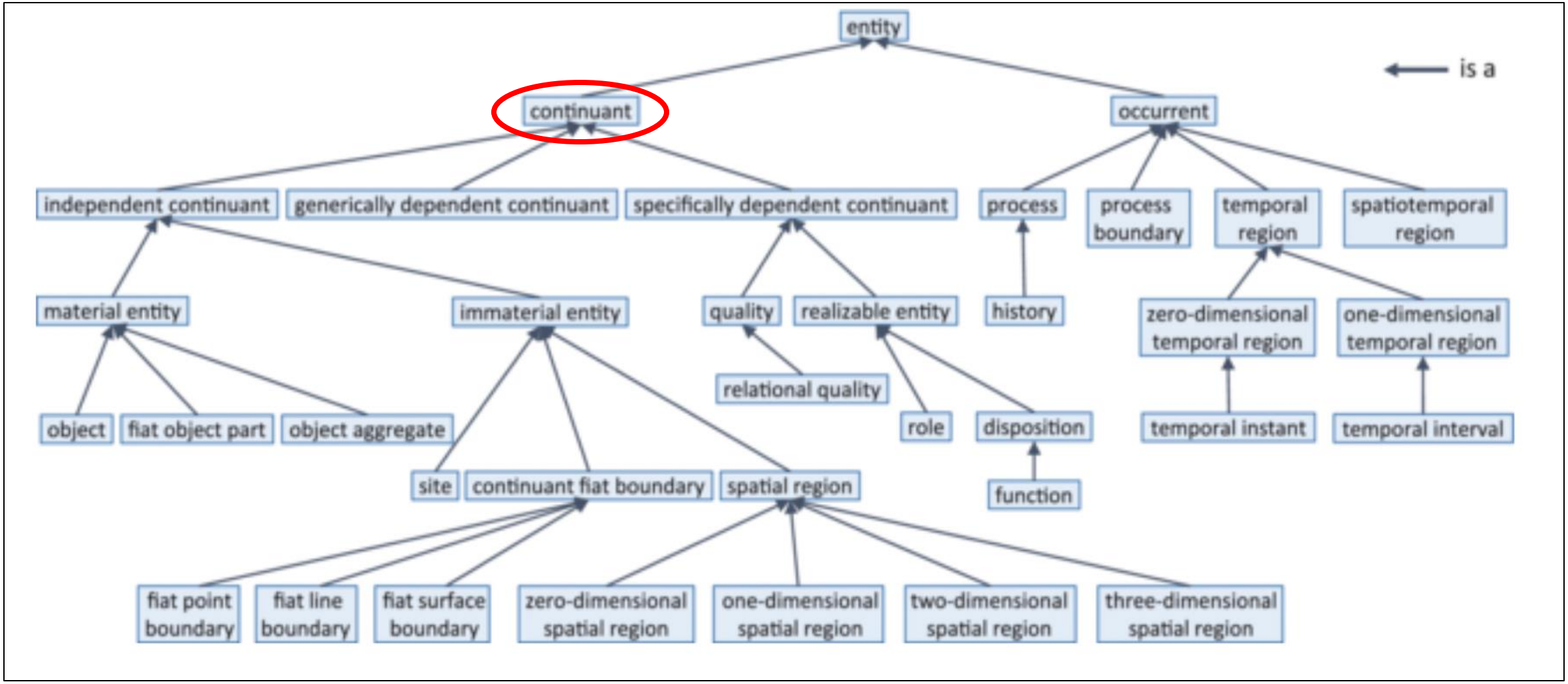

Organization Ontology
Logistics Ontology
Tracability Ontology
Facilities Ontology
SCRO
Assets Ontology
Locations Ontology
Skills Ontology
Observation Ontology
IMPORTS
IOF Core
IMPORTS
BFO
cco
domain-independent
Orbit
Service Ont.
MTConnect
PSS
owl:Thing
entity
continuant
Product
Manufactured Product
Resource
Human Resource
Intangible Resource
Material Resource
generically dependent continuant
Information Content Entity
independent continuant
immaterial entity
material entity
specifically dependent continuant
quality
realizable entity
occurrent
Process Aggregate
Manufacturing Operations
Process History
process
Agentive Process
Planned Process
Assembly Process
Business Process
Manufacturing Operation
Manufacturing Process
Task
Transport Process
history
process profile
process boundary
spatiotemporal region


Organization Ontology
Logistics Ontology
Tracability Ontology
Facilities Ontology
SCRO
Assets Ontology
Locations Ontology
Skills Ontology
Observation Ontology
IMPORTS
IOF Core
IMPORTS
BFO
Basic Formal Ontology (BFO)
Upper Ontology:
Common Core Ontology:
Domain Ontology:
Extended Relation Ontology
Event Ontology
Agent Ontology
Quality Ontology
Artifact Ontology
Geospatial Ontology
Time Ontology
Ethnicity Ontology
Occupation Ontology
Curriculum Ontology
Citizenship Ontology
Hydrographic Feature Ontology
Information Entity Ontology
Physiographic Feature Ontology
Currency Unit Ontology
Units of Measure Ontology
Emotion Ontology
Space Objects Ontology
Sensor Ontology
Watercraft Ontology
Agent Information Ontology
CCO
domain-independent Orbit
Service Ont.
MTConnect
PSS

Hub-and-Spoke

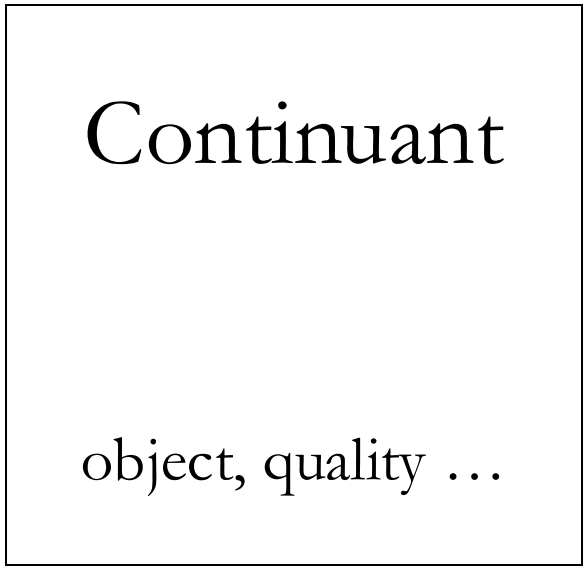


Continuant

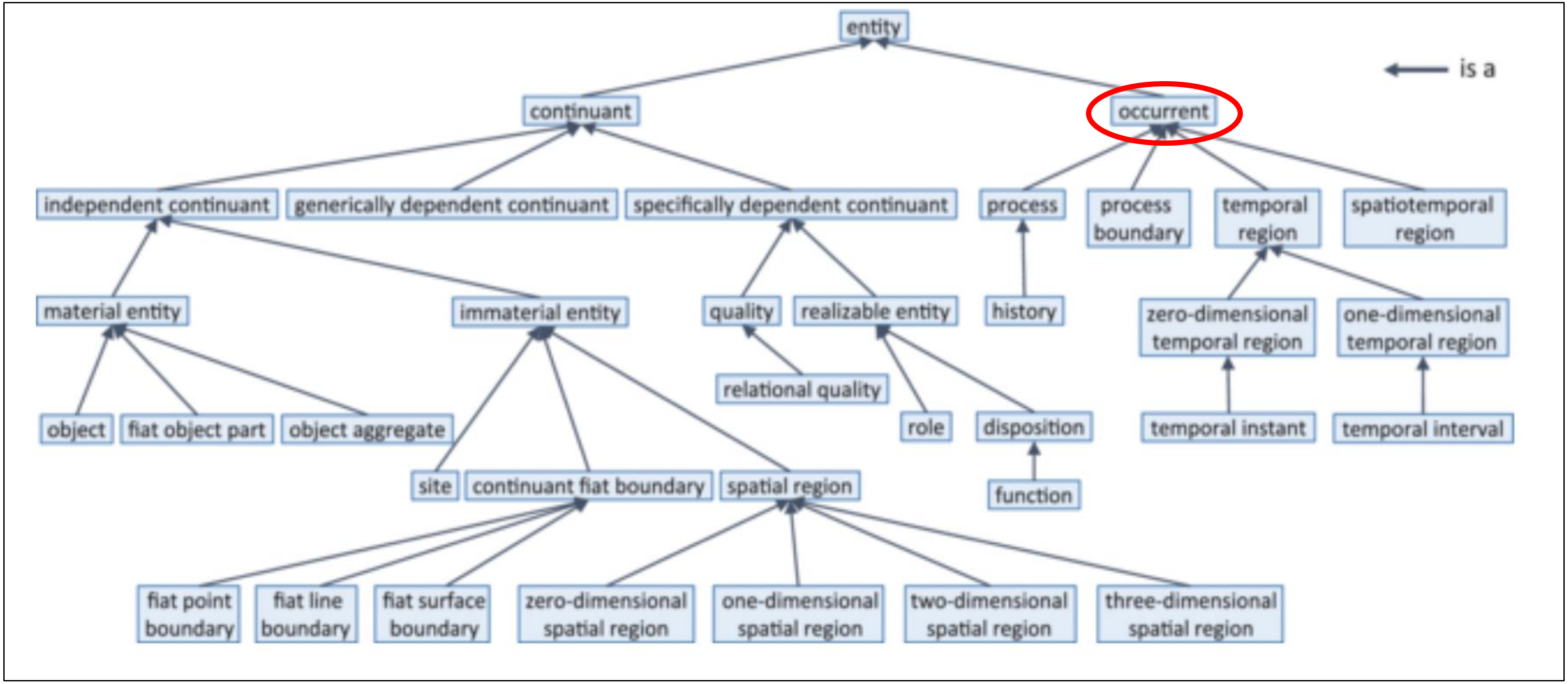

entity
is a
continuant
occurrent
independent continuant
generically dependent continuant
specifically dependent continuant
process
process boundary
temporal region
spatiotemporal region
material entity
immaterial entity
quality
realizable entity
history
zero-dimensional temporal region
one-dimensional temporal region
relational quality
object
fiat object part
object aggregate
role
disposition
temporal instant
temporal interval
site
continuant fiat boundary
spatial region
function
fiat point boundary
fiat line boundary
fiat surface boundary
zero-dimensional spatial region
one-dimensional spatial region
two-dimensional spatial region
three-dimensional spatial region

Continuant

- Continuants exist in time, wholly present whenever they exist at all; they are entities that lack temporal parts



Occurrent


entity
is a
continuant
occurrent
independent continuant
generically dependent continuant
specifically dependent continuant
process
process boundary
temporal region
spatiotemporal region
material entity
immaterial entity
quality
realizable entity
history
zero-dimensional temporal region
one-dimensional temporal region
relational quality
object
fiat object part
object aggregate
role
disposition
temporal instant
temporal interval
site
continuant fiat boundary
spatial region
function
fiat point boundary
fiat line boundary
fiat surface boundary
zero-dimensional spatial region
one-dimensional spatial region
two-dimensional spatial region
three-dimensional spatial region

Occurrent

- Occurrents exist over time, in that they have temporal parts

Continuant	Occurrent
object, quality …	process, event

Parthood

- Among the most important logical relationships is parthood

- Which in BFO comes in two flavors:
  - continuant parthood
  - occurrent parthood

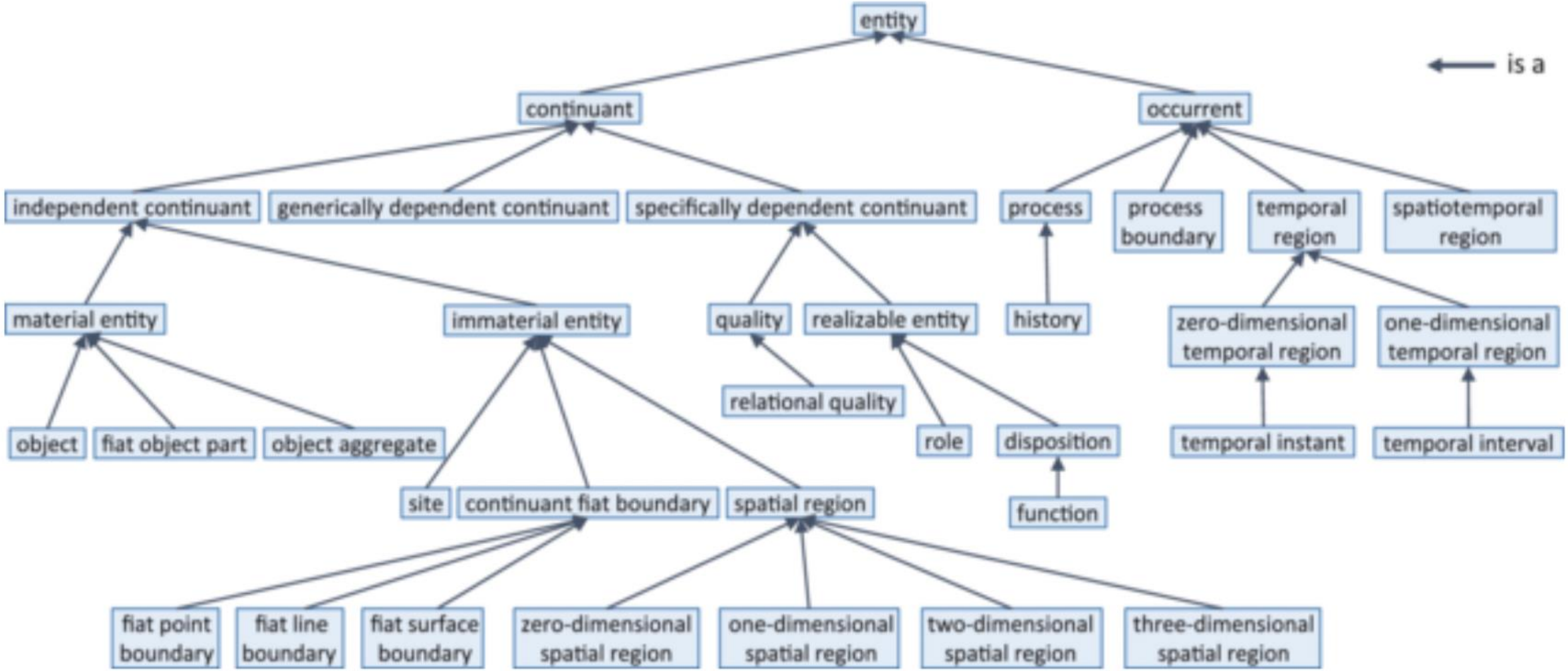
- Reflecting that the class Continuant is closed under parthood, and Occurrent is as well

No continuant may have or be part of any occurrent

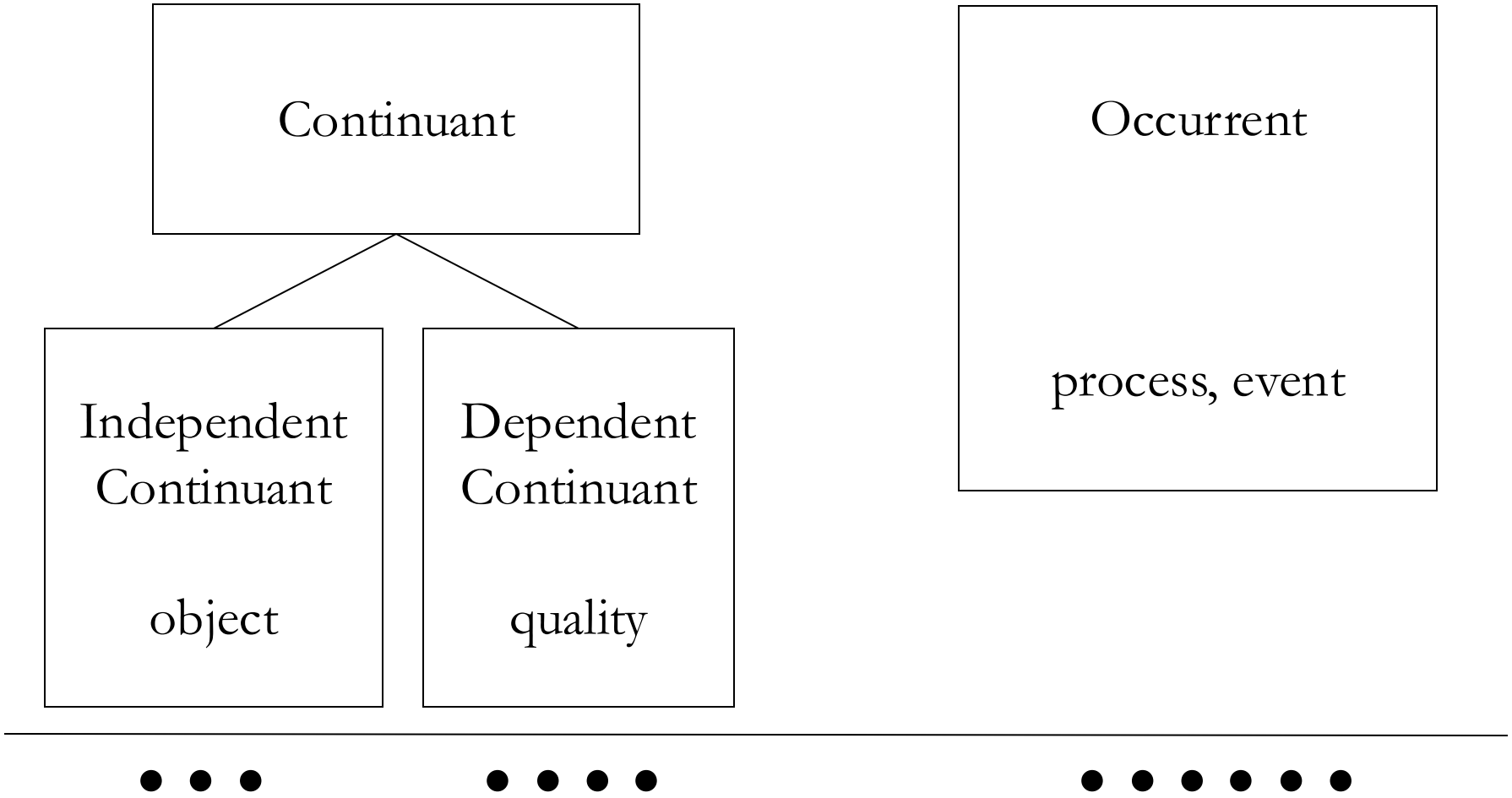
No occurrent may have or be part of any continuant

Classes represent collections of instances
For example: the class of *tables* falls under the class of *objects* and your dinner table would be an instance of the former

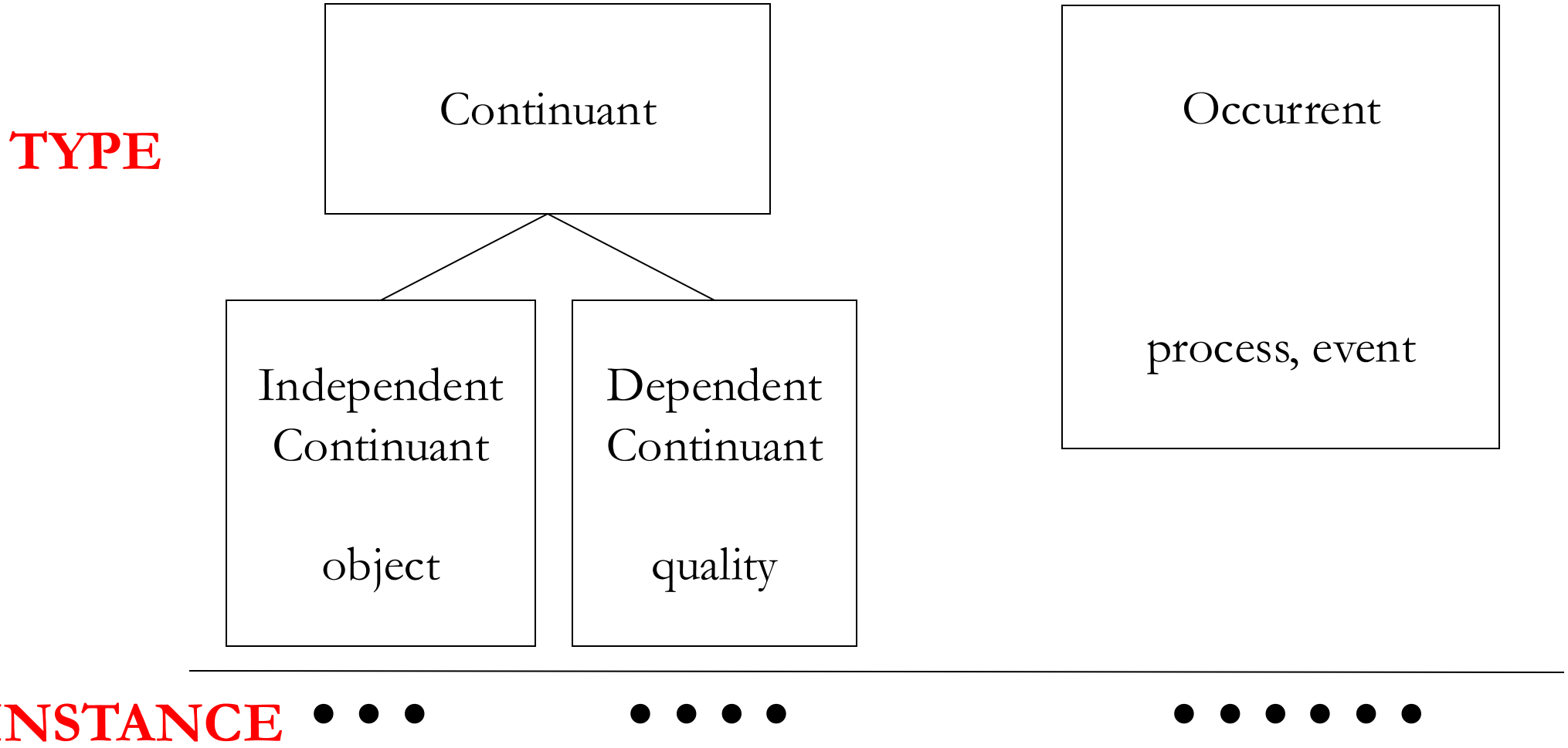
Class A *is_a* Class B means any instance of Class A is an instance of Class B



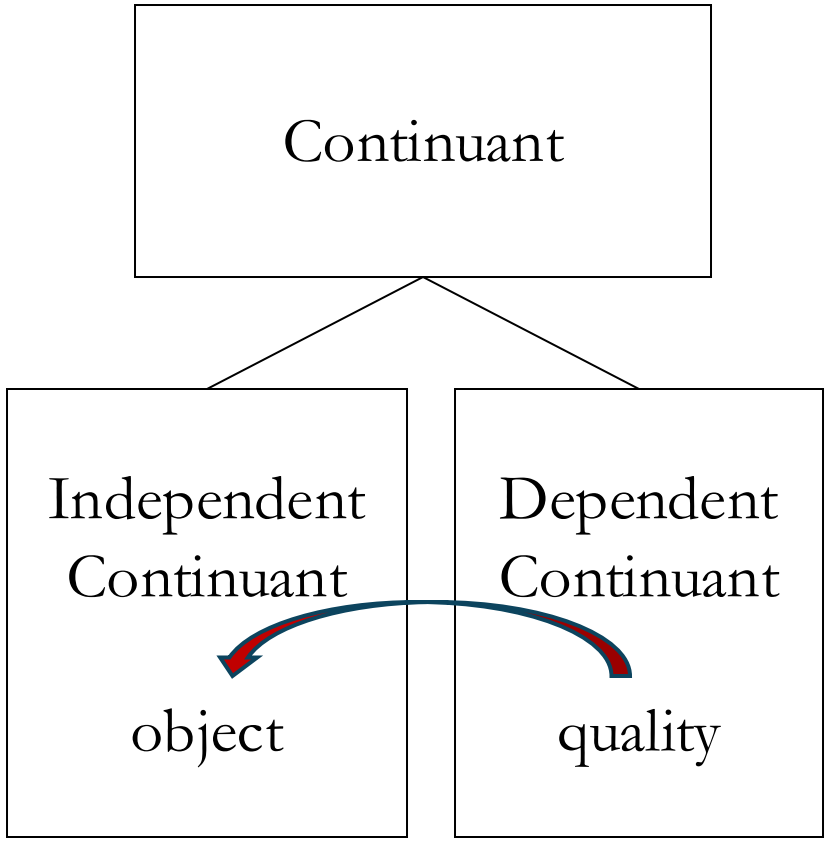
Types and Tokens

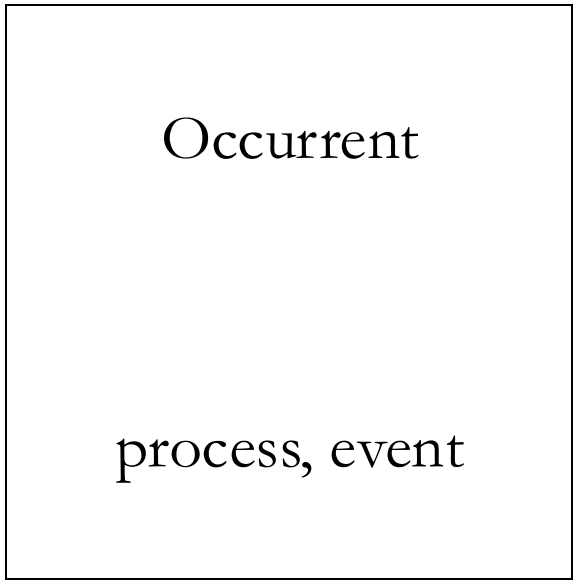


Types and Instances

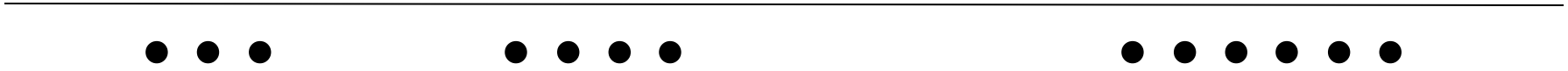


(In)dependence





Some continuants depend for their existence on others



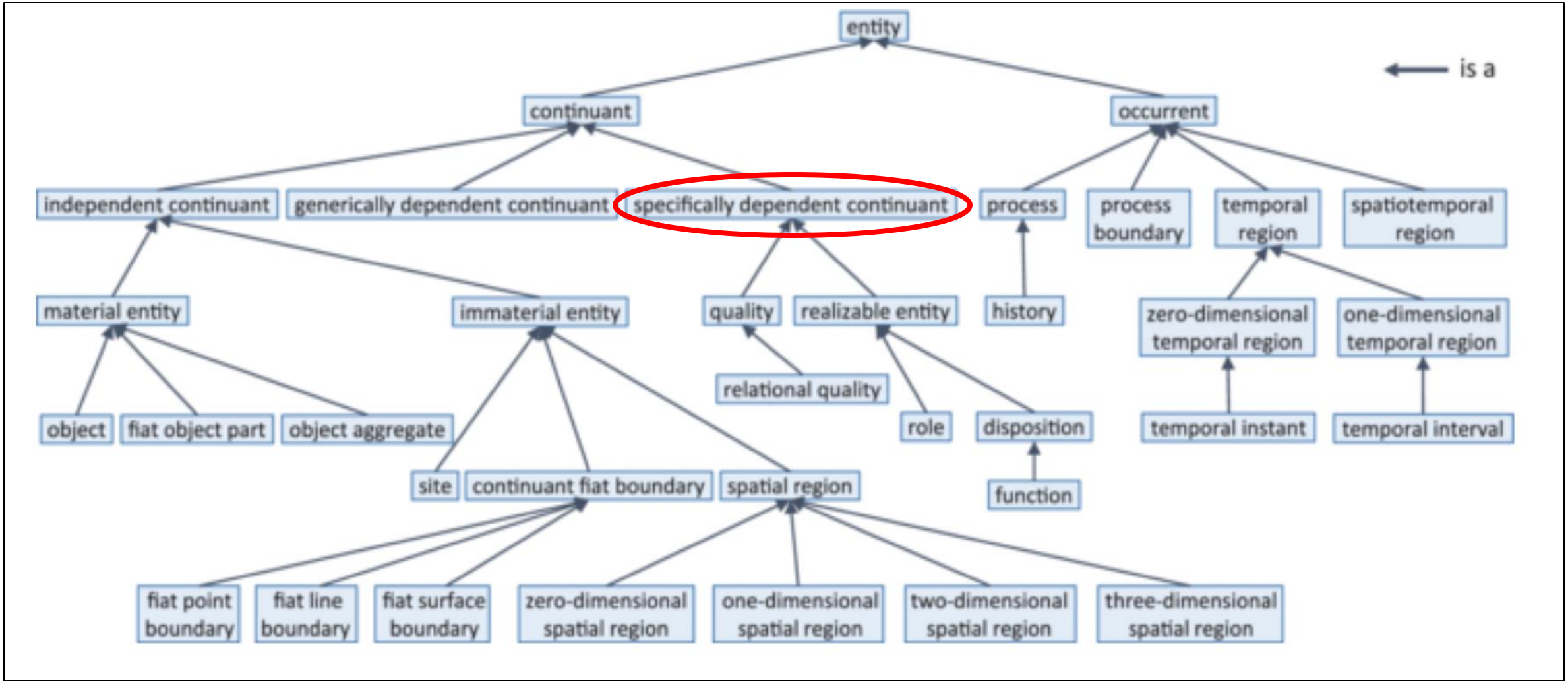
Dependence

- For certain entities, their existence depends on the existence of something else
- Other entities do not depend on any other entities for their existence
- The latter are categorized in BFO as **independent continuants**
- The former include **specifically dependent** and **generically dependent entities**, as well as **processes**

Dependence

- For certain entities, their existence depends on the existence of something else
- Other entities do not depend on any other entities for their existence
- The latter are categorized in BFO as **independent continuants**
- The former include **specifically dependent** and **generically dependent entities**, as well as **processes**

Specifically Dependent Continuant



Specifically Dependent Continuant

- These are continuants that in every case **specifically depend on** some independent contiuant for their existence

- For example, the mass of a tomato specifically depends on a given tomato, the shape of your smile depends on your face, and so on

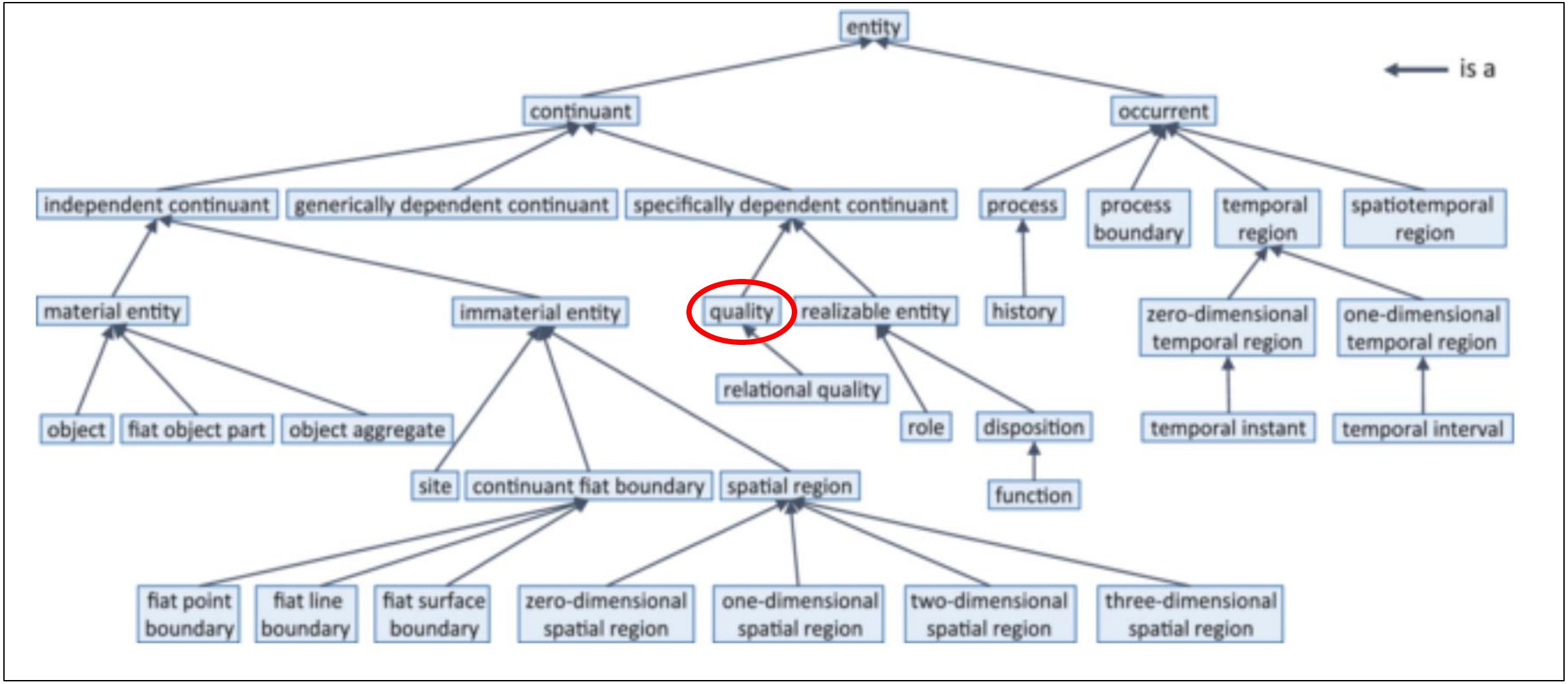
Specifically Dependent Continuant

- These are continuants that in every case **specifically depend on** some independent contiuant for their existence

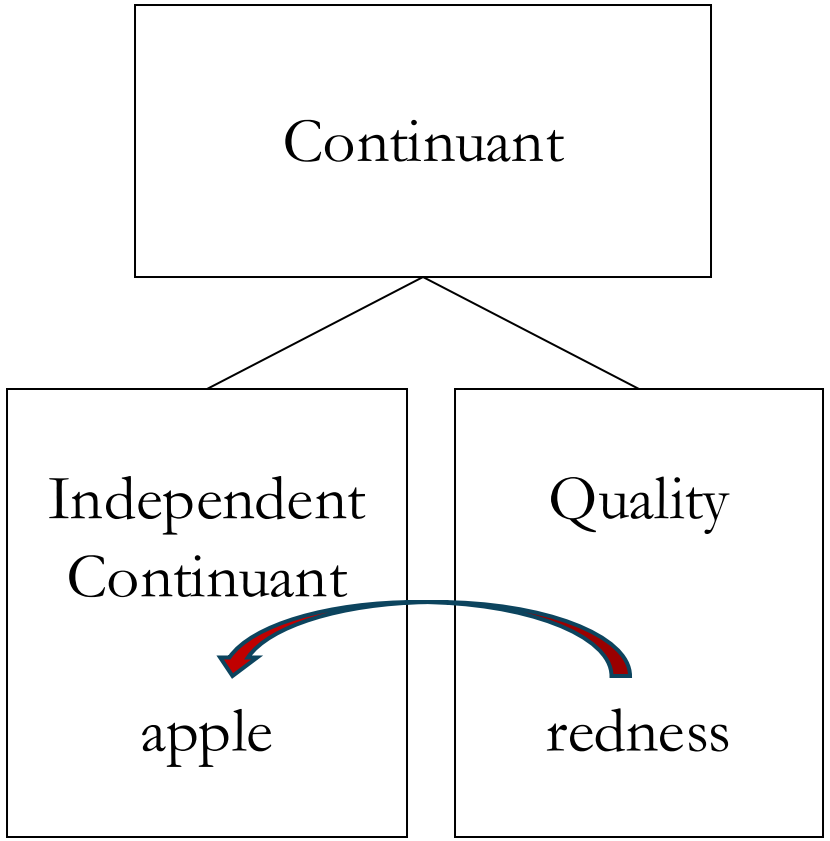
- For example, the mass of a tomato specifically depends on a given tomato, the shape of your smile depends on your face, and so on

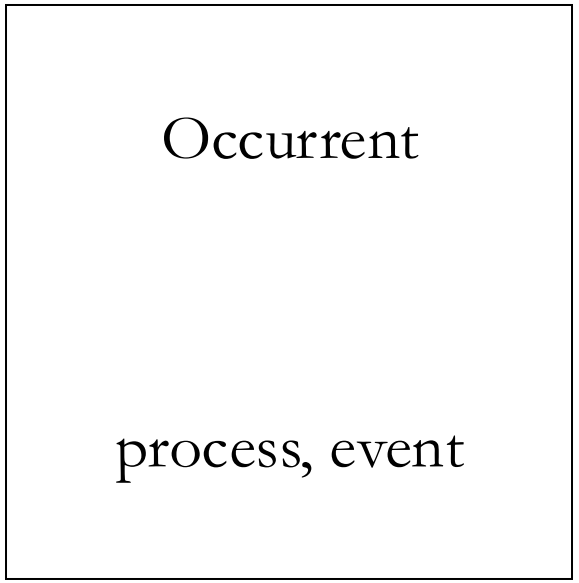
- Importantly, SDCs cannot migrate across bearers, i.e. the specific shape of your smile **depends on you** and so cannot **specifically depend on** me

Quality


entity
is a
continuant
occurrent
independent continuant
generically dependent continuant
specifically dependent continuant
process
process boundary
temporal region
spatiotemporal region
material entity
immaterial entity
quality
realizable entity
history
zero-dimensional temporal region
one-dimensional temporal region
relational quality
object
fiat object part
object aggregate
role
disposition
temporal instant
temporal interval
site
continuant fiat boundary
spatial region
function
fiat point boundary
fiat line boundary
fiat surface boundary
zero-dimensional spatial region
one-dimensional spatial region
two-dimensional spatial region
three-dimensional spatial region

Qualities





Qualities inhere in independent continuants

• • • • • •

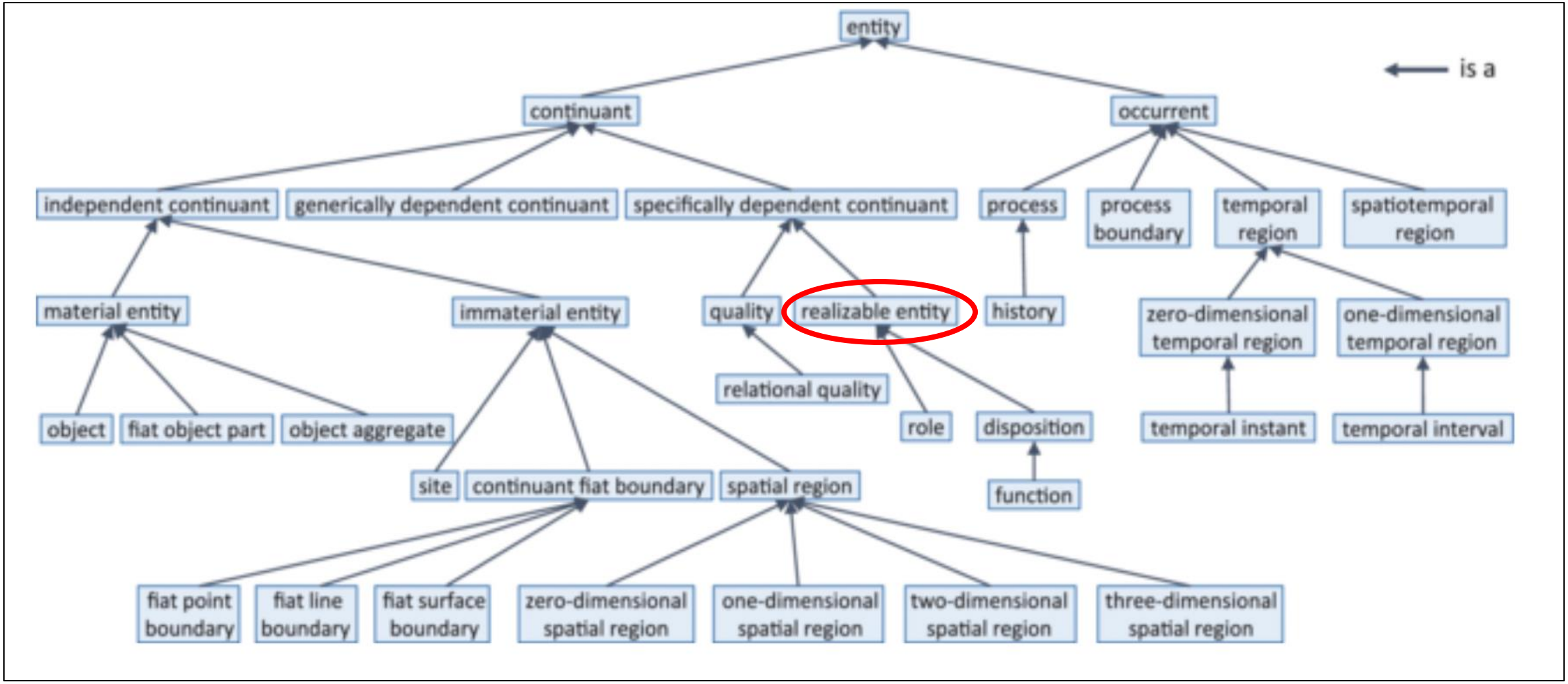
Quality

- In BFO, instances of Quality are said to manifest in full whenever they manifest at all

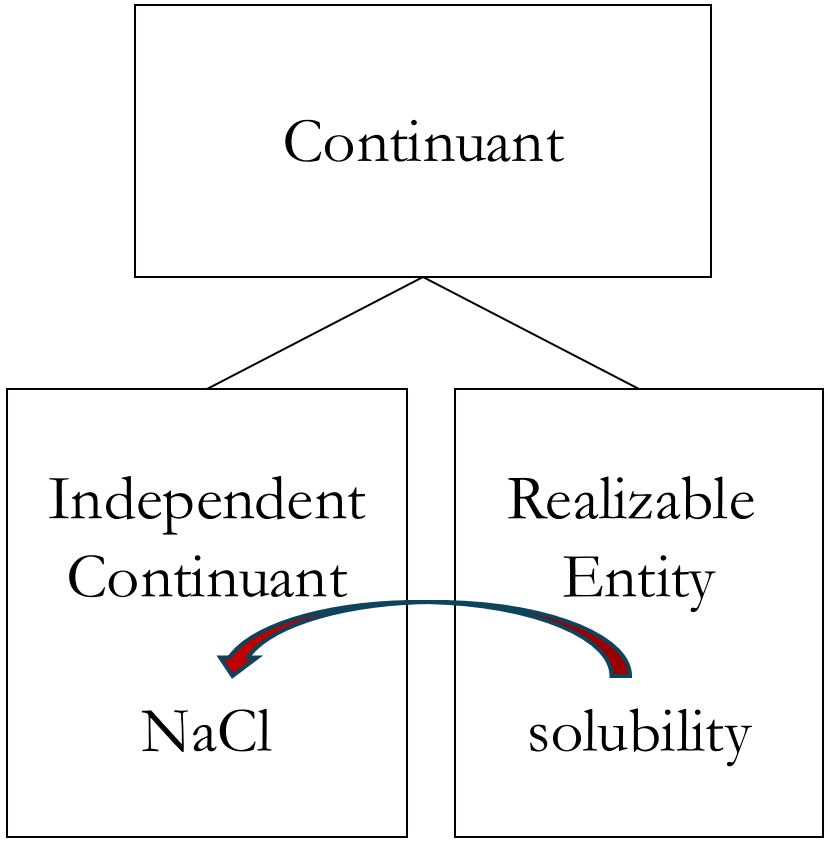
- For example, when an apple bears a redness quality, there is nothing more to that quality than the redness

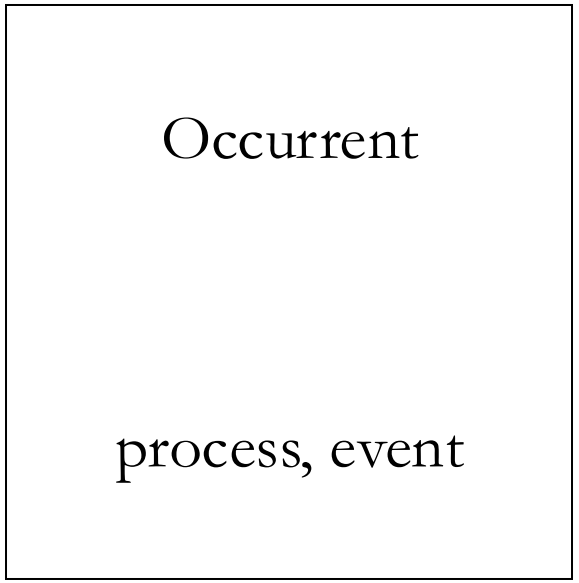
- Similarly, the shape of the smile on your face is there for the world to see whenever it is there at all; there is nothing more to the shape than what is presented on your face

Realizable Entity

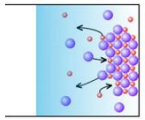


Realizable Entity





Not all dependent entities fully manifest when they exist



• • • • • •

Realizable Entity

- Attributes of some material bearer that only become manifest under certain conditions

- Put another way, realizable entities underwrite what bearers can do

Rule of Thumb

There is a portion of sodium chloride before you

- Suppose I ask "Does this portion exhibit a lattice structure?"
- You need only look at the salt to find an answer
- Suppose I ask "Is this portion soluble in unsaturated H_2O?"
- You cannot simply look at the salt to find an answer

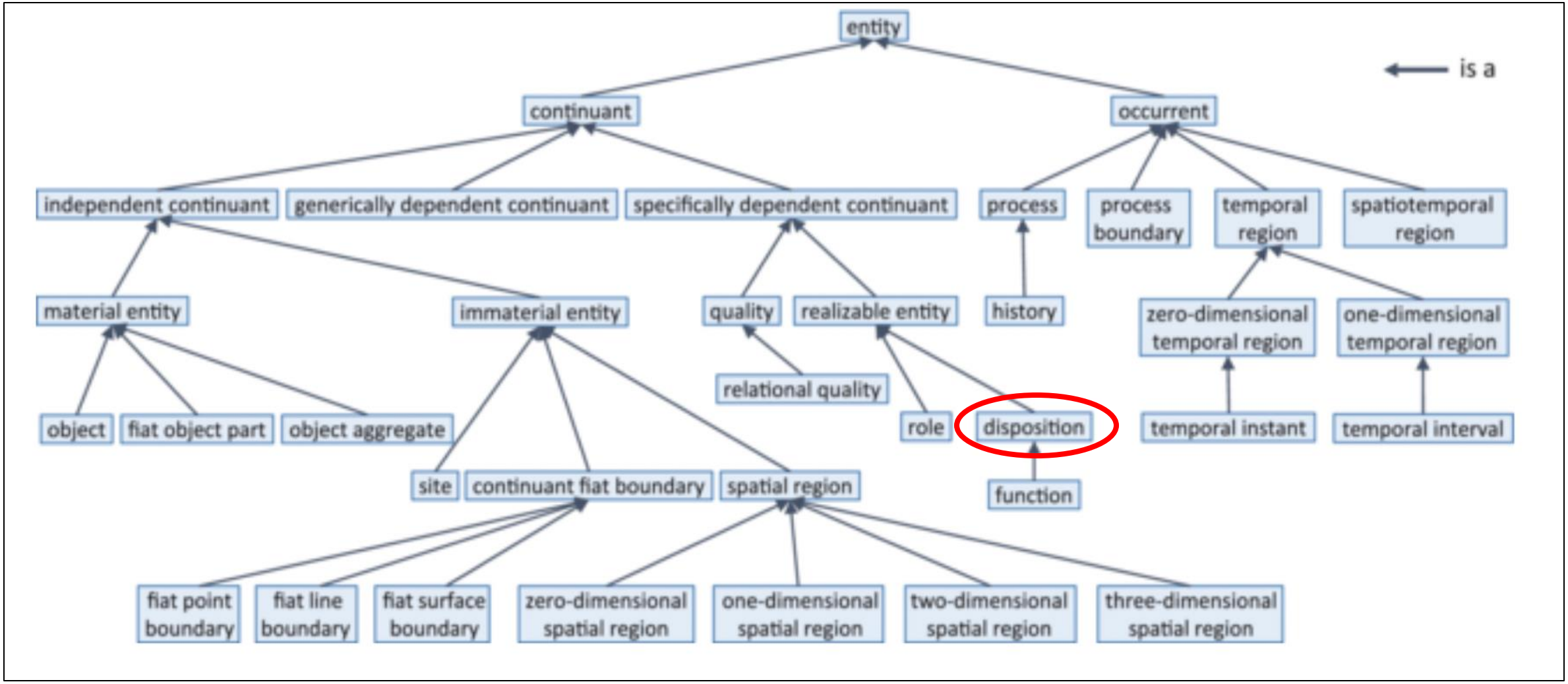
Modality

- Realizable Entities are how BFO represents **modality**

- The way the world is often differs from the way the world could be, could have been, and will be

- A yoga mat is black and you can see this; it could support a sleeping cat without you seeing it

Disposition

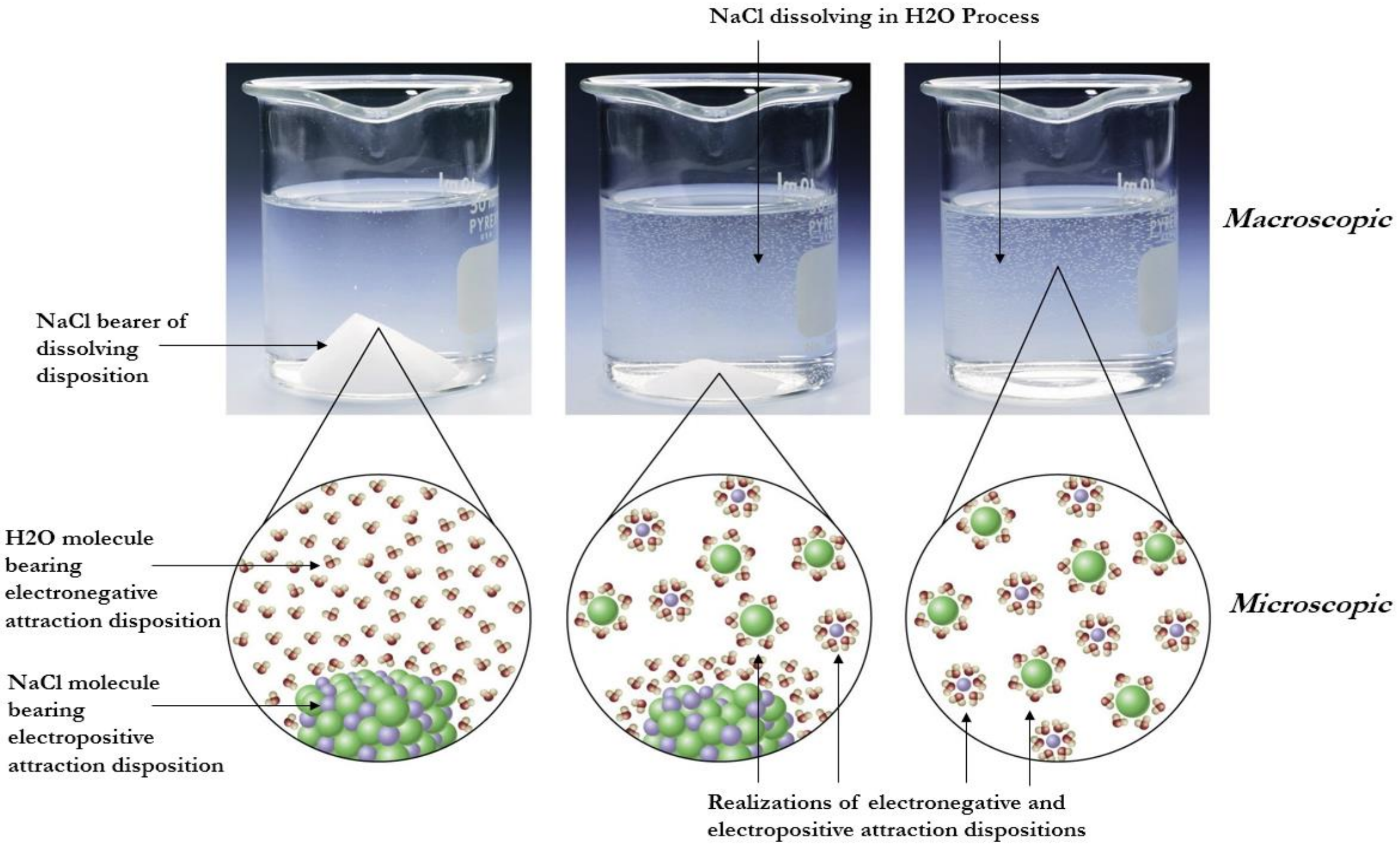


Disposition

- Attributes of some material bearer that only become manifest under certain conditions

Disposition

A realizable entity such that if it ceases to exist, then its bearer is physically changed, and its realization occurs when and because this bearer is in some special physical circumstances, and this realization occurs in virtue of the bearer's physical make-up


NaCl dissolving in H2O Process
Macroscopic
NaCl bearer of dissolving disposition
H2O molecule bearing electronegative attraction disposition
NaCl molecule bearing electropositive attraction disposition
Microscopic
Realizations of electronegative and electropositive attraction dispositions

Disposition

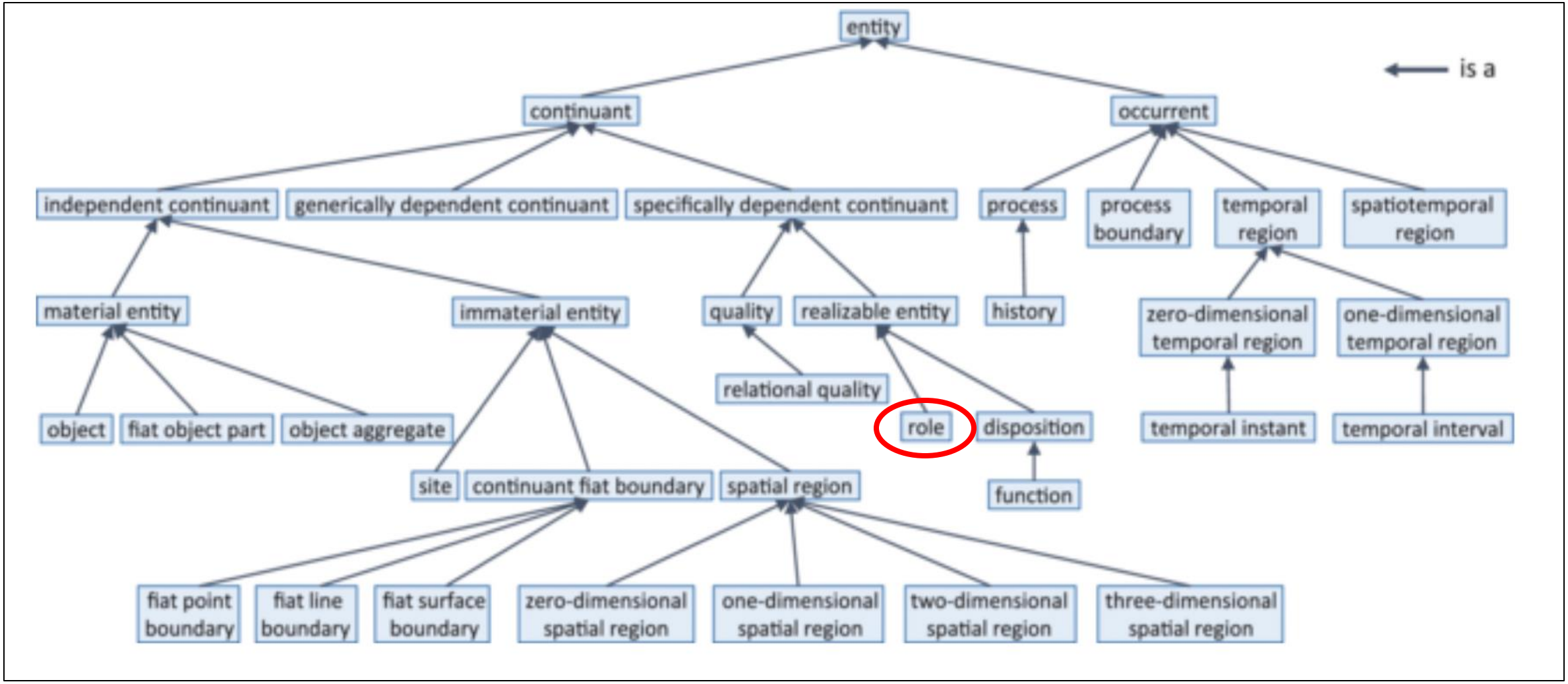
- Attributes of some material bearer that only become manifest under certain conditions

Disposition

A realizable entity such that if it ceases to exist, then its bearer is physically changed, and its realization occurs when and because this bearer is in some special physical circumstances, and this realization occurs in virtue of the bearer's physical make-up

INTERNALLY GROUNDED

Role


entity
is a
continuant
occurrent
independent continuant
generically dependent continuant
specifically dependent continuant
process
process boundary
temporal region
spatiotemporal region
material entity
immaterial entity
quality
realizable entity
history
zero-dimensional temporal region
one-dimensional temporal region
relational quality
object
fiat object part
object aggregate
role
disposition
temporal instant
temporal interval
site
continuant fiat boundary
spatial region
function
fiat point boundary
fiat line boundary
fiat surface boundary
zero-dimensional spatial region
one-dimensional spatial region
two-dimensional spatial region
three-dimensional spatial region

Role

- Attributes of some material bearer that only become manifest under certain conditions

Role

A realizable entity that exists because there is some single bearer that is in some special physical, social, or institutional set of circumstances in which this bearer does not have to be, and is not such that, if it ceases to exist, then the physical make-up of the bearer is thereby changed.

Role

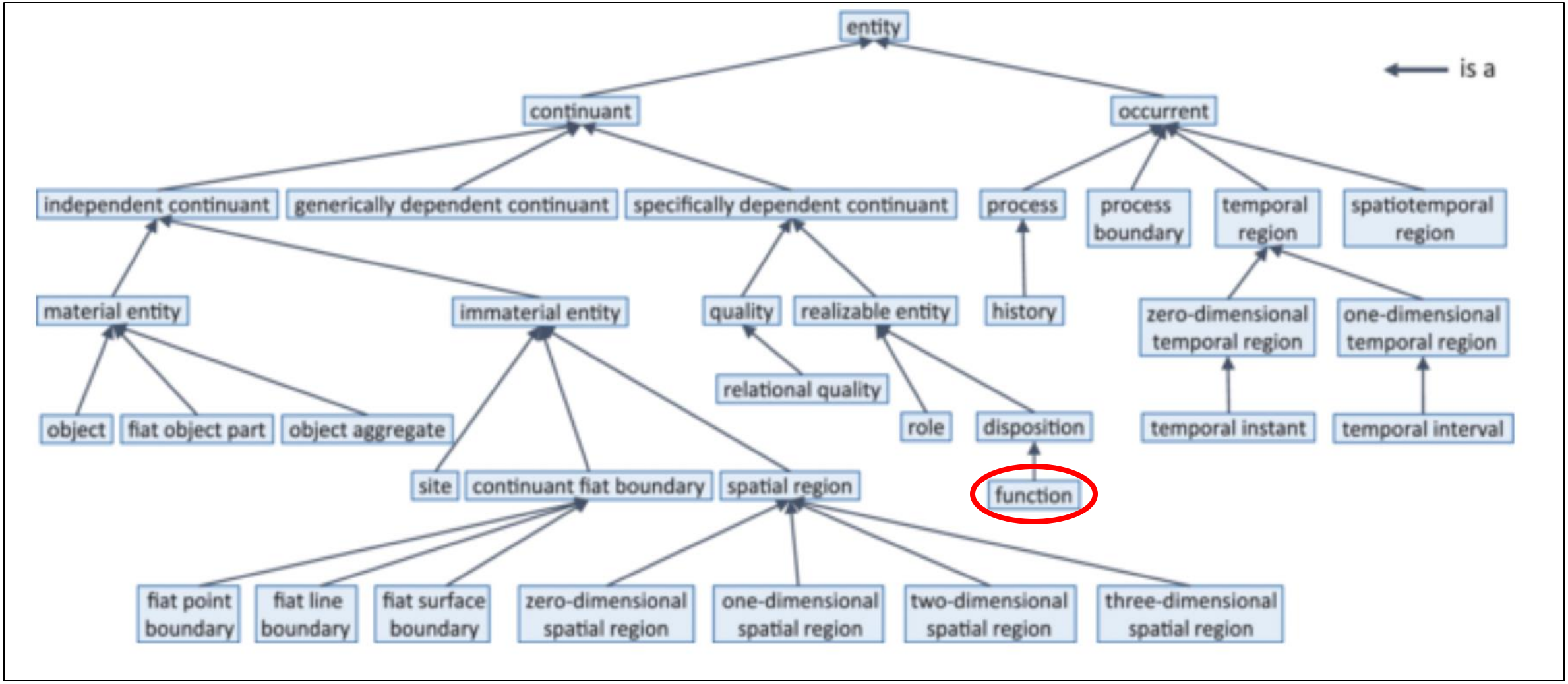
- Attributes of some material bearer that only become manifest under certain conditions

Role

A realizable entity that exists because there is some single bearer that is in some special physical, social, or institutional set of circumstances in which this bearer does not have to be, and is not such that, if it ceases to exist, then the physical make-up of the bearer is thereby changed.

EXTERNALLY GROUNDED

Function


entity
is a
continuant
occurrent
independent continuant
generically dependent continuant
specifically dependent continuant
process
process boundary
temporal region
spatiotemporal region
material entity
immaterial entity
quality
realizable entity
history
zero-dimensional temporal region
one-dimensional temporal region
relational quality
object
fiat object part
object aggregate
role
disposition
temporal instant
temporal interval
site
continuant fiat boundary
spatial region
function
fiat point boundary
fiat line boundary
fiat surface boundary
zero-dimensional spatial region
one-dimensional spatial region
two-dimensional spatial region
three-dimensional spatial region

Function

- Attributes of some material bearer that only become manifest under certain conditions

Function

A disposition that exists in virtue of the bearer's physical make-up and this physical make-up is something the bearer possesses because it came into being, either through evolution (in the case of natural biological entities) or through intentional design (in the case of artefacts), in order to realize processes of a certain sort.

Function

- Attributes of some material bearer that only become manifest under certain conditions

Function

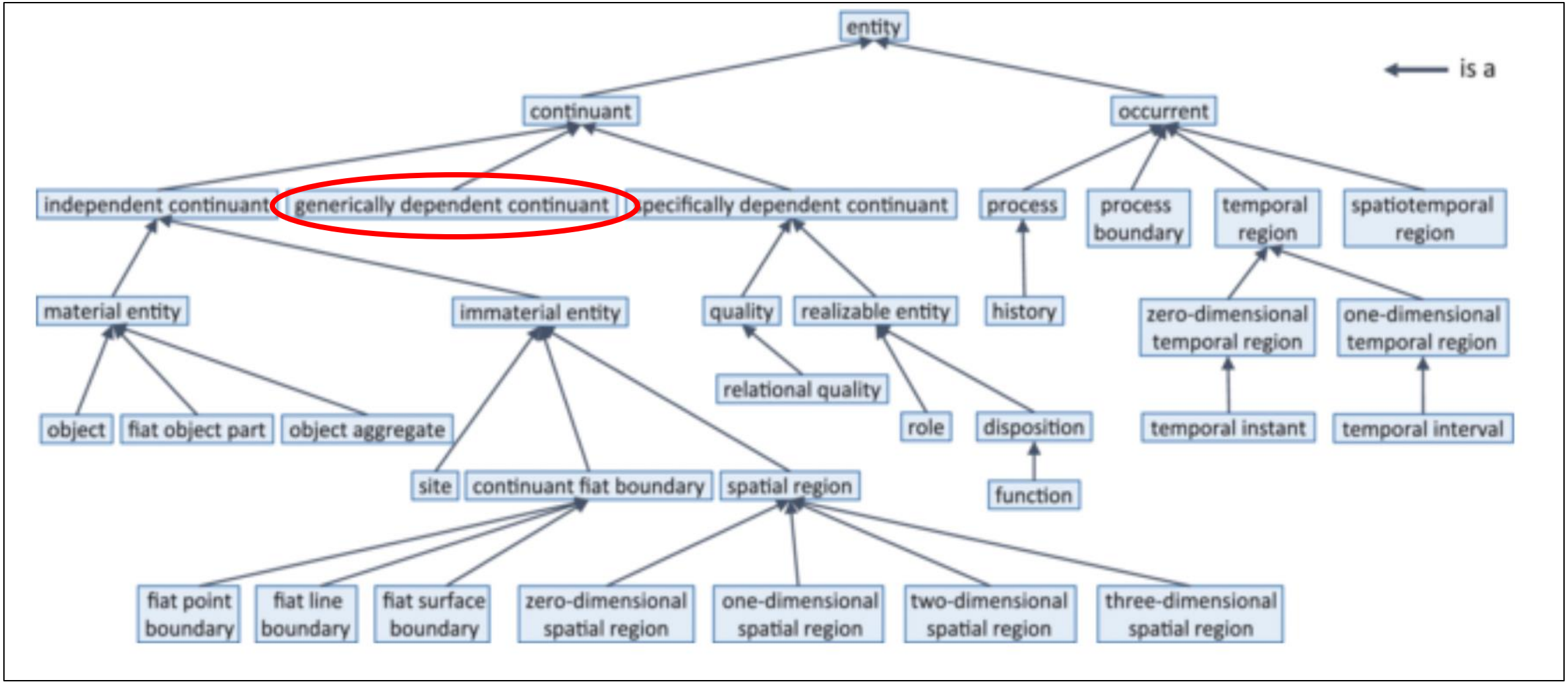
A disposition that exists in virtue of the bearer's physical make-up and this physical make-up is something the bearer possesses because it came into being, either through evolution (in the case of natural biological entities) or through intentional design (in the case of artefacts), in order to realize processes of a certain sort.

PURPOSE GROUNDED

Dependence

- For certain entities, their existence depends on the existence of something else
- Other entities do not depend on any other entities for their existence
- The latter are categorized in BFO as **independent continuants**
- The former include **specifically dependent** and **generically dependent entities**, as well as **processes**

Generically Dependent Continuant



Pattern Recognition

- We're disposed to recognize patterns with our perceptual faculties…

Patterns

- There is a need to represent dependent entities that could migrate across bearers

- This need led to **generically dependent continuants**, continuants that are in some sense copyable, i.e. patterns

- For example, “Snow is white” and “Schnee ist weiß” may be used to express numerically identical content, i.e. the same pattern

Real Patterns

- Some patterns are necessarily **about** something; some patterns are not
- "Snow is white" expresses content that is **about** snow
- "cm" or "." are not necessarily **about** anything; they are nevertheless patterns
- Most **generically dependent entities** represented in BFO extensions are patterns that are **about** something

Aboutness

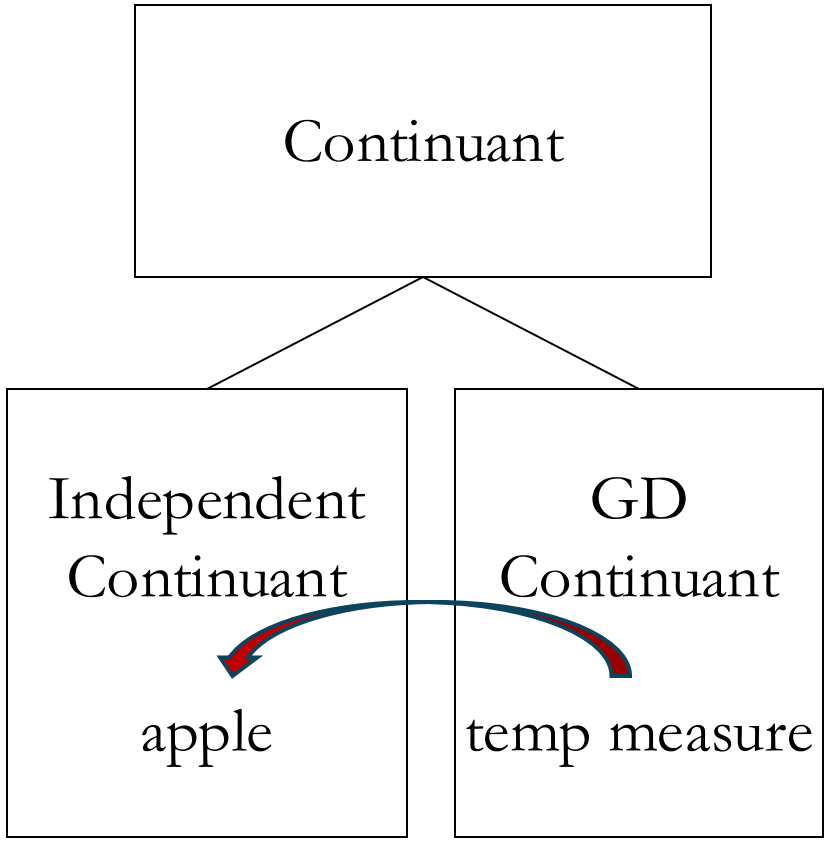
- **Information** is a pattern that is **about** something

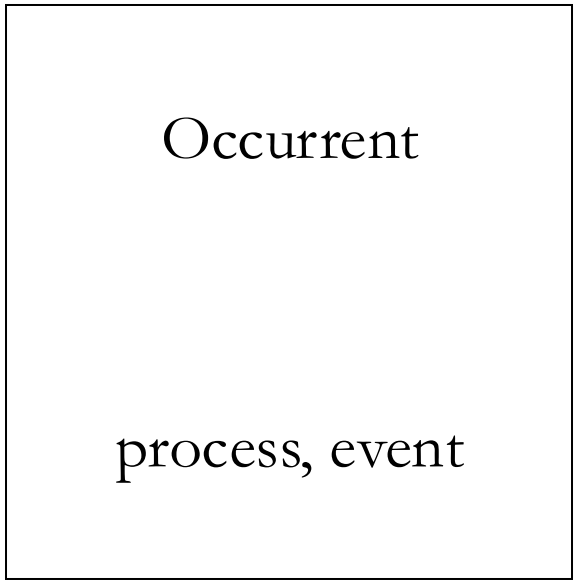
- In BFO extensions - such as the Information Artifact Ontology and the Information Entity Ontology - information is represented by the class **Information Content Entity**

- Where the "is about" relation is understood to be primitive:

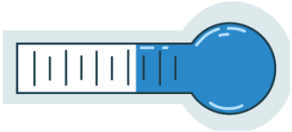
definition [language: en]
A primitive relationship between an Information Content Entity and some Entity.

Generically Dependent Continuant



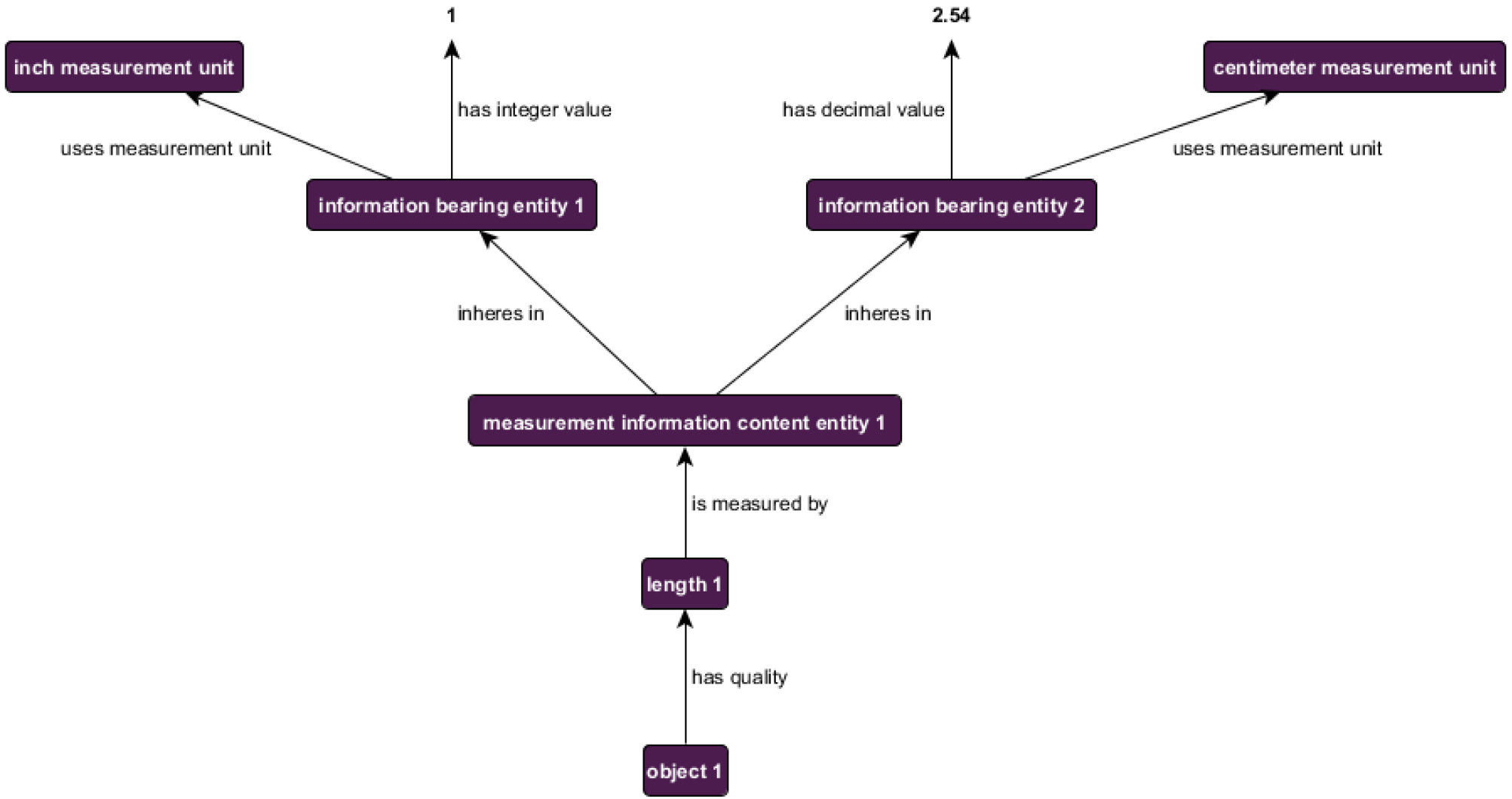


Many measures are about independent continuants

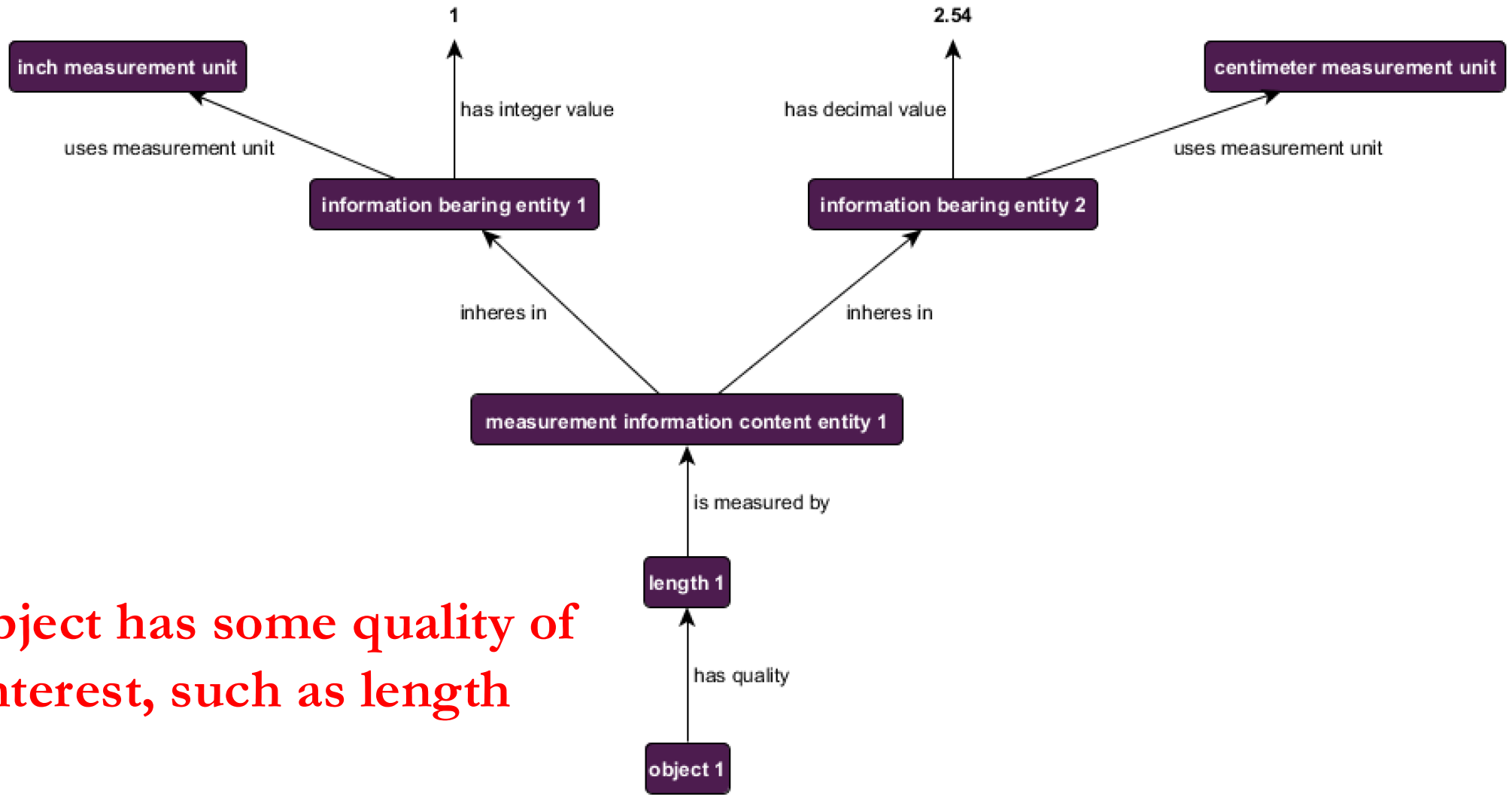


• • • • • •

CCO Measurement

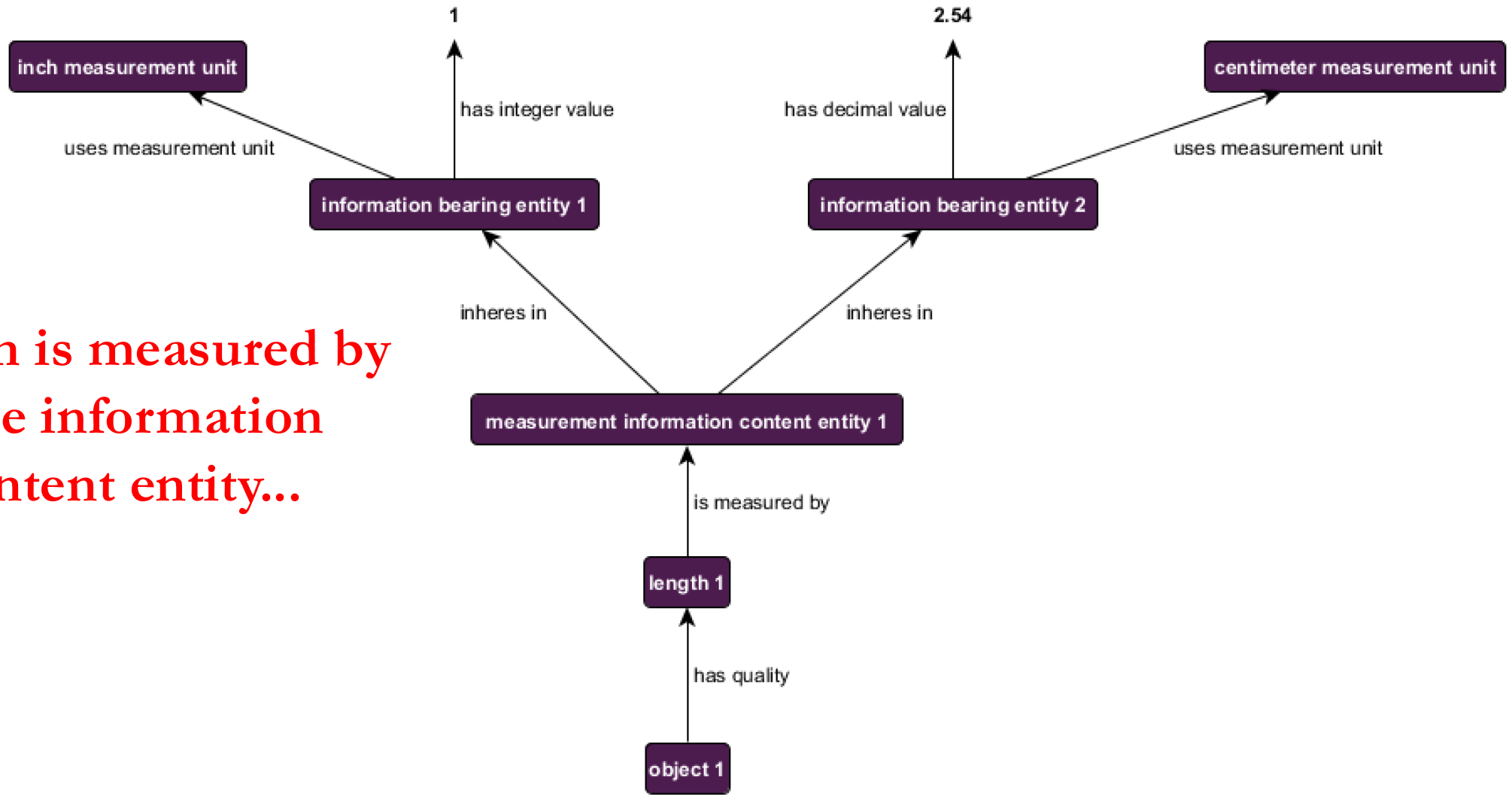


CCO Measurement



An object has some quality of interest, such as length

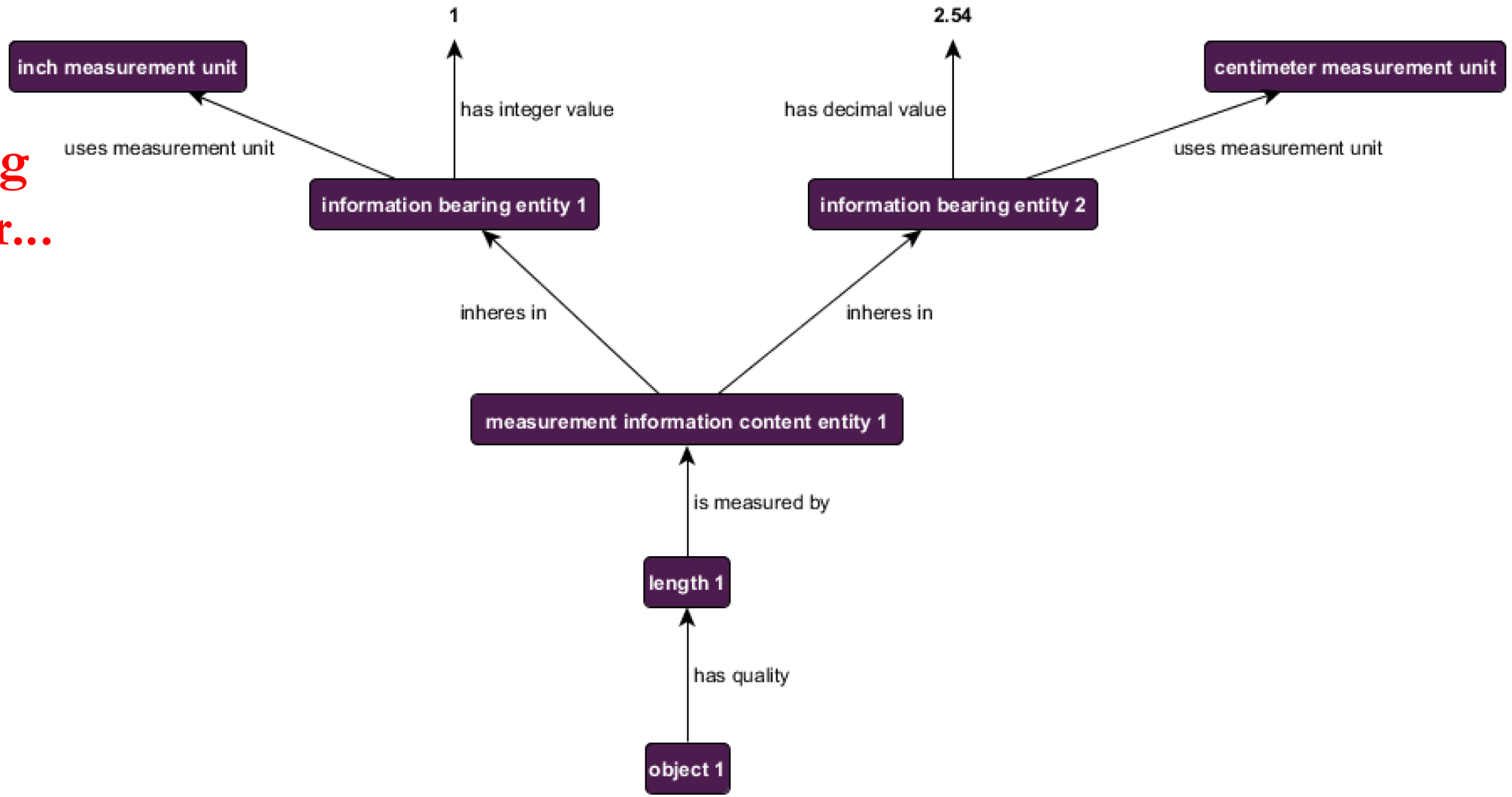
CCO Measurement



...which is measured by some information content entity...

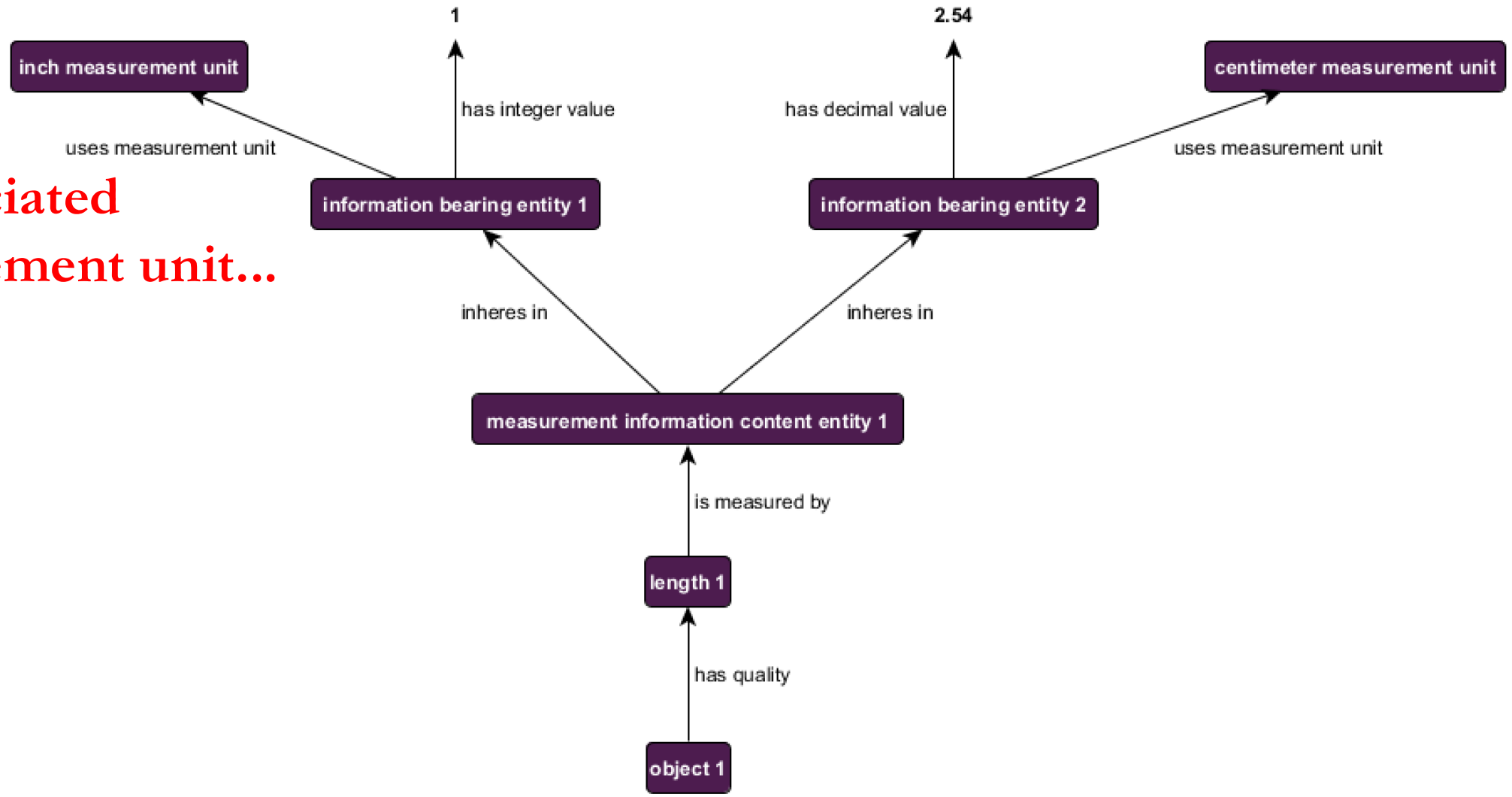
CCO Measurement

...inhering in a bearer...

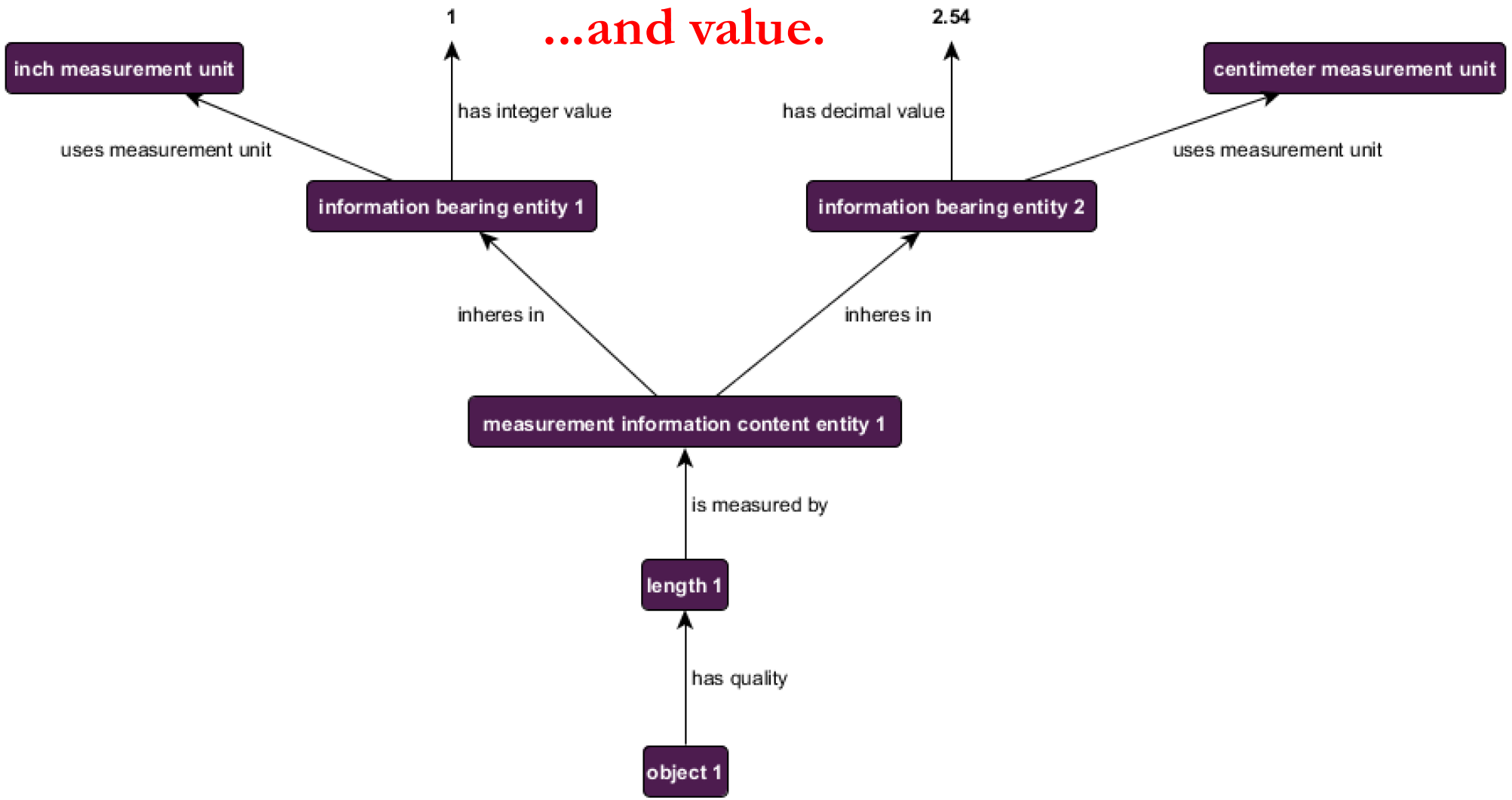


CCO Measurement

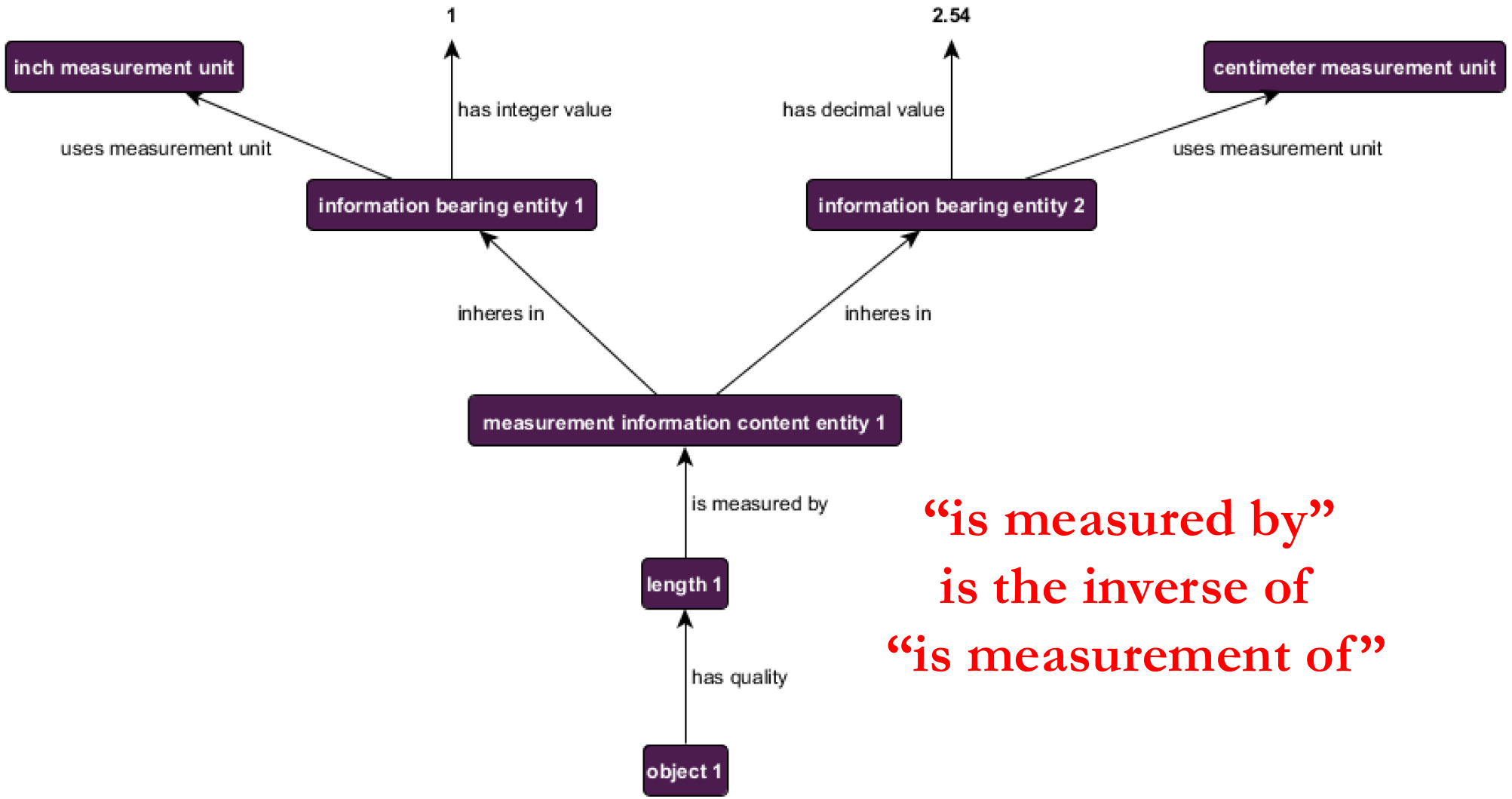
...with an associated measurement unit...



CCO Measurement

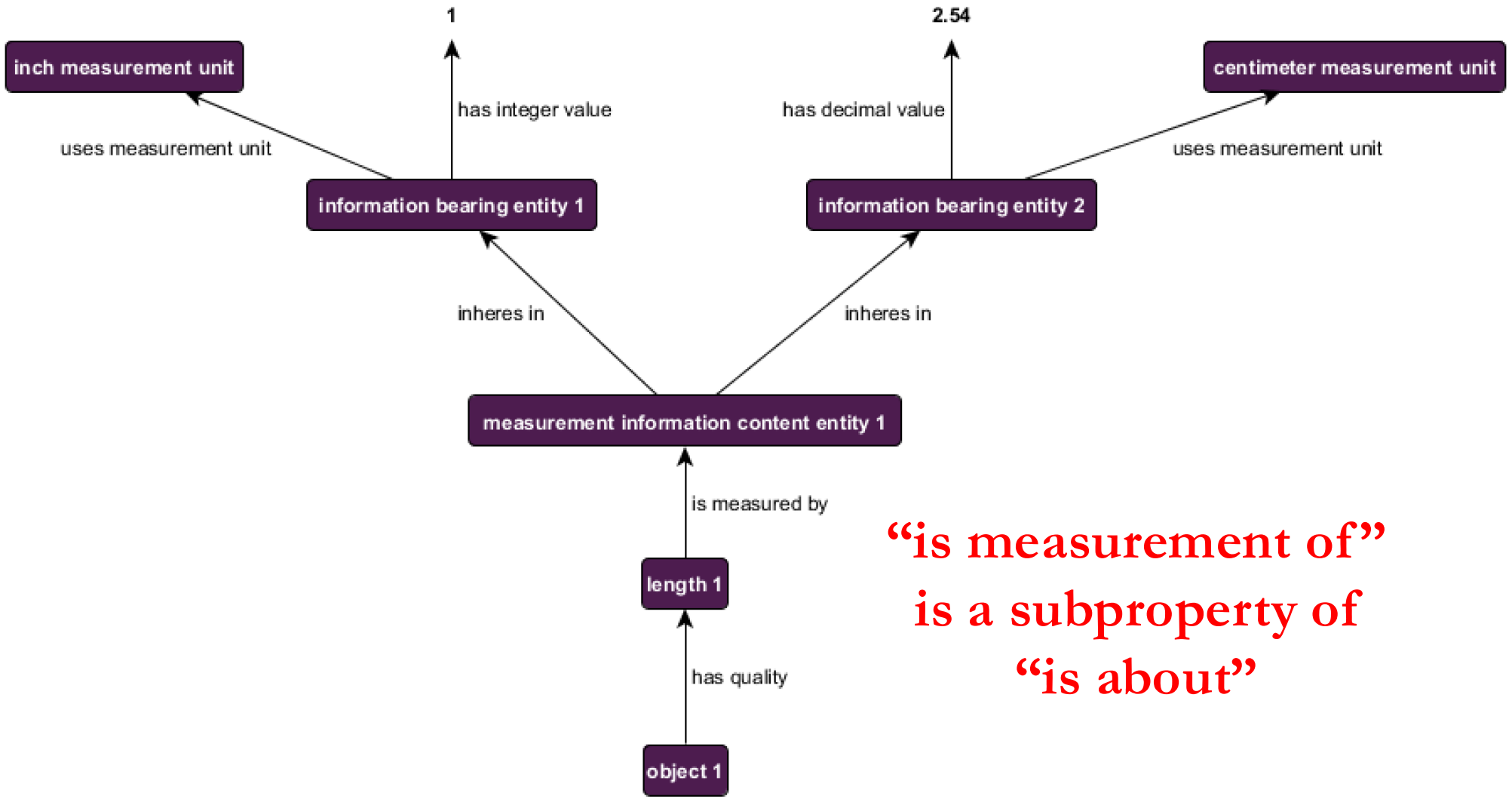


CCO Measurement



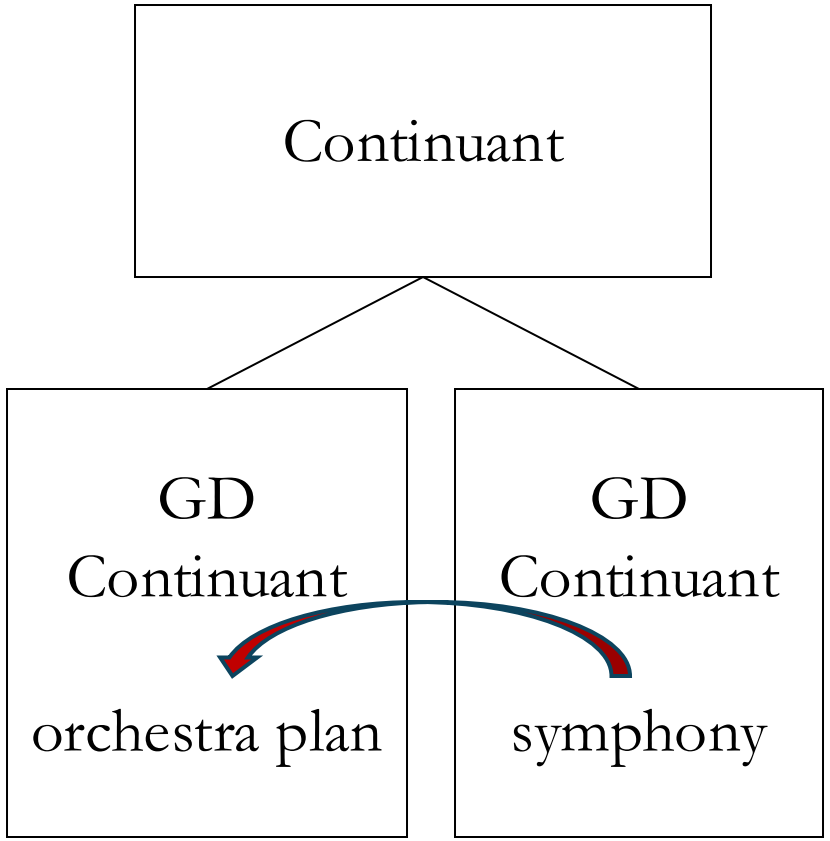
"is measured by"
is the inverse of
"is measurement of"

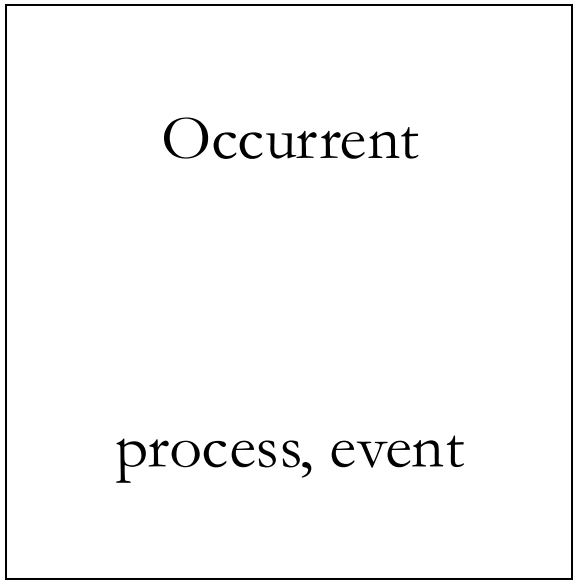
CCO Measurement



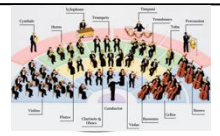
"is measurement of"
is a subproperty of
"is about"

Generically Dependent Continuant





GDCs may be about other GDCs

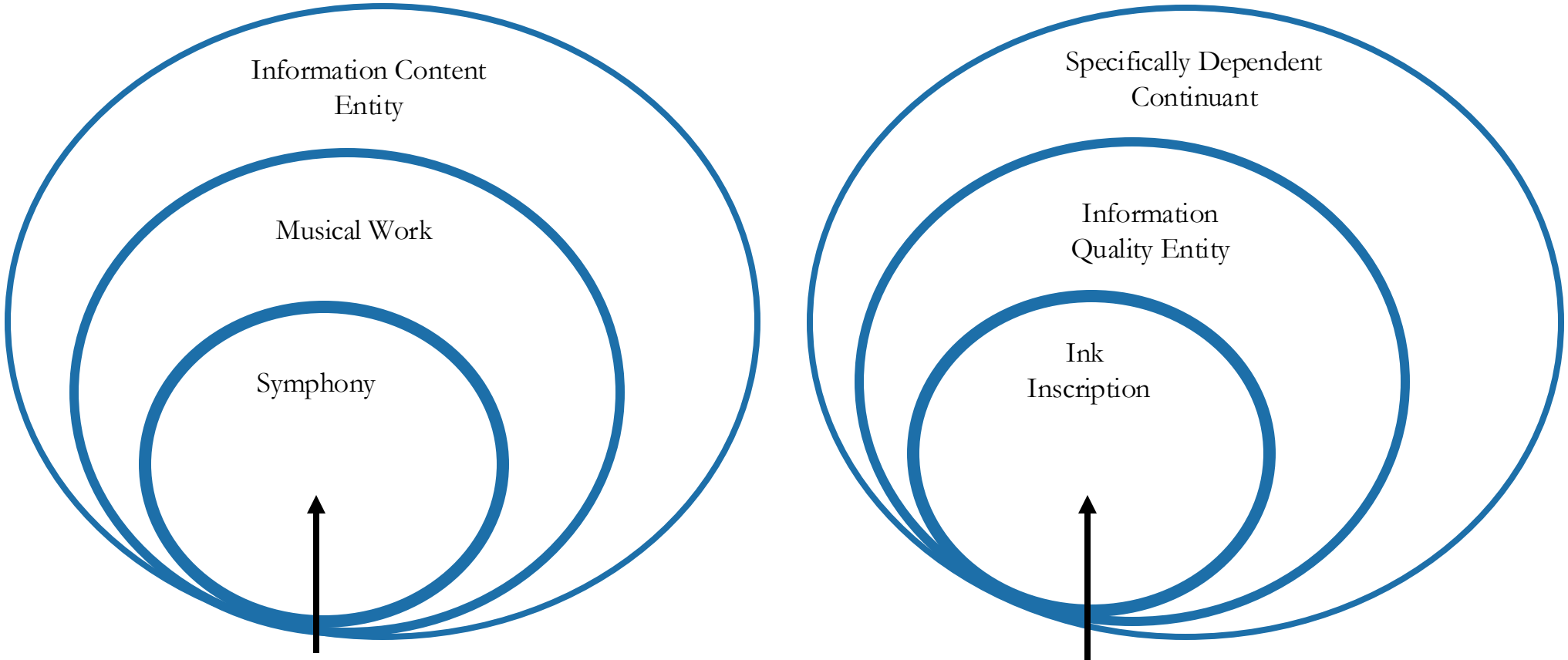


Beethoven's #9

• • • • • •

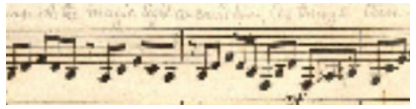
#9 Concretized in Ink Inscription

CLASSES



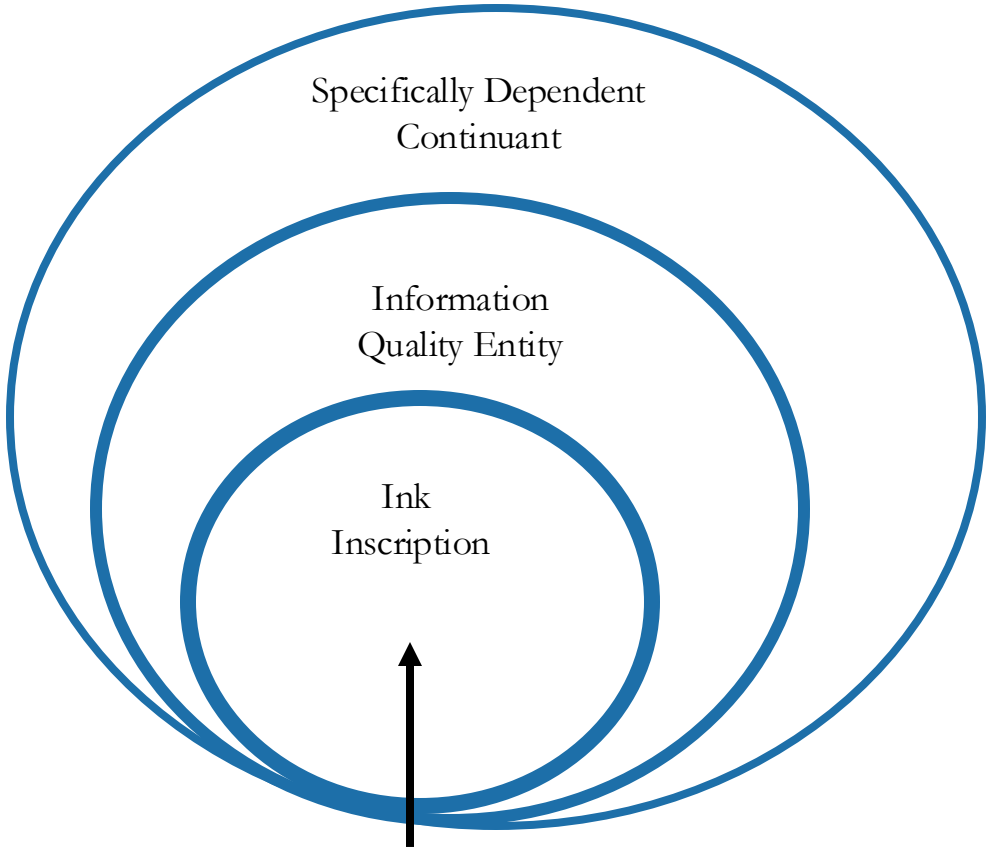
INSTANCES

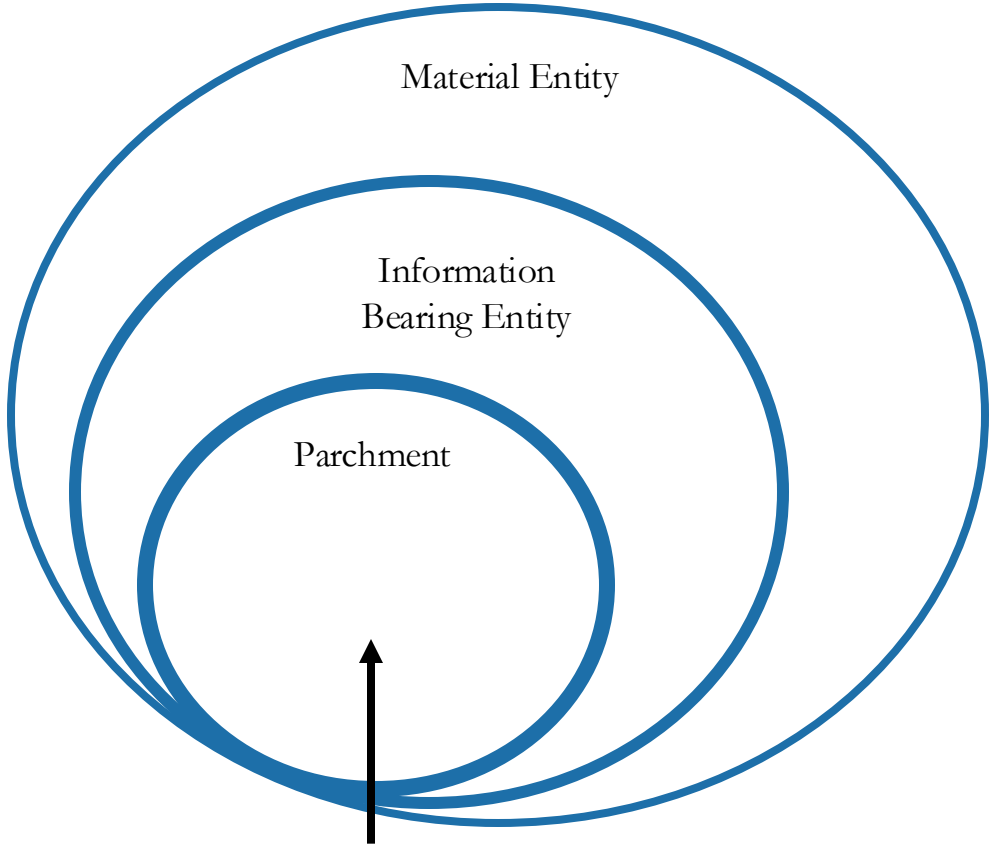
Beethoven's #9



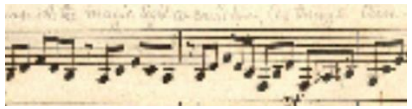
Ink Inscription inheres in Parchment

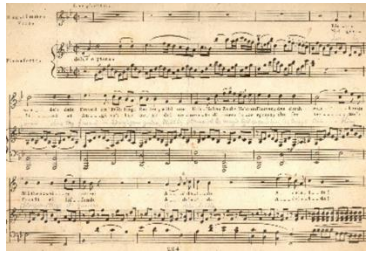
CLASSES





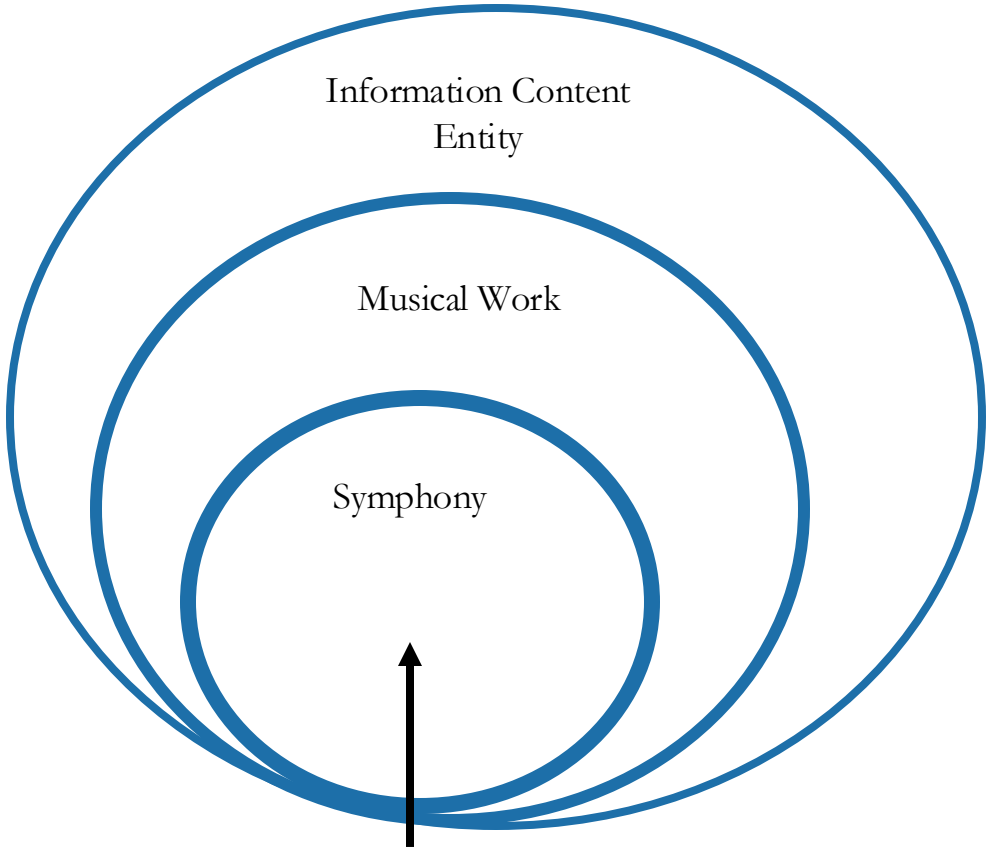
INSTANCES

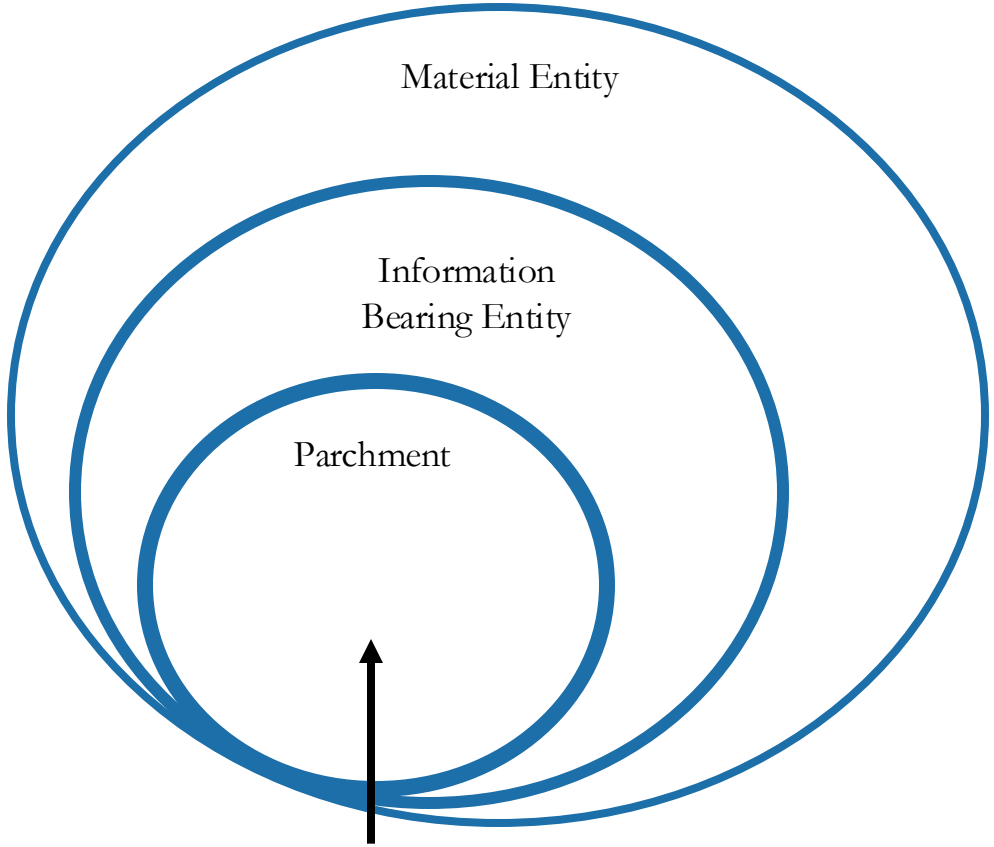




#9 Generically Dependent On Parchment

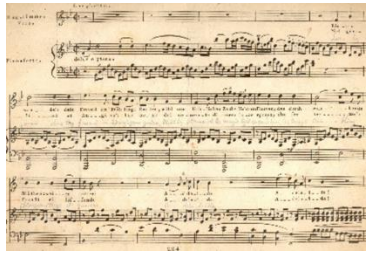
CLASSES





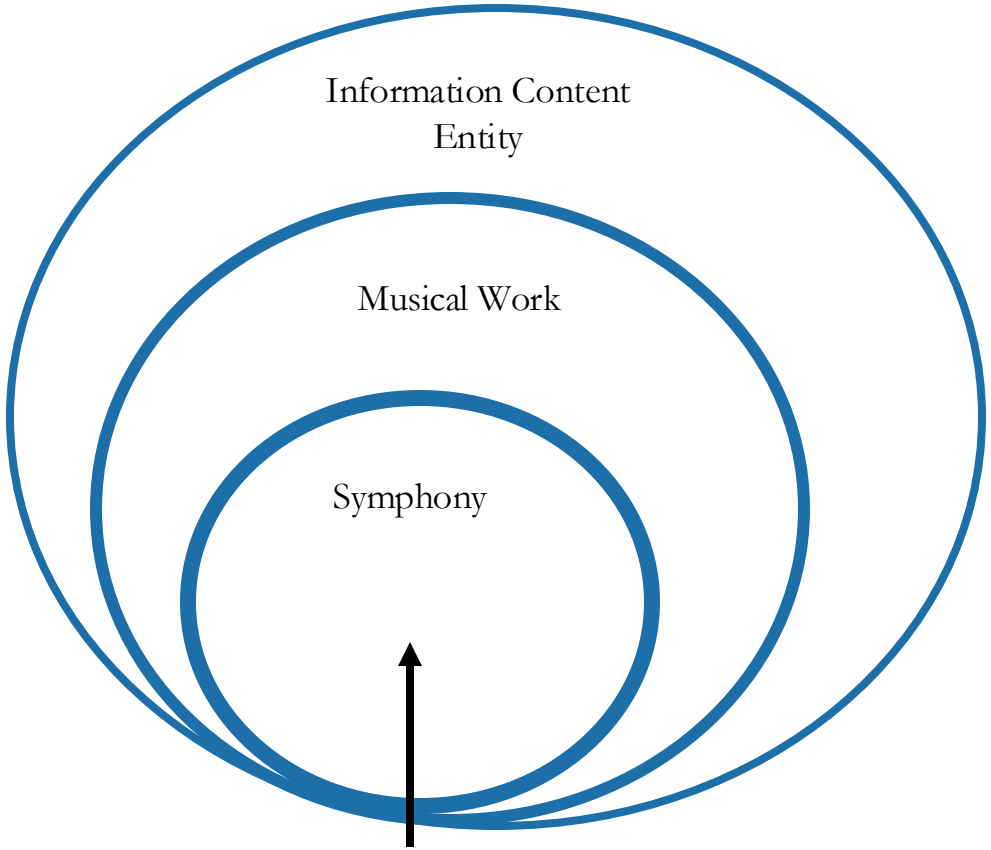
INSTANCES

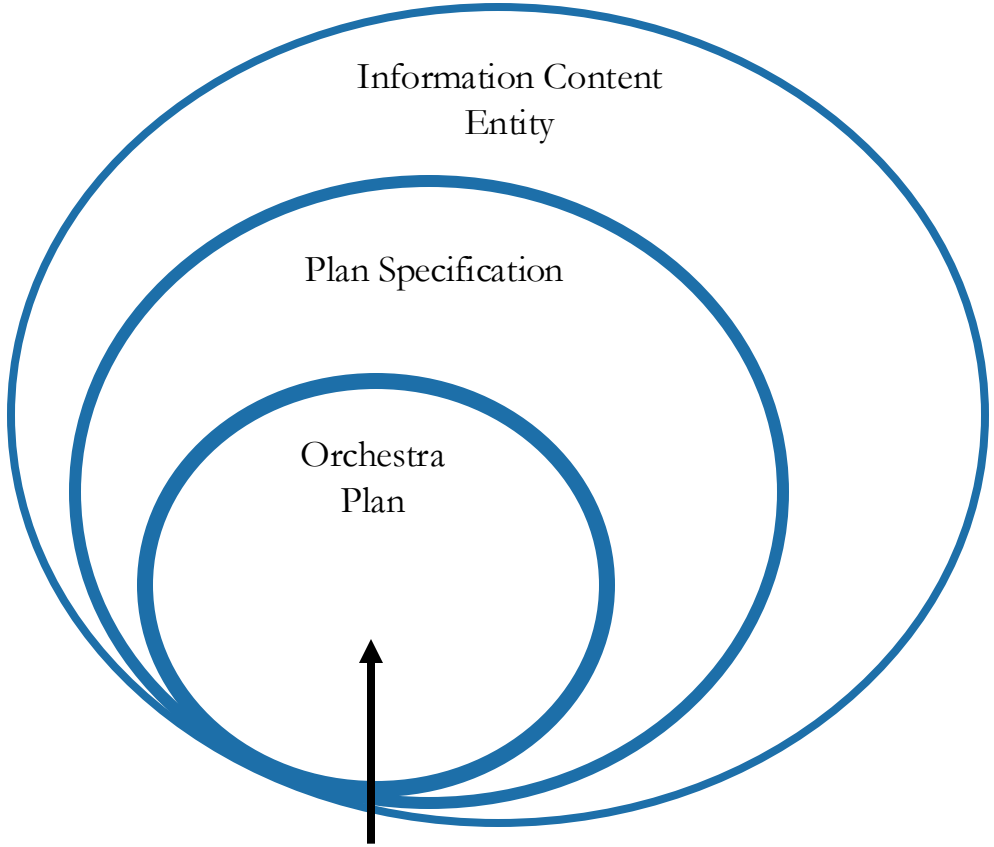
Beethoven's #9



#9 is about Orchestra Plan

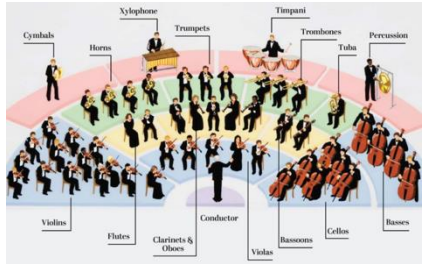
CLASSES





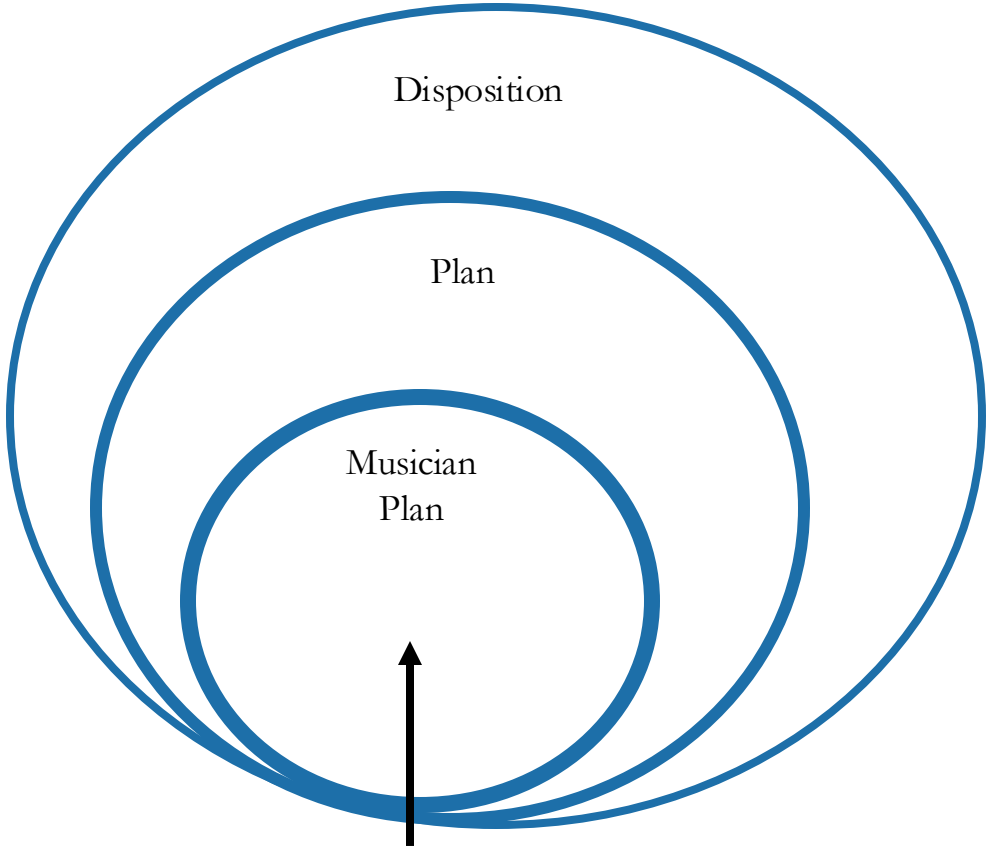
INSTANCES

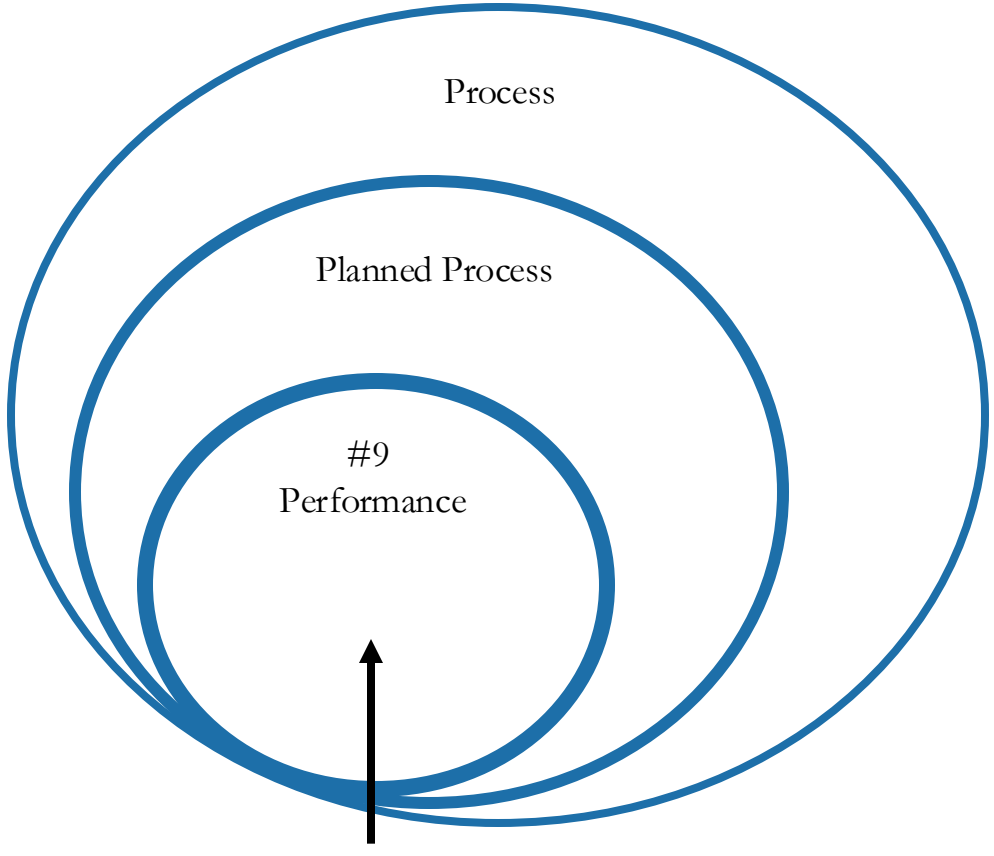
Beethoven's #9



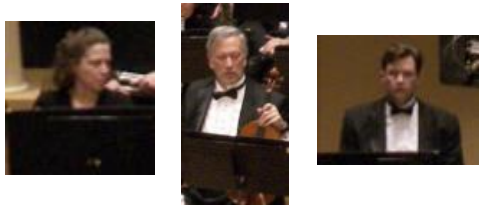
Musician Plan realized in #9 Performance

CLASSES





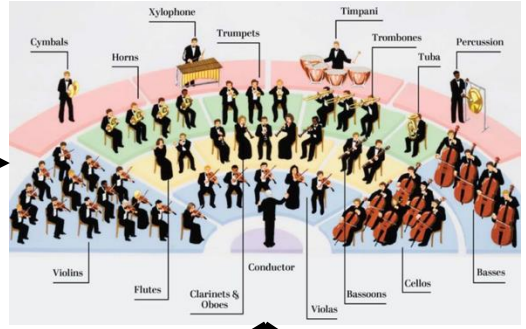
INSTANCES



Beethoven's #9th Symphony

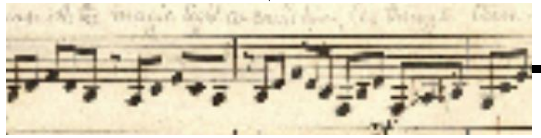
Beethoven's #9

is about

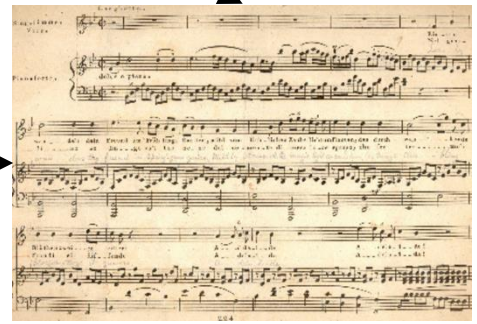


generically depends on

concretized in



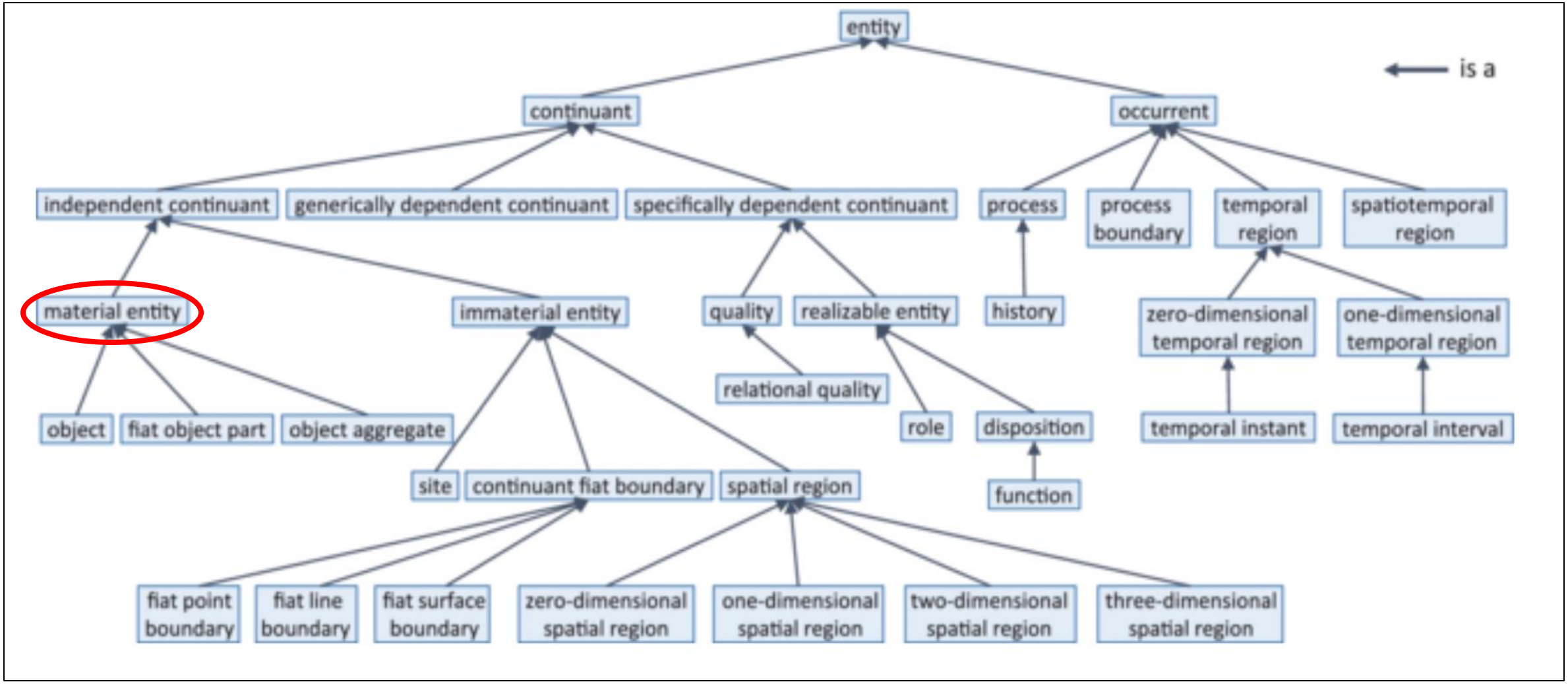
inheres in



concretized in

has realization

Material Entity



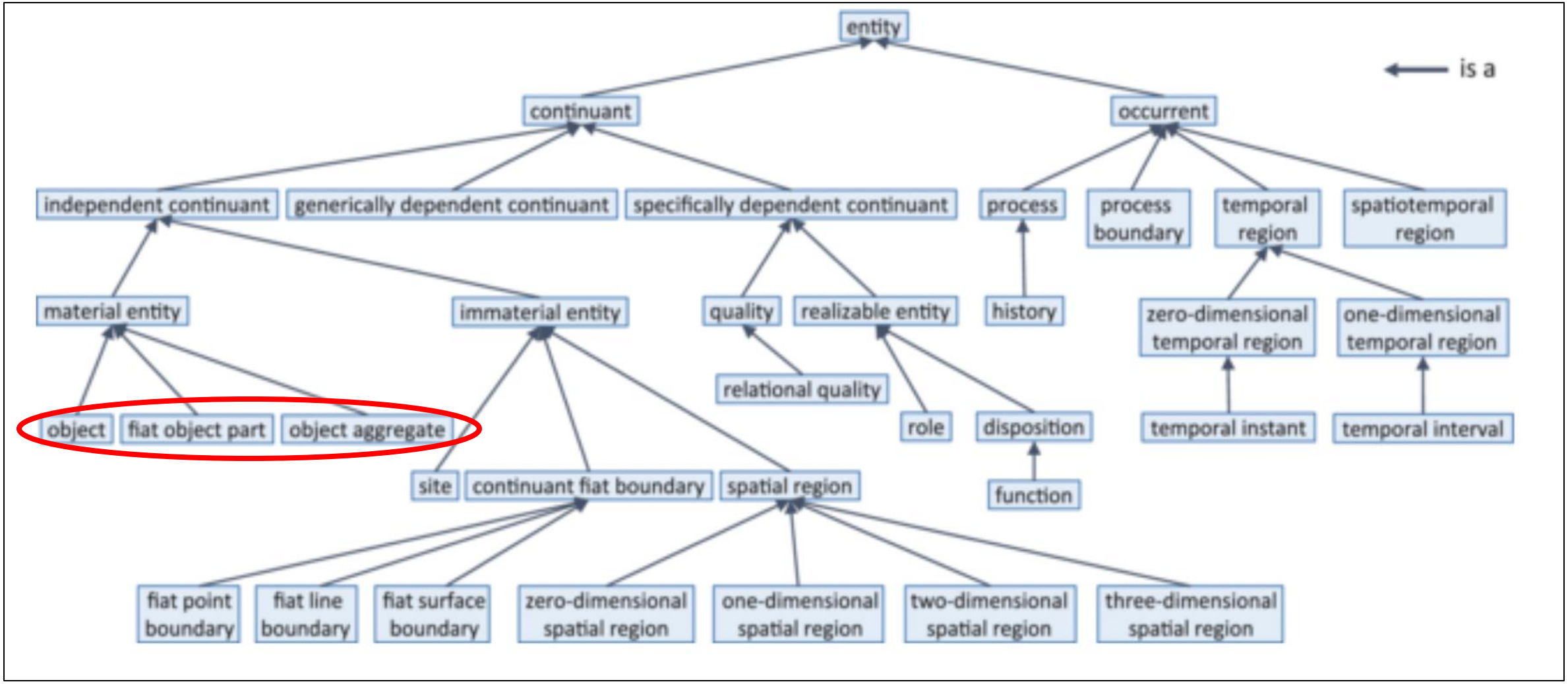
Material Entity

- Many independent continuants discussed thus far depend on instances falling under the class **Material Entity**, which includes all independent continuants having matter as part

- Apples, people, cars, blankets, viruses, tanks, etc. thus fall

- Subclasses include objects, object aggregates, and fiat object parts

Subclasses of Material Entity



Object

- An object is a material entity that manifests causal unity, where its instances are maximal with respect to that causal unity

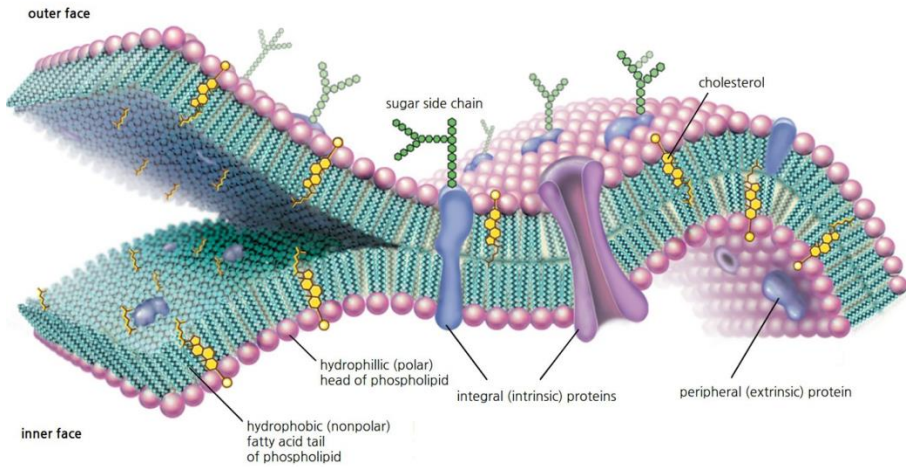
Object

- An object is a material entity that manifests **causal unity**, where its instances are maximal with respect to that causal unity

Object

- An object is a material entity that manifests **causal unity**, where its instances are maximal with respect to that causal unity

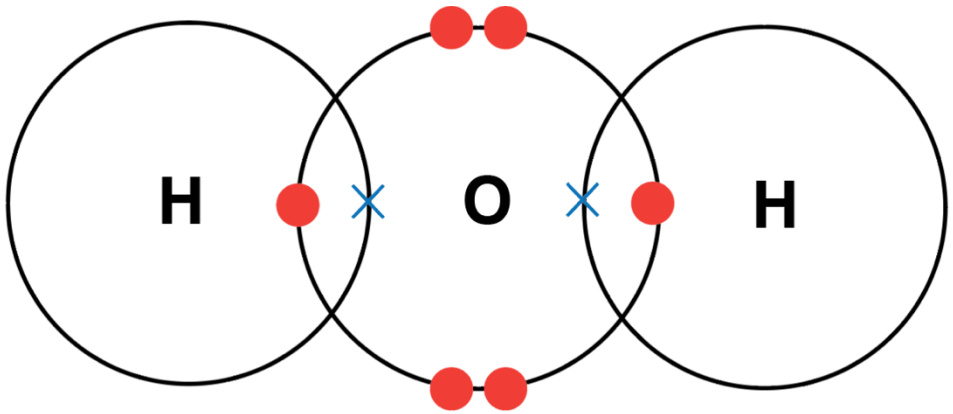
- Examples of causal unity:
  - Physical covering, e.g. interior of the object are covered by a connected membrane



Object

- An object is a material entity that manifests **causal unity**, where its instances are maximal with respect to that causal unity

- Examples of causal unity:
  - Physical covering, e.g. interior of the object are covered by a connected membrane
  - Internal forces, e.g. ionic bonds holding together molecules



Object

- An object is a material entity that manifests **causal unity**, where its instances are maximal with respect to that causal unity

- Examples of causal unity:
  - Physical covering, e.g. interior of the object are covered by a connected membrane
  - Internal forces, e.g. ionic bonds holding together molecules
  - Engineered assembly, e.g. mechanical assembly through screws or fasteners

Rule of Thumb

If moving a proper part of some material entity requires moving other material parts of that entity, there is likely causal unity between them

Object Aggregate

- Defined such that any and all members of the aggregate are objects which do not share any parts in common, i.e. are pairwise disjoint

- For example, one can define the object aggregate that is all instruments in an orchestra, or all members of a band

- More generally, the "X aggregate" is intended to be a recipe that may be applied to other classes, e.g. "aggregate of roles"

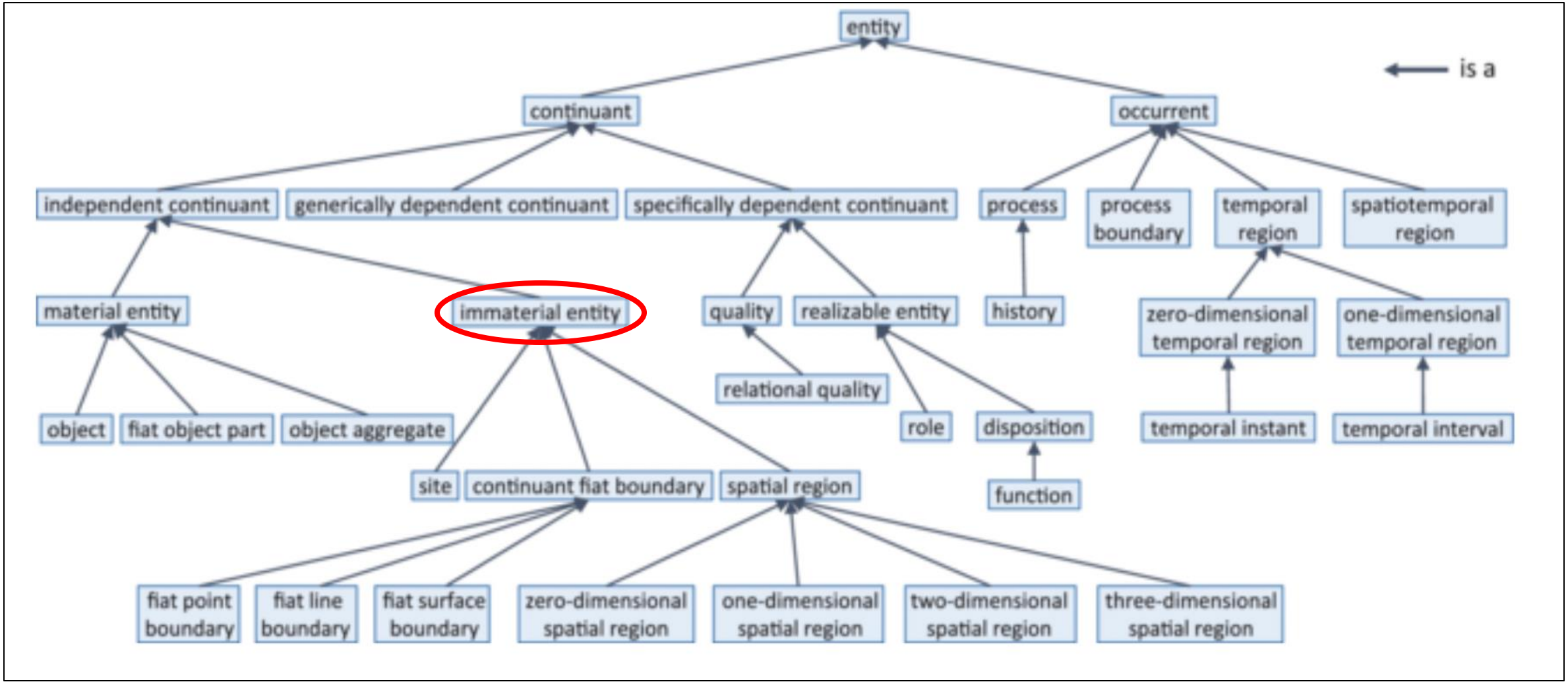
Fiat Object Part

- Certain parts of objects that are not themselves objects, warrant catergorization beyond merely being identified as parts

- For example, a so-called **bona fide** object part of the Earth, which would be an object, such as an island, may be divided into northern and southern **fiat** object parts

- Northern and southern portions of a given island exist regardless of whether we delineate them so

Immaterial Entity



Any entity that has a material entity as part is a material entity

But material entities may have immaterial entities as parts

Immaterial Entity

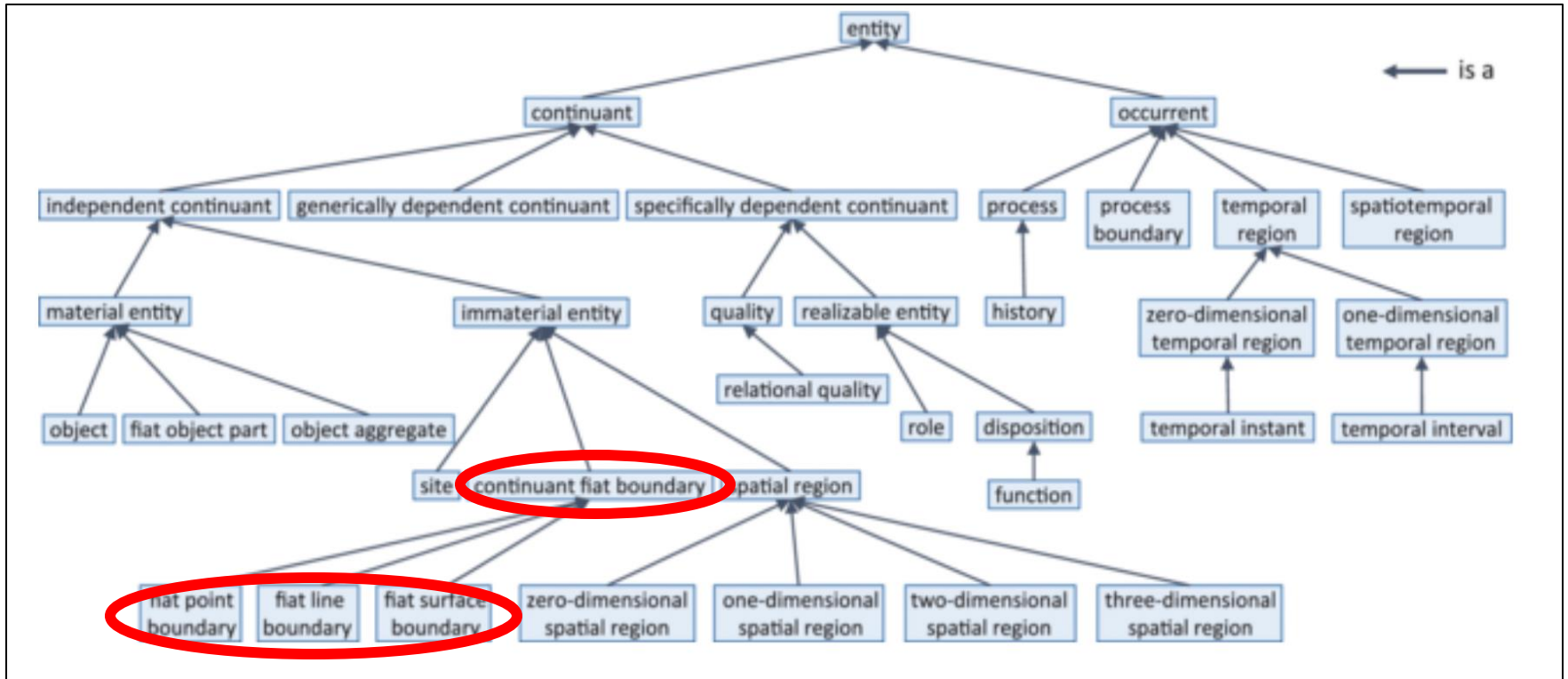
- Not all independent continuants have matter as parts

- Territorial boundaries, internal hulls of ships, interiors of capsules, etc. are not identical to whatever material is often associated with them

- For example, an archaeologist seeking the site through which a contemporary river used to flow, is not looking for the material the river used to flow through, for that is lost to time

Continuant Fiat Boundary

- *Continuant Fiat Boundary* $=_{\text{def}}$ An immaterial entity such that there is no time t when it has a spatial region as continuant part & whose location is determined in relation to some material entity



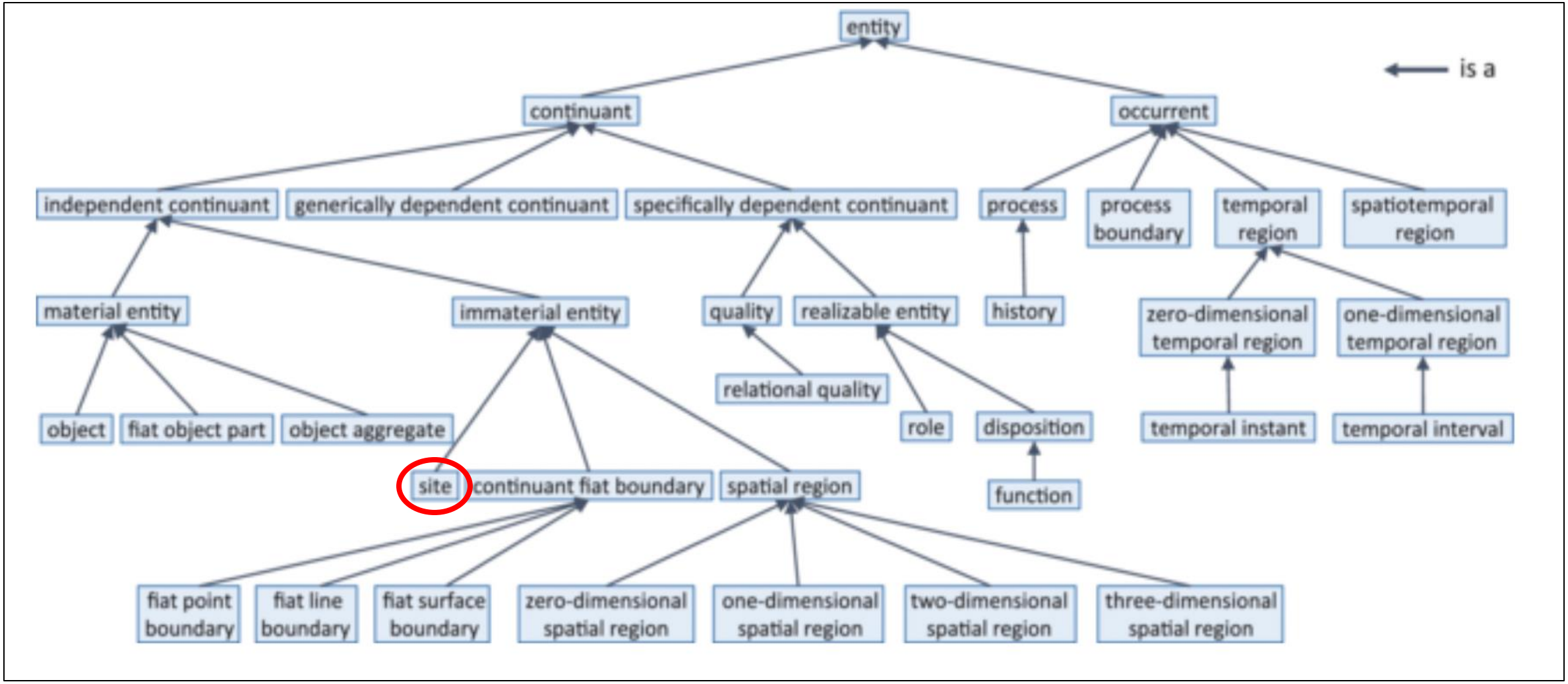
Continuant Fiat Boundary

- *Continuant Fiat Boundary* $=_{\text{def}}$ An immaterial entity such that there is no time t when it has a spatial region as continuant part & whose location is determined in relation to some material entity

- In BFO, **objects** are three-dimensional and have two-dimensional boundaries, e.g. surfaces

- There are no three-dimensional boundaries, because boundaries are always entities of some lower dimension

Site


entity
is a
continuant
occurrent
independent continuant
generically dependent continuant
specifically dependent continuant
process
process boundary
temporal region
spatiotemporal region
material entity
immaterial entity
quality
realizable entity
history
zero-dimensional temporal region
one-dimensional temporal region
relational quality
object
fiat object part
object aggregate
role
disposition
temporal instant
temporal interval
site
continuant fiat boundary
spatial region
function
fiat point boundary
fiat line boundary
fiat surface boundary
zero-dimensional spatial region
one-dimensional spatial region
two-dimensional spatial region
three-dimensional spatial region

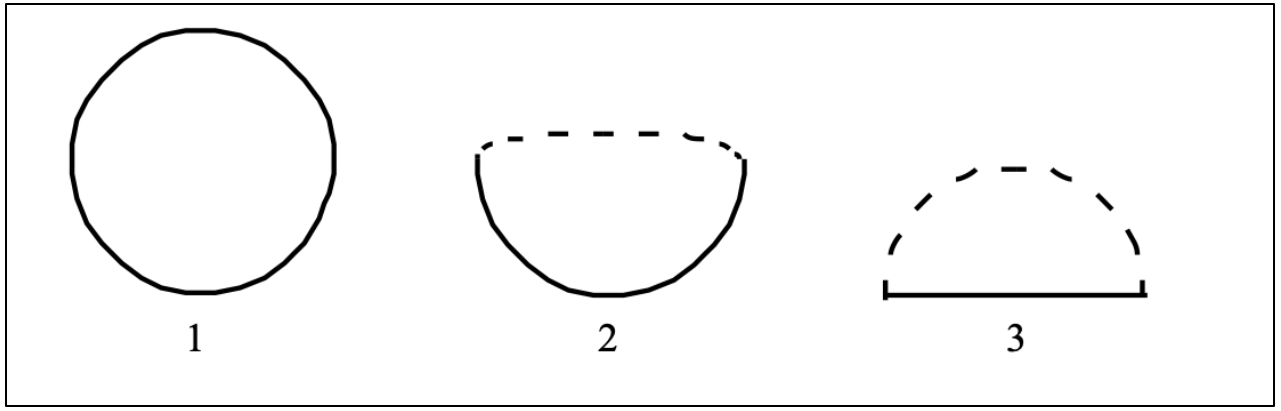
Site

- *Site* $=_{\text{def}}$ A three-dimensional immaterial entity whose boundaries either (partially or wholly) coincide with the boundaries of one or more material entities or have locations determined in relation to some material entity

Site

- *Site* $=_{\text{def}}$ A three-dimensional immaterial entity whose boundaries either (partially or wholly) coincide with the boundaries of one or more material entities or have locations determined in relation to some material entity

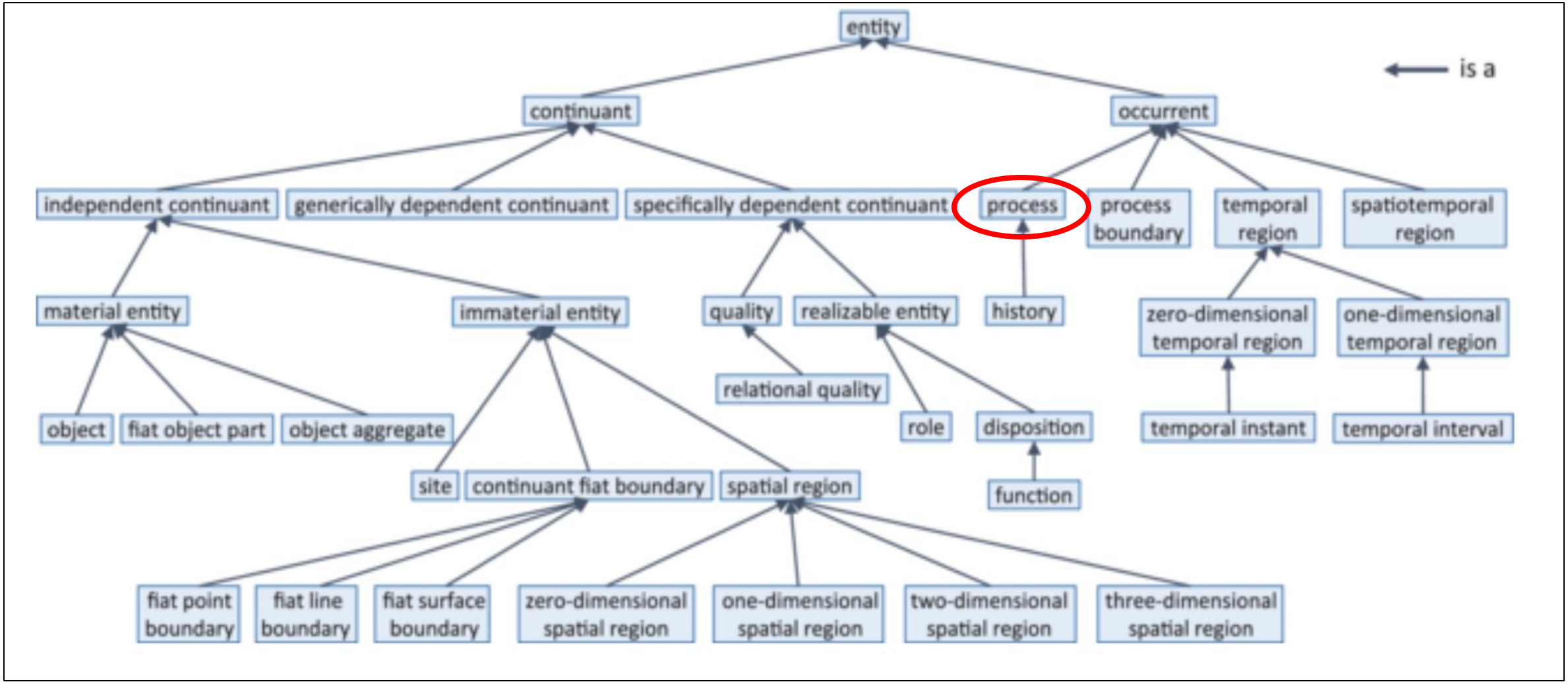
- Examples:
  - A rabbit hole
  - The interior of your bedroom
  - The hold of a ship
  - The cockpit of an aircraft



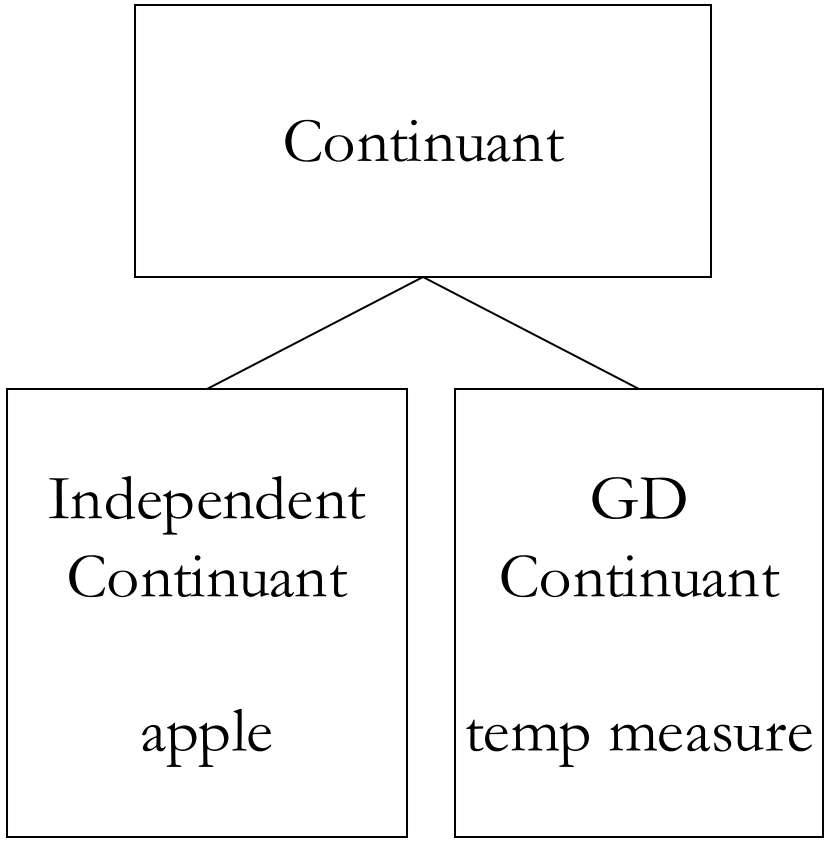
Dependence

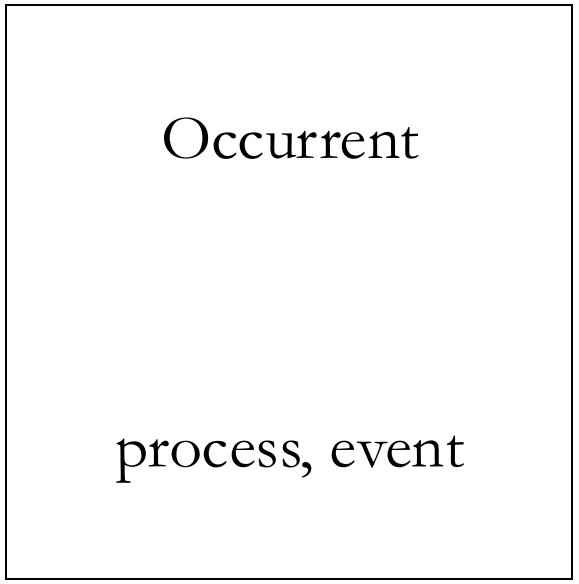
- For certain entities, their existence depends on the existence of something else
- Other entities do not depend on any other entities for their existence
- The latter are categorized in BFO as **independent continuants**
- The former include **specifically dependent** and **generically dependent entities**, as well as **processes**

Process

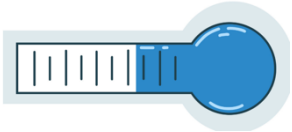

entity
is a
continuant
occurrent
independent continuant
generically dependent continuant
specifically dependent continuant
process
process boundary
temporal region
spatiotemporal region
material entity
immaterial entity
quality
realizable entity
history
zero-dimensional temporal region
one-dimensional temporal region
relational quality
object
fiat object part
object aggregate
role
disposition
temporal instant
temporal interval
site
continuant fiat boundary
spatial region
function
fiat point boundary
fiat line boundary
fiat surface boundary
zero-dimensional spatial region
one-dimensional spatial region
two-dimensional spatial region
three-dimensional spatial region

Process



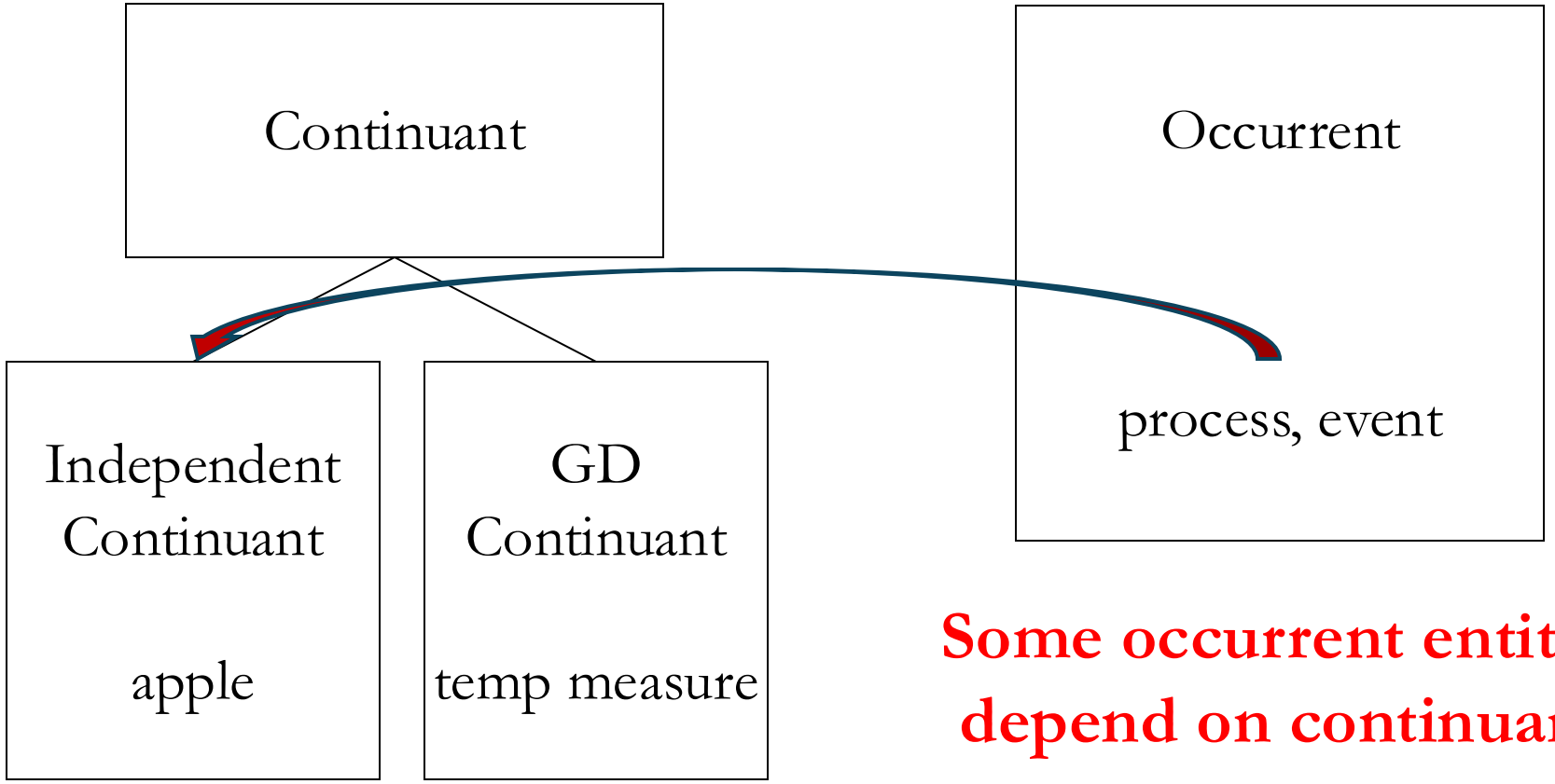


Some occurrent entities depend on continuants

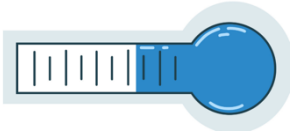
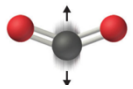
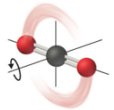


• • • • • •

Process



Some occurrent entities depend on continuants

Processes

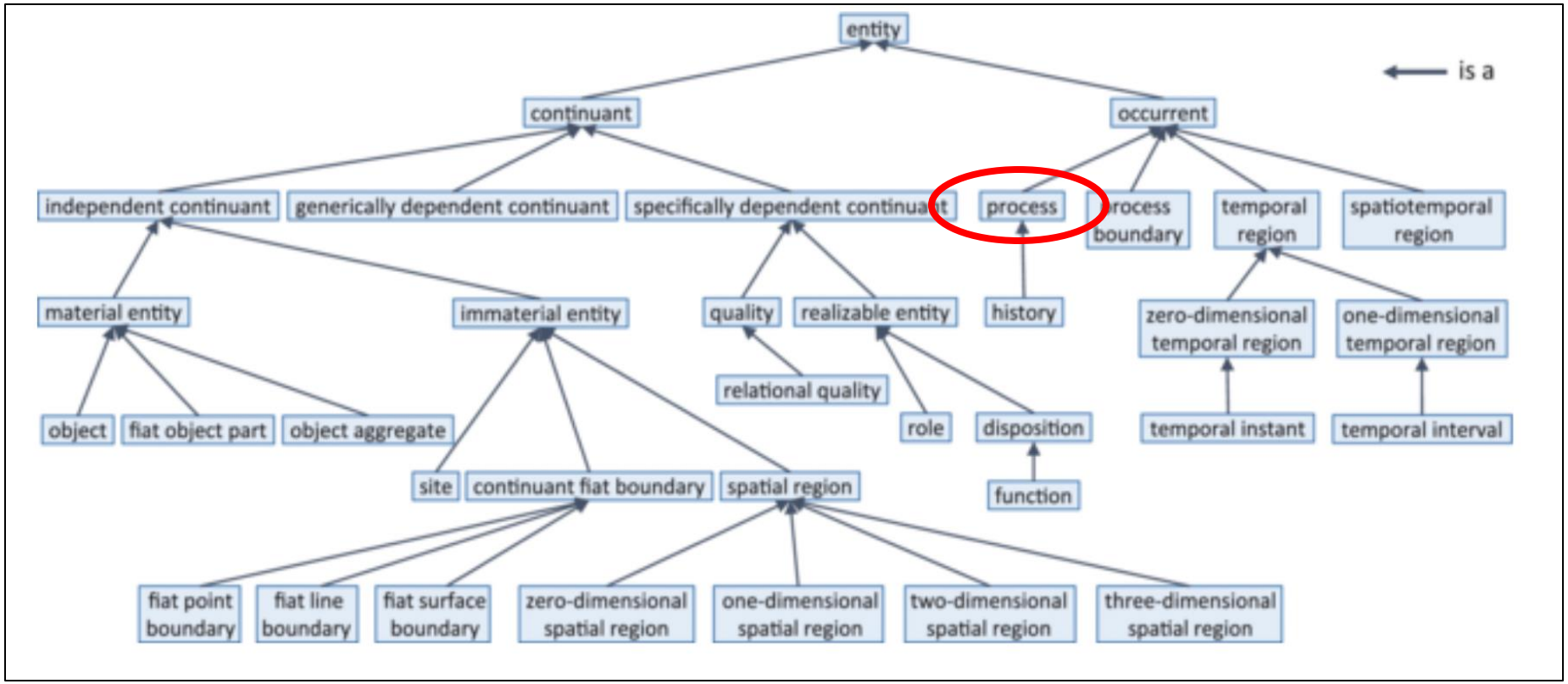
- Are where happenings live…

- All processes in BFO have at least one **temporal part** and are such that there is some **material entity** which **participates in** the process

- **participates in** is a minimal relationship connecting specifically, generically, and independent continuants to **process**

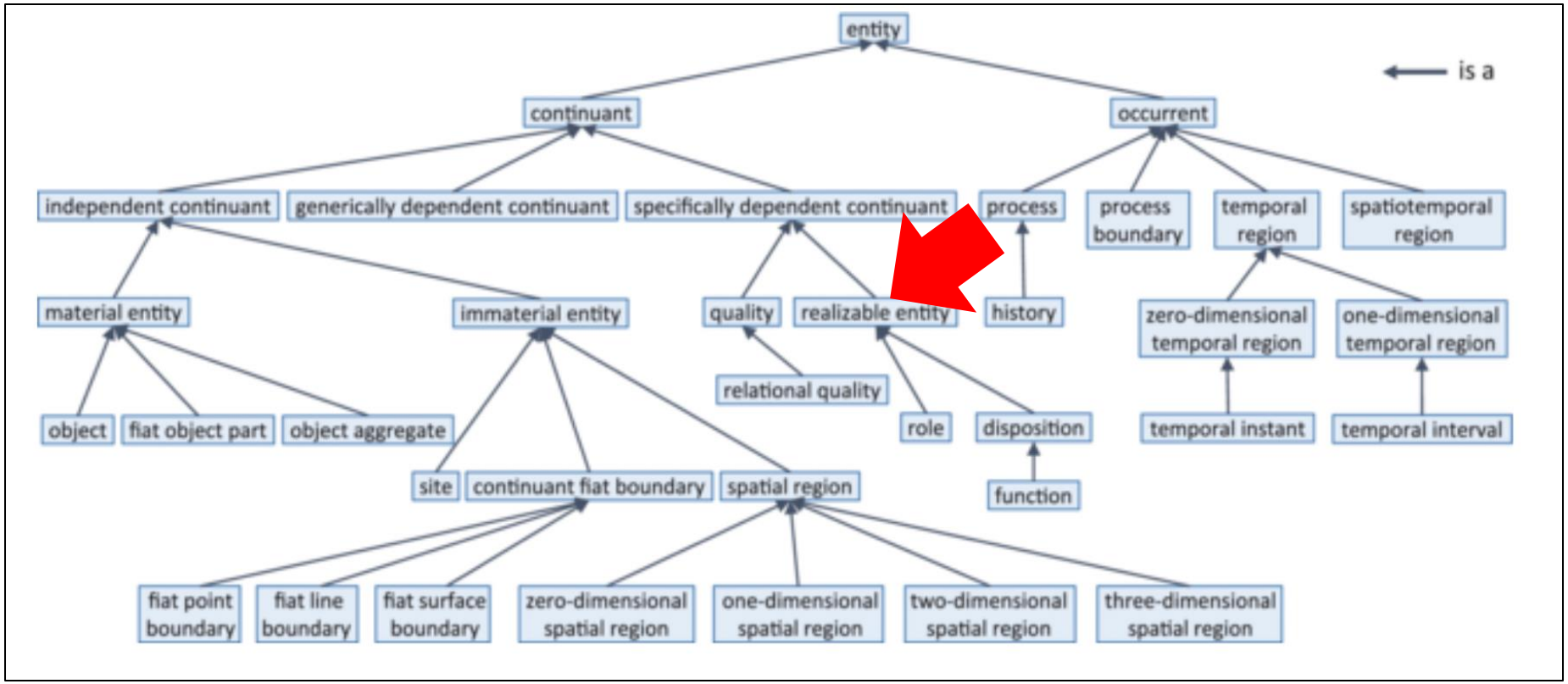
Dependency Chain

- A given **process** may have realization some realizable entity, which inheres in some independent continuant



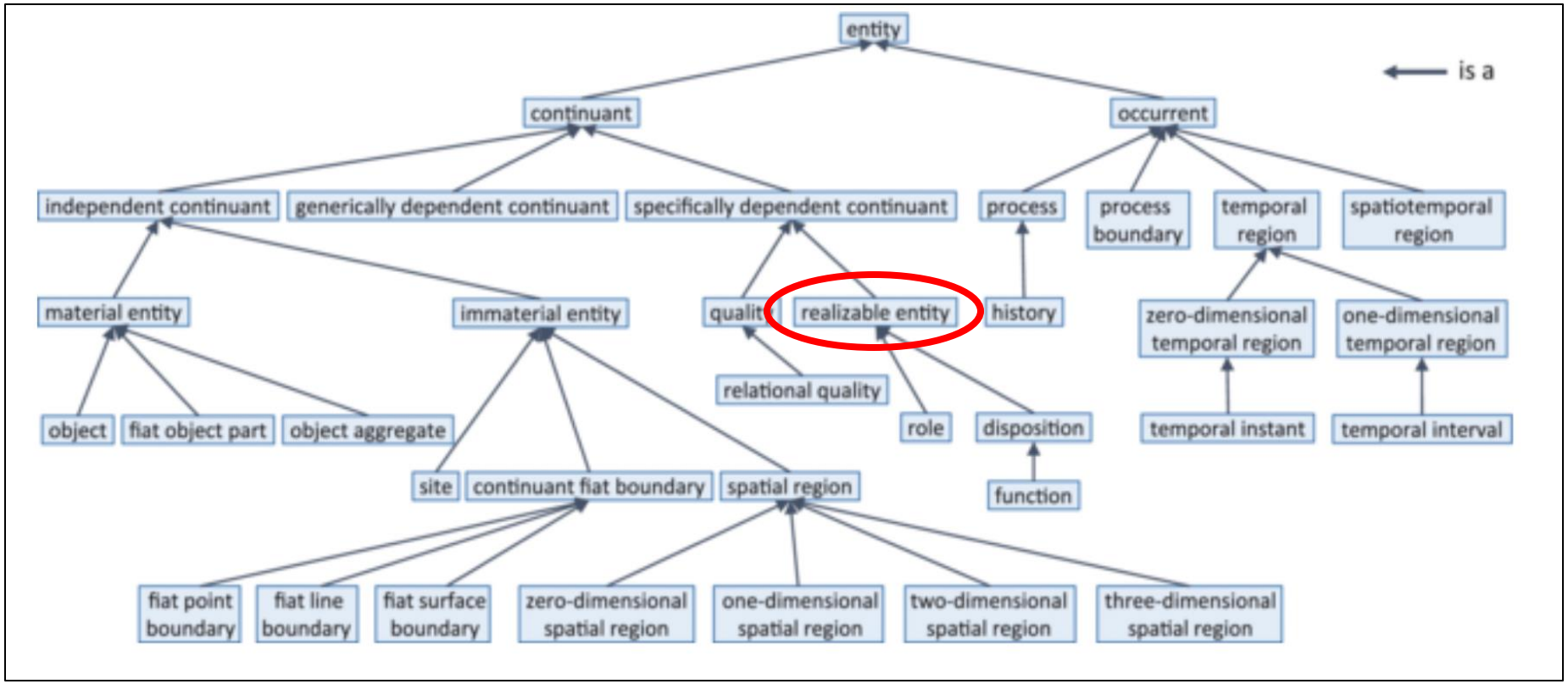
Dependency Chain

- A given **process** may **have realization** some realizable entity, which inheres in some independent continuant



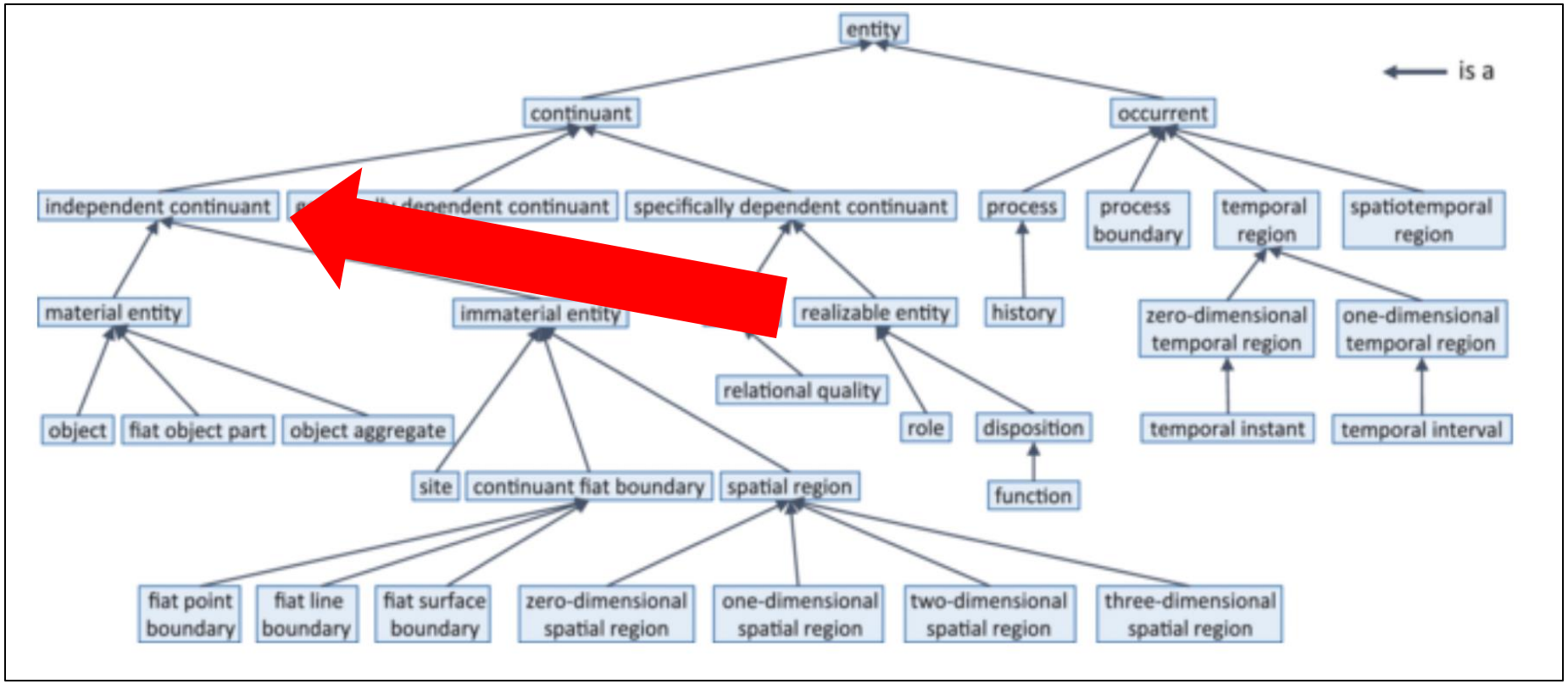
Dependency Chain

- A given **process** may **have realization** some **realizable entity**, which inheres in some independent continuant



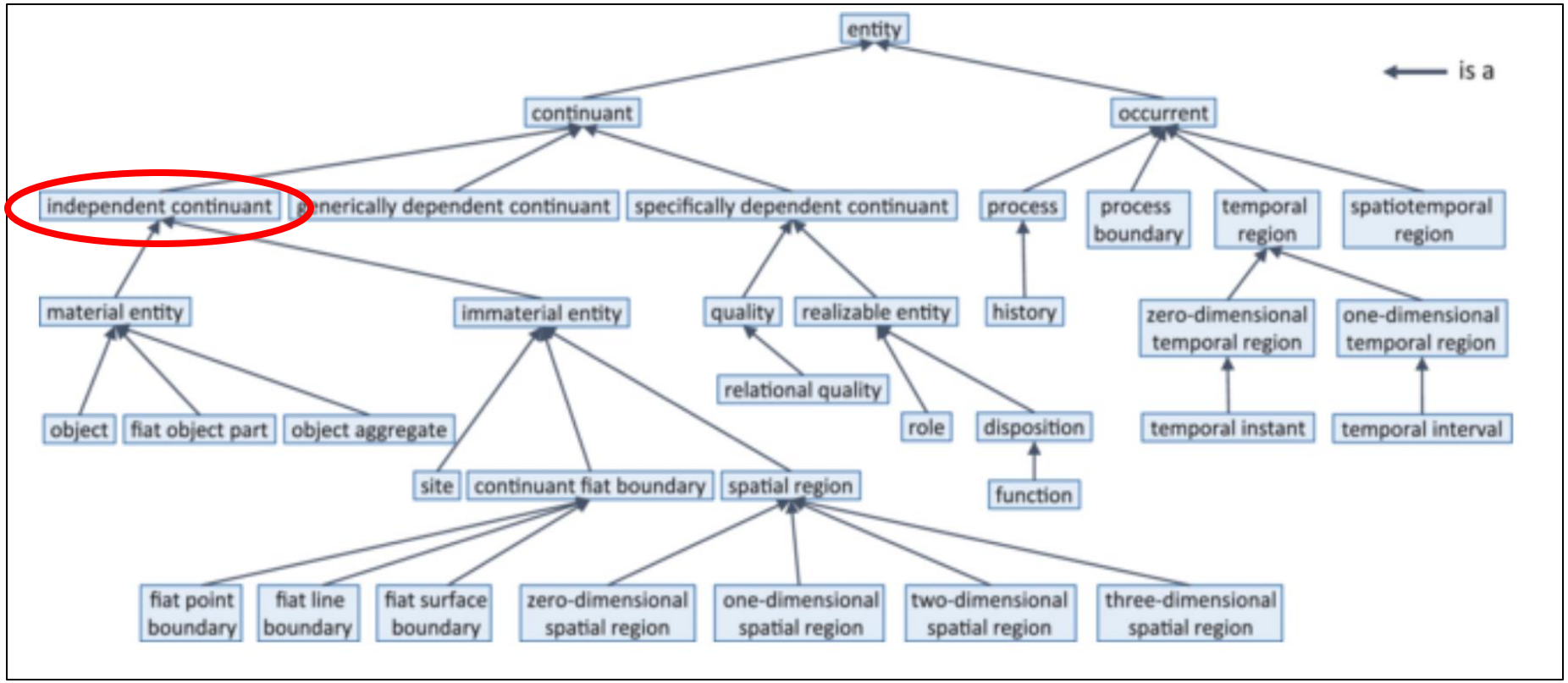
Dependency Chain

- A given **process** may **have realization** some **realizable entity**, which **inheres in** some independent continuant

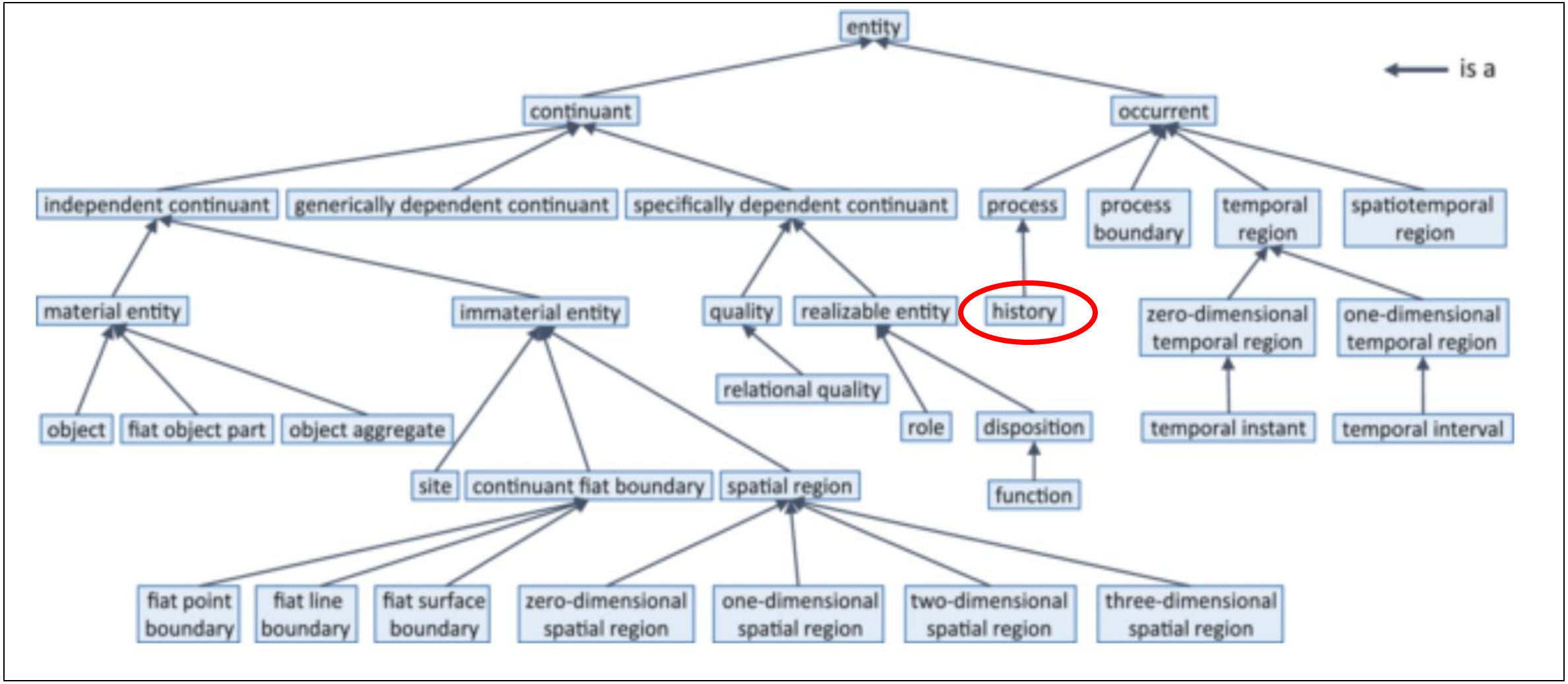


Dependency Chain

- A given **process** may **have realization** some **realizable entity**, which **inheres in** some **independent continuant**



History


entity
is a
continuant
occurrent
independent continuant
generically dependent continuant
specifically dependent continuant
process
process boundary
temporal region
spatiotemporal region
material entity
immaterial entity
quality
realizable entity
history
zero-dimensional temporal region
one-dimensional temporal region
relational quality
object
fiat object part
object aggregate
role
disposition
temporal instant
temporal interval
site
continuant fiat boundary
spatial region
function
fiat point boundary
fiat line boundary
fiat surface boundary
zero-dimensional spatial region
one-dimensional spatial region
two-dimensional spatial region
three-dimensional spatial region

History

- Is the sum total of all processes associated with a given material entity

- Every instance of history corresponds to one and only one instance of material entity; any instance of material entity corresponds to one and only one instance of history

- For example, the history that is my life is my history and mine alone, just as the history of the material entity that is this building belongs to the building

Change over Time

- In BFO, instances of material entities:

Change over Time

- In BFO, instances of material entities:
  - Have matter as parts

Change over Time

- In BFO, instances of material entities:
  - Have matter as parts
  - Gain or lose qualities,

Change over Time

- In BFO, instances of material entities:
  - Have matter as parts
  - Gain or lose qualities, parts,

Change over Time

- In BFO, instances of material entities:
  - Have matter as parts
  - Gain or lose qualities, parts, occupy different locations, etc.

Change over Time

- In BFO, instances of material entities:
  - Have matter as parts
  - Gain or lose qualities, parts, occupy different locations, etc. over the course of their history

Change over Time

- In BFO, instances of material entities:
  - Have matter as parts
  - Gain or lose qualities, parts, occupy different locations, etc. over the course of their history

- An apple in an orchard ripens, reddens, and sweetens, before spoiling, developing blotches, etc. on a fruit basket

Processes Do Not Change

- An intuitive understanding of change is the gain or loss of specifically dependent continuants

- In BFO, occurrents do not bear specifically dependent continuants, and so cannot – strictly speaking – gain or lose them

Processes are Changes

- As a consequence, characterizing:
  - increasing velocity of this vehicle
  - changing direction of this airplane
  - lowered volume of this alarm

- Are not understood in terms of properties of processes

- In BFO, processes do not change, they *are* changes

Outline

- Resource Description Framework (RDF)
- RDF Schema (RDFs)
- Modeling with Basic Formal Ontology
- Exercises

Task 1

In aircraft_data.xlsx you will find a row for the Airbus A320 Neo.

Construct a BFO-conformant design pattern reflecting the content of every column associated with that row.

Task 2

In aircraft_data.xlsx you will find a row for the Airbus A321-111, designed to have a maximum knot approach speed of 142. However, after 5 approaches, an instance has obtained an average maximum knot approach speed of 139.

Construct a BFO-conformant design pattern reflecting the preceding phenomena.

Task 3

In soc_structure_definitions.xlsx you will find three "SOC_TITLE" entries that mention "Aerospace Engineer".

Construct a BFO-conformant design pattern that reflects all three entries and their respective "SOC Definitions".

Task 4

In employment_wage_May_2024.xlsx you will find three "OCC_TITLE" entries that mention "Aerospace Engineer".

Construct a BFO-conformant design pattern that reflects all three entries and their associated column information.

Appendix

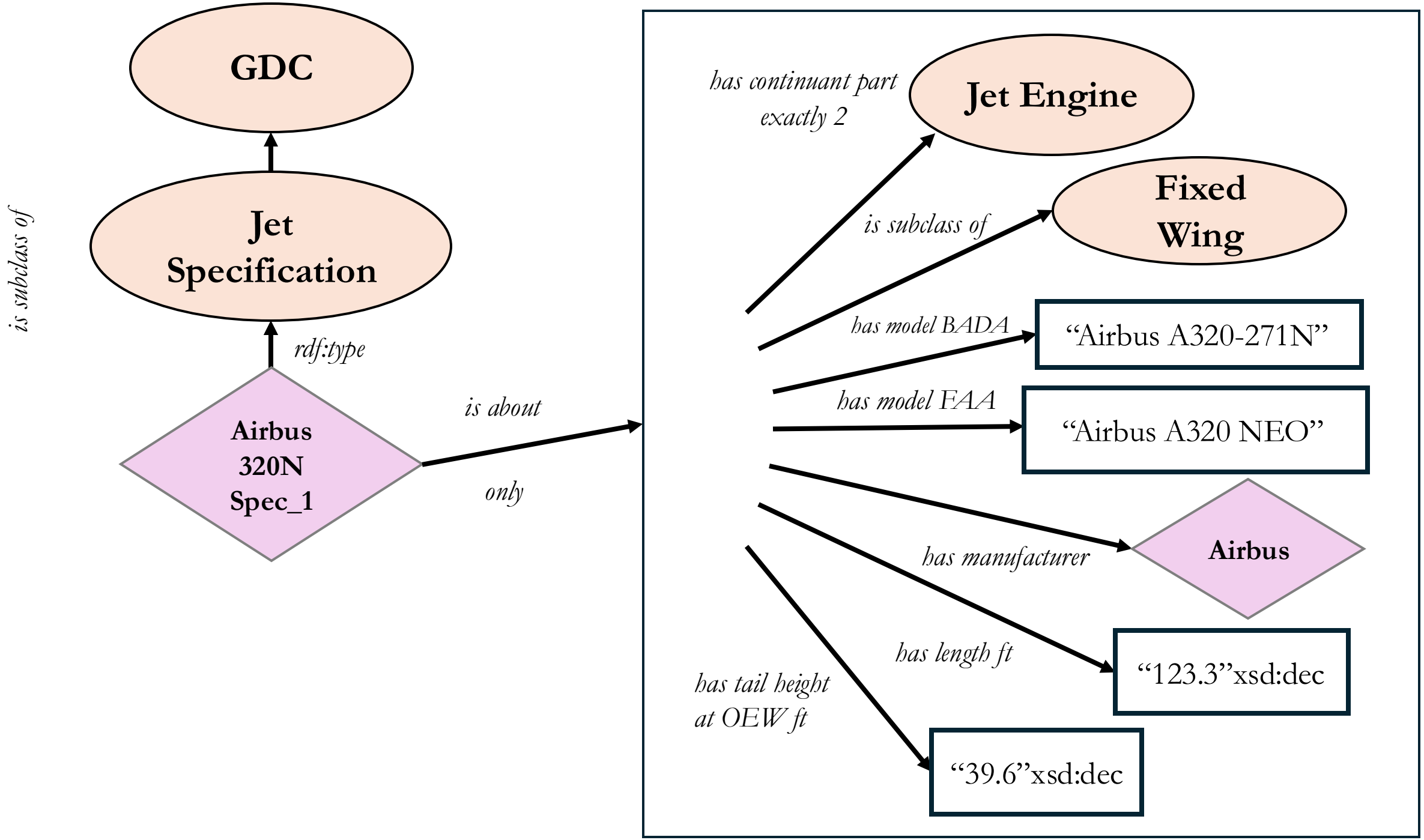
- Design Pattern Example for Task 1

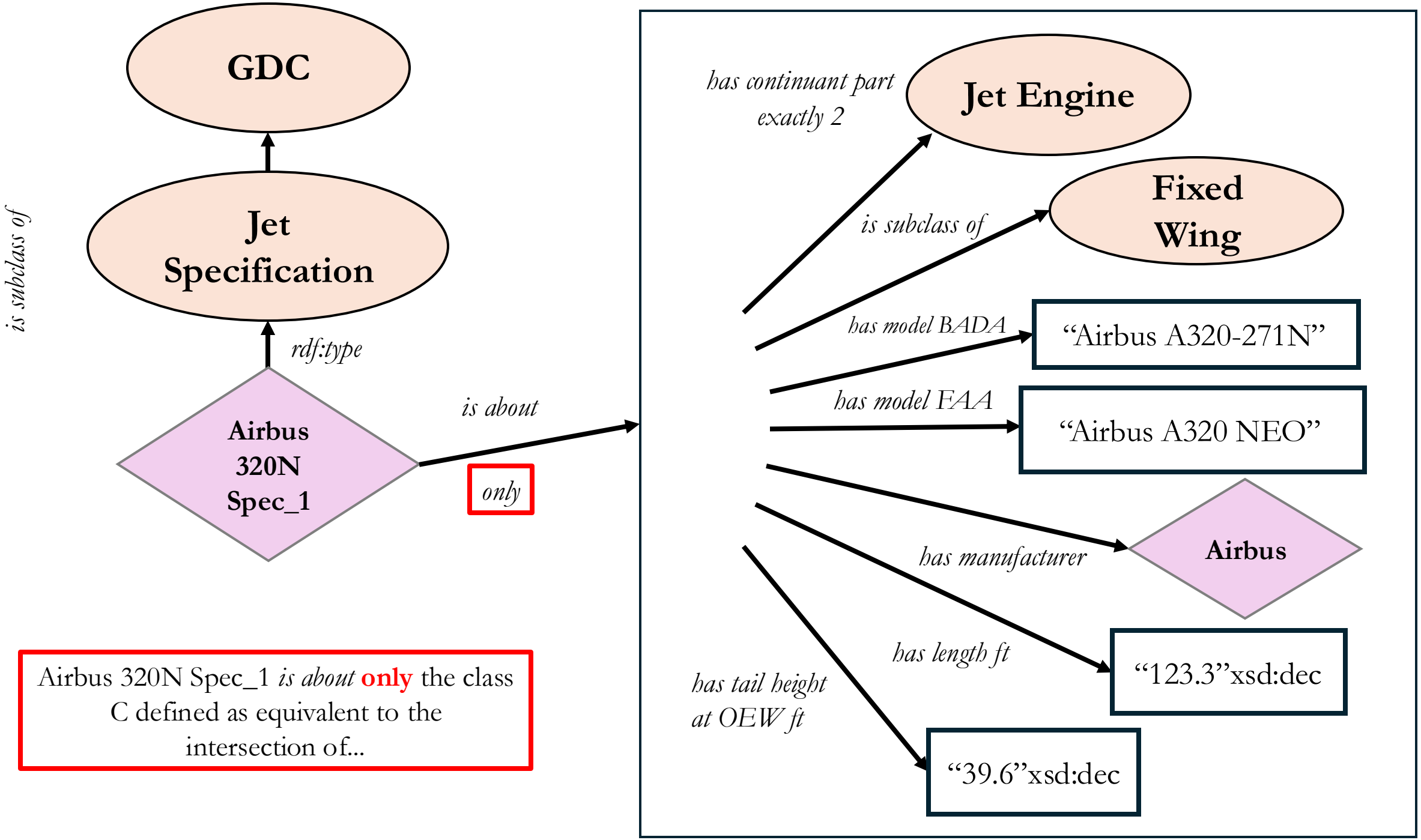
- General Advice

Appendix

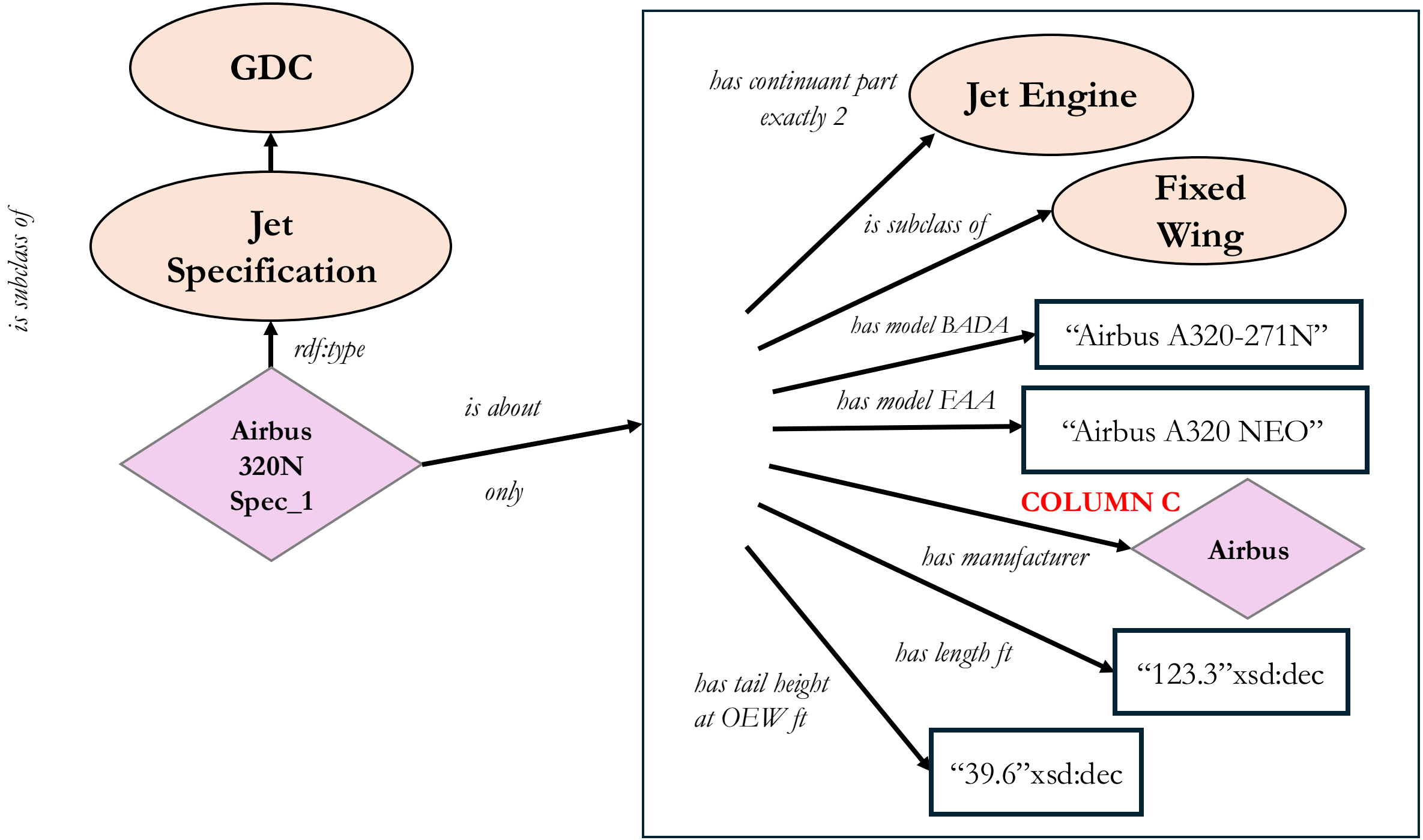
- Design Pattern Example for Task 1

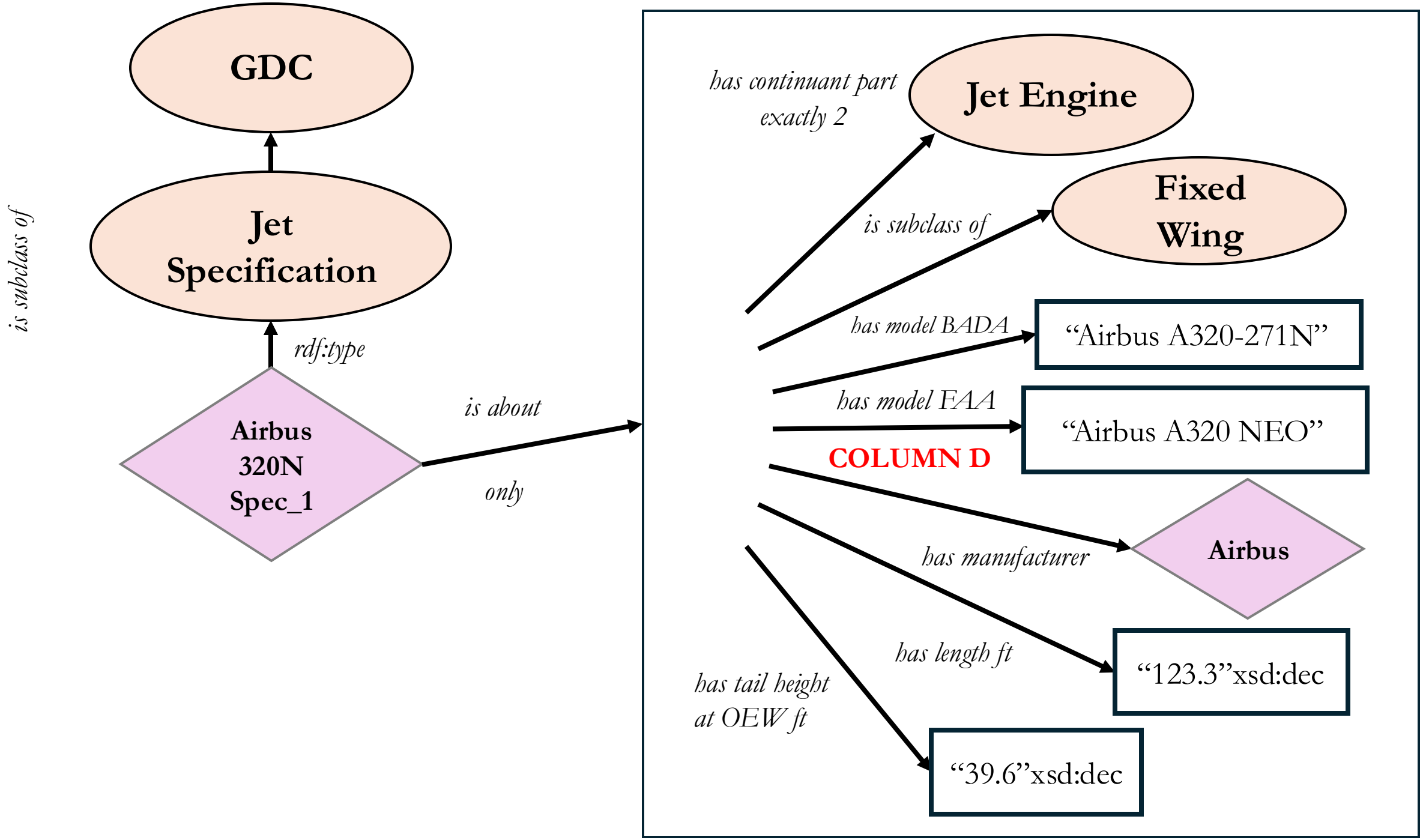
- General Advice

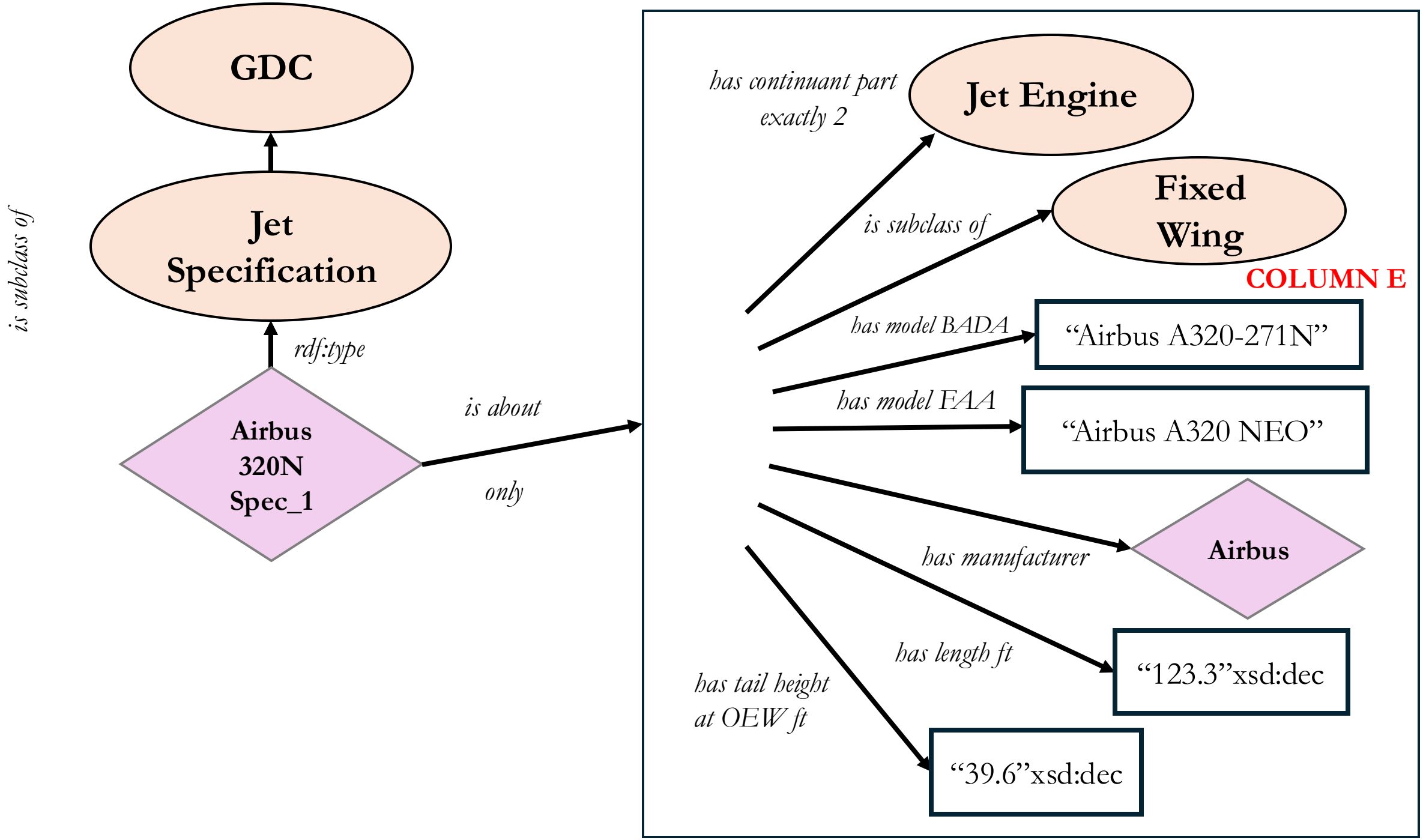

GDC
is subclass of
Jet Specification
rdf:type
Airbus 320N Spec_1
is about
only
has continuant part exactly 2
Jet Engine
is subclass of
Fixed Wing
has model BADA
"Airbus A320-271N"
has model FAA
"Airbus A320 NEO"
has manufacturer
Airbus
has length ft
"123.3"xsd:dec
has tail height at OEW ft
"39.6"xsd:dec

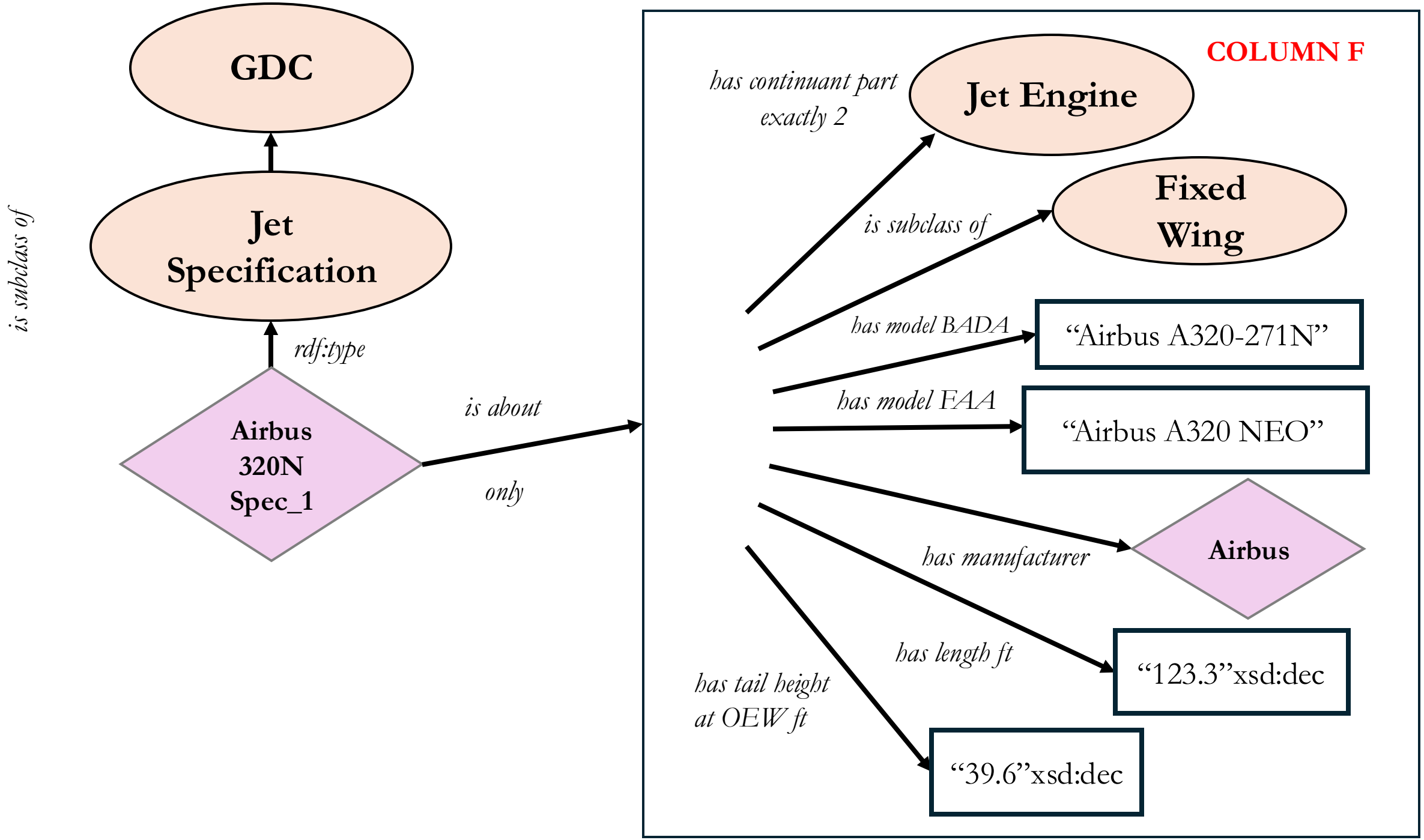


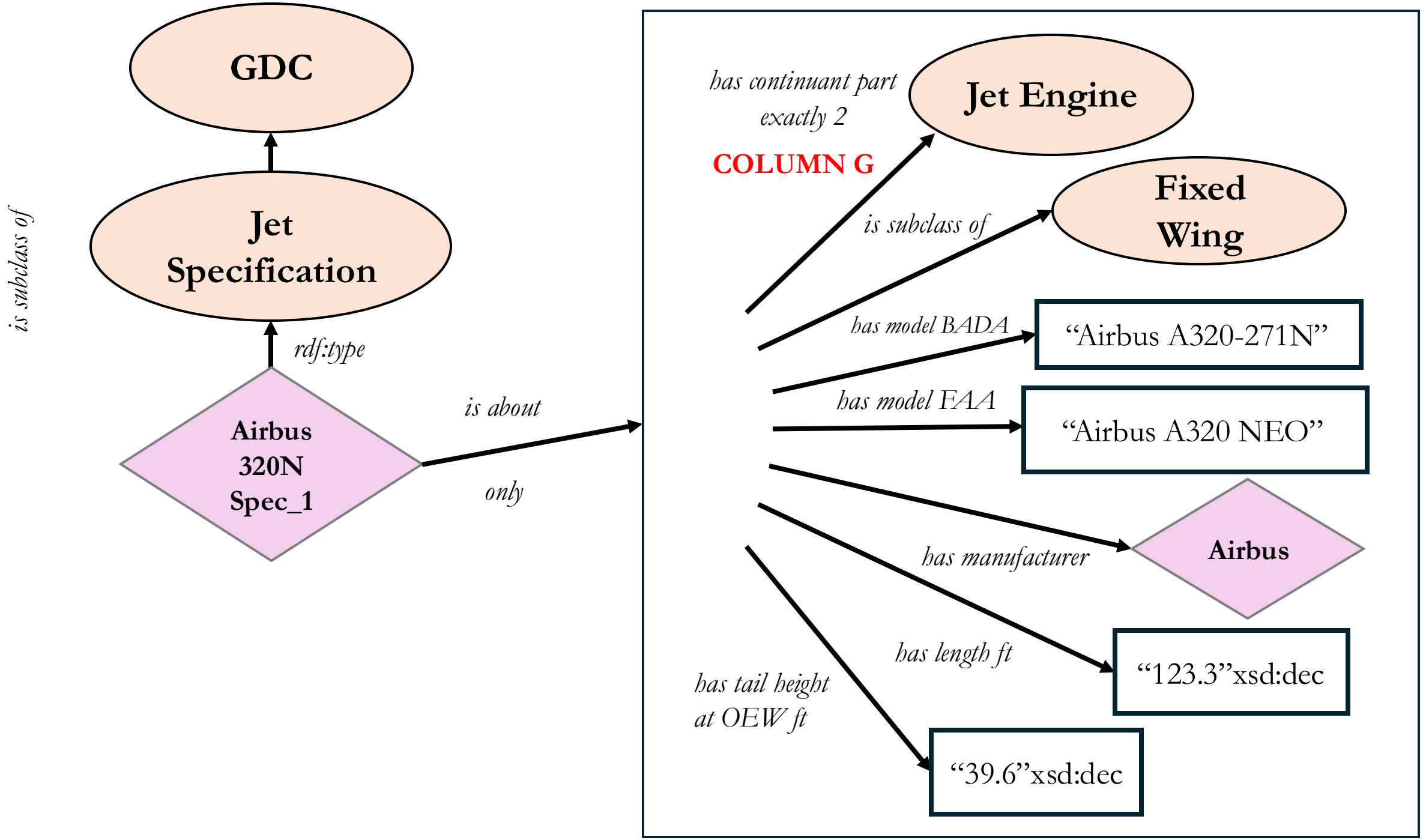
Airbus 320N Spec_1 *is about* **only** the class C defined as equivalent to the intersection of...

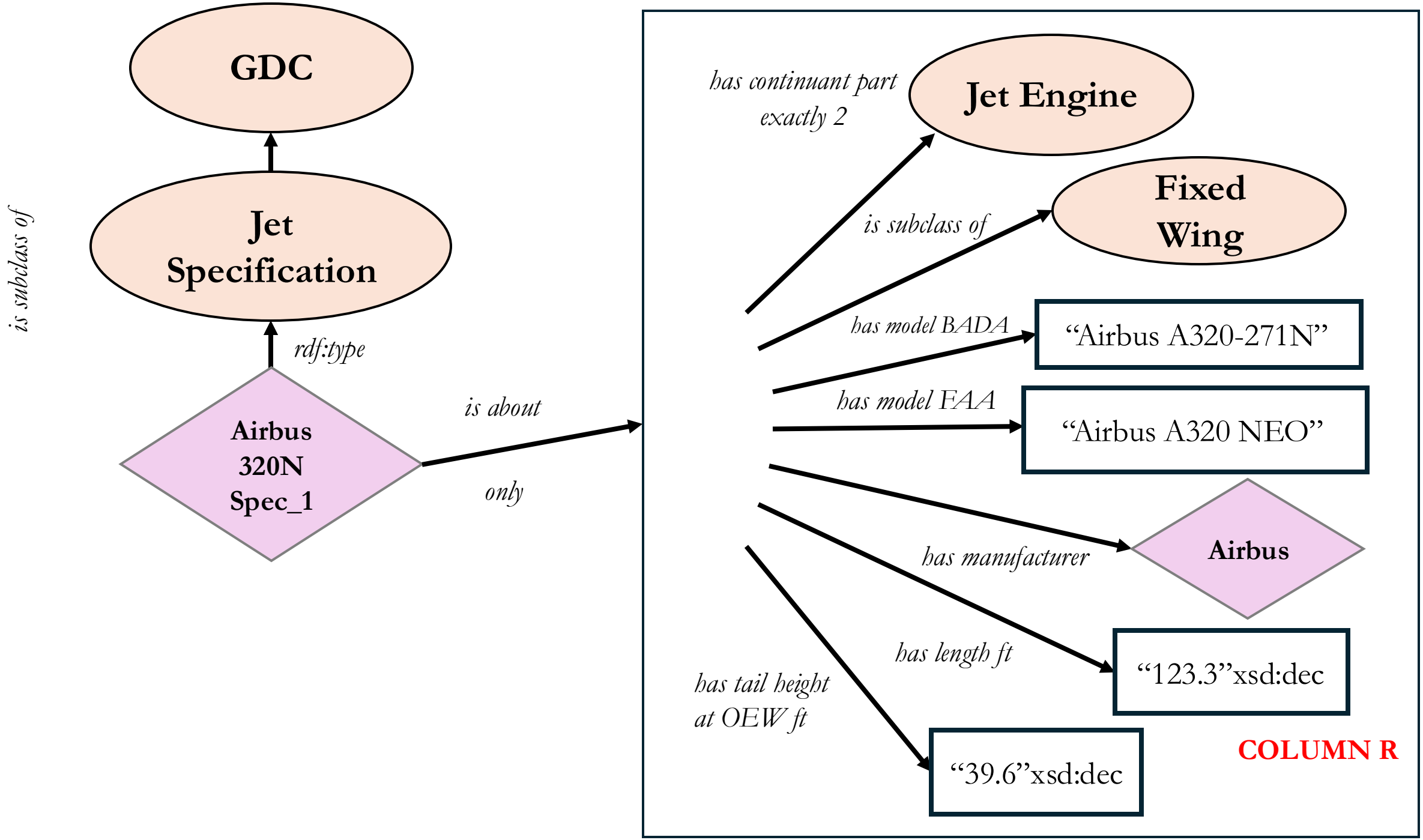

GDC
Jet Specification
is subclass of
rdf:type
Airbus 320N Spec_1
is about
only
has continuant part exactly 2
Jet Engine
is subclass of
Fixed Wing
has model BADA
"Airbus A320-271N"
has model FAA
"Airbus A320 NEO"
COLUMN C
has manufacturer
Airbus
has length ft
"123.3"xsd:dec
has tail height at OEW ft
"39.6"xsd:dec

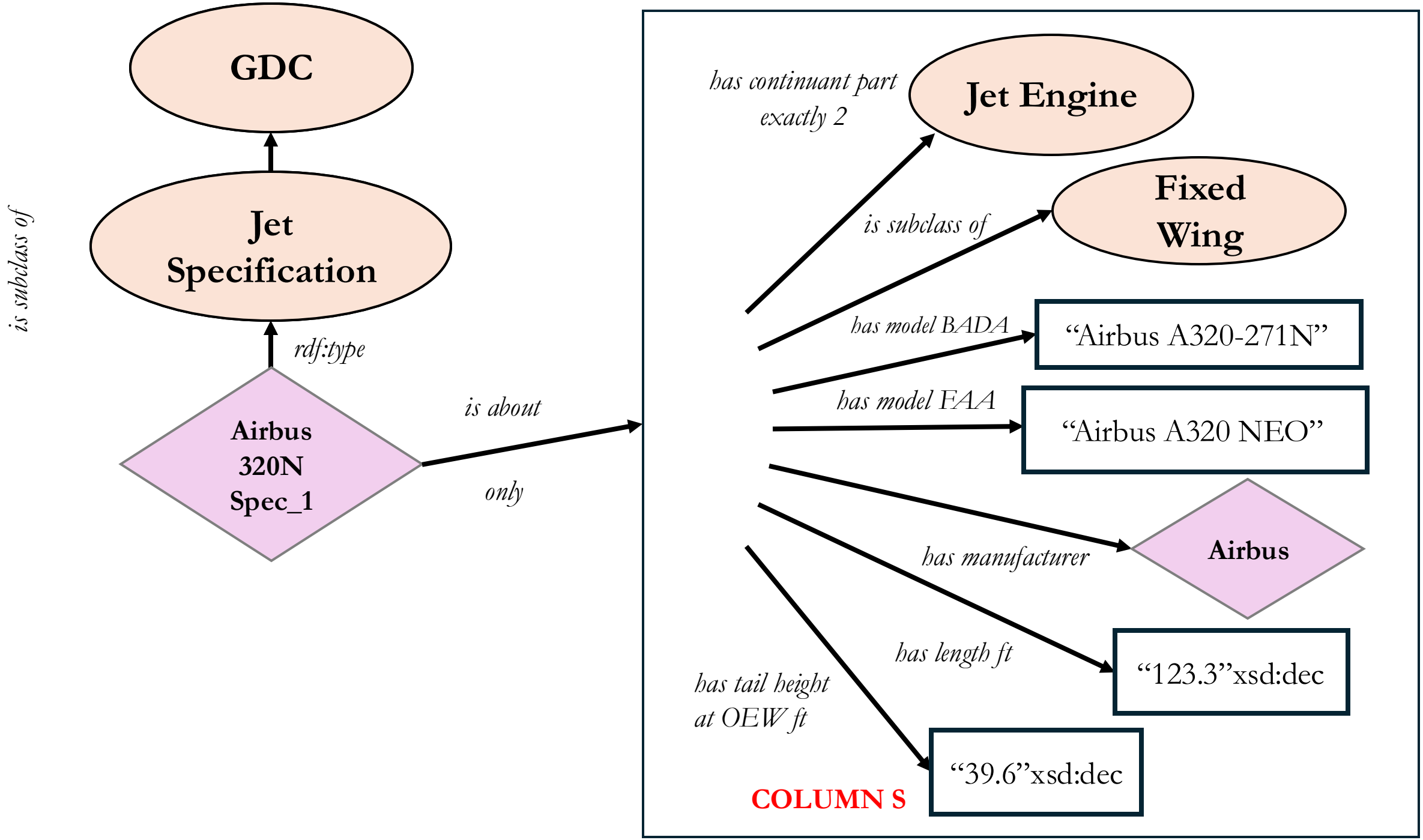

GDC
is subclass of
Jet Specification
rdf:type
Airbus 320N Spec_1
is about
only
has continuant part exactly 2
Jet Engine
is subclass of
Fixed Wing
has model BADA
"Airbus A320-271N"
has model FAA
"Airbus A320 NEO"
COLUMN D
has manufacturer
Airbus
has length ft
"123.3"xsd:dec
has tail height at OEW ft
"39.6"xsd:dec

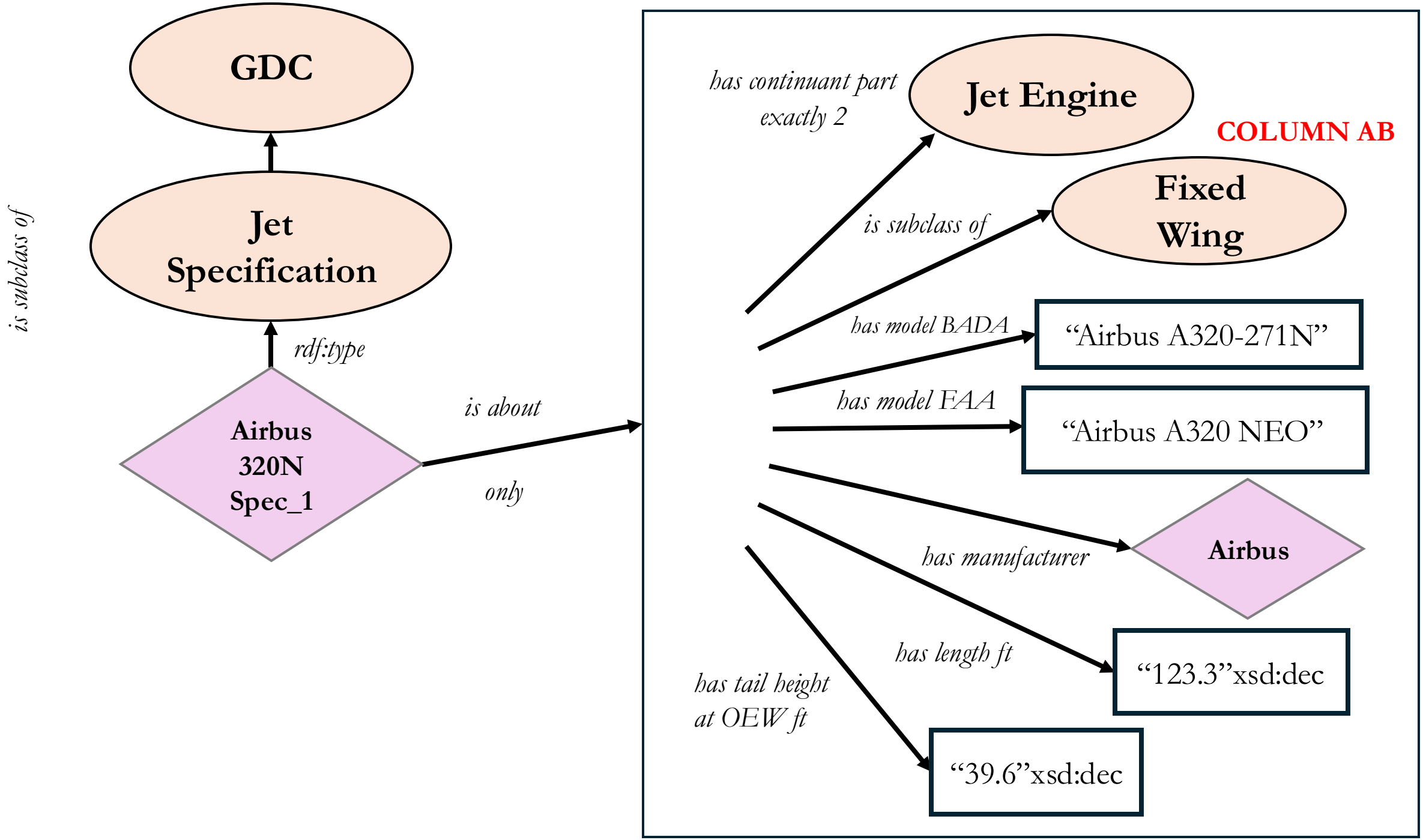

GDC
is subclass of
Jet Specification
rdf:type
Airbus 320N Spec_1
is about
only
has continuant part exactly 2
Jet Engine
is subclass of
Fixed Wing
COLUMN E
has model BADA
"Airbus A320-271N"
has model FAA
"Airbus A320 NEO"
has manufacturer
Airbus
has length ft
"123.3"xsd:dec
has tail height at OEW ft
"39.6"xsd:dec


GDC
Jet Specification
is subclass of
rdf:type
Airbus 320N Spec_1
is about
only
COLUMN F
has continuant part exactly 2
Jet Engine
is subclass of
Fixed Wing
has model BADA
"Airbus A320-271N"
has model FAA
"Airbus A320 NEO"
has manufacturer
Airbus
has length ft
"123.3"xsd:dec
has tail height at OEW ft
"39.6"xsd:dec


GDC
is subclass of
Jet Specification
rdf:type
Airbus 320N Spec_1
is about
only
has continuant part exactly 2
COLUMN G
Jet Engine
is subclass of
Fixed Wing
has model BADA
"Airbus A320-271N"
has model FAA
"Airbus A320 NEO"
has manufacturer
Airbus
has length ft
"123.3"xsd:dec
has tail height at OEW ft
"39.6"xsd:dec


GDC
is subclass of
Jet Specification
rdf:type
Airbus 320N Spec_1
is about
only
has continuant part exactly 2
Jet Engine
is subclass of
Fixed Wing
has model BADA
"Airbus A320-271N"
has model FAA
"Airbus A320 NEO"
has manufacturer
Airbus
has length ft
"123.3"xsd:dec
has tail height at OEW ft
"39.6"xsd:dec
COLUMN R


GDC
is subclass of
Jet Specification
rdf:type
Airbus 320N Spec_1
is about
only
has continuant part exactly 2
Jet Engine
is subclass of
Fixed Wing
has model BADA
"Airbus A320-271N"
has model FAA
"Airbus A320 NEO"
has manufacturer
Airbus
has length ft
"123.3"xsd:dec
has tail height at OEW ft
"39.6"xsd:dec
COLUMN S


GDC
is subclass of
Jet Specification
rdf:type
Airbus 320N Spec_1
is about
only
COLUMN AB
has continuant part exactly 2
Jet Engine
is subclass of
Fixed Wing
has model BADA
"Airbus A320-271N"
has model FAA
"Airbus A320 NEO"
has manufacturer
Airbus
has length ft
"123.3"xsd:dec
has tail height at OEW ft
"39.6"xsd:dec

Classes | Object properties

Individuals: Airbus 320N Spec_1

- Airbus
- Airbus 320N Spec_1

Annotations: Airbus 320N Spec_1

Annotations

rdfs:label [language: en]

Airbus 320N Spec_1

dc:creator

http://orcid.org/0000-0002-1118-1738

Description | Manchester syntax rendering | Turtle rendering

Description: Airbus 320N Spec_1

Types

'is about' only ('fixed wing aircraft' and ('has continuant part' exactly 2) and ('has model FAA' value "Airbus A320 NEO") and ('has model BADA' value "Airbus A320-271N") and ('has length ft' value "123.3") and ('has tail height at OEW ft' value "39.6") and ('has manufacturer' value Airbus))

Appendix

- Design Pattern Example for Task 1

- General Advice

Research

- You will undoubtably find these tasks challenging; you will get stuck, frustrated, be unsure what to do, and so on
- This is by design
- I don’t give students fish, I teach them how to fish
- Trust the process

Research

- Take advantage of the following:
  - You are **encouraged** that you do your own research; you should not assume I've covered everything you need to know to complete the tasks
  - There are **several students** in the program that have worked through similar seminars with me, you may ask them for help, you may search GitHub for their previous submissions all of which are still online
  - Leverage LLMs, search the web, etc. You are welcome to **get as much help** as you need; that is how most solve challenges in the real world
  - It is **OK to be wrong**; I'd rather you fail in this safe environment than outside the classroom where it is perhaps less safe
  - Exercises in this seminar are in many ways unfair, **my grading** however, is fair