# LOD – The logic of Descriptions

## Theories

## LOD - The logic of Descriptions

- TBoxes and terminologies
- LOD theories
- Unfolding
- Reasoning

# LOD - TBox (definition)

**Definition (TBox)** A TBox is a finite set of **concept inclusions**, i.e. a finite set of expressions of the type

$$C \sqsubseteq D$$

where *C* is a concept (an etype) and *D* is an atomic formula.

**Observation 1:** “T” in “”TBox stands for “Term”

**Observation 2:** TBox is a theory, according to our earlier terminology

**Observation 3:** A LOD theory is a set of constraints on the domain structure

**Observation 4:** In general *C* is allowed to be an atomic formula. Our constraint is motivated by our interest in modeling language (definitions and descriptions).

## LOD – acyclic TBox (definition)

**Definition (Acyclic TBox).** A **TBox** is **acyclic** if it satisfies the following properties:

1. any concept can appear at most once on the left side of a definition
2. it is acyclic (there are no definition cycles)

**Observation 1:** the simplest case of cycle is the subsumption

$$C \sqsubseteq C$$

**Observation 2:** Acyclicity is crucial in the definition of concepts and in the description of their properties (as etypes).

## LOD - Terminology (definition)

**Definition (Definitional TBox, Terminology )** A **TBox** is **definitional** (it is a **terminology**) if it satisfies the following properties:

1. It is acyclic
2. It contains only concept equivalences

**Observation**: A concept inclusion $A \sqsubseteq C$ can always be transformed in a definition $A \equiv C \sqcap A_C$ with a suitable $A_C$

## LOD Terminology (example)

*Family relations*

- Person ≡ ∃hasname.String ⊓ ∀HasJob.Organization
- Woman ≡ Person ⊓ Female
- Man ≡ Person ⊓ ¬ Woman
- Mother ≡ Woman ⊓ ∃hasChild.Person
- Father ≡ Man ⊓ ∃hasChild.Person
- Parent ≡ Father ⊔ Mother

## LOD - The logic of Descriptions

- TBoxes and terminologies
- LOD theories
- Unfolding
- Reasoning

## LOD theories

- Lexicons
- Lexical teleontologies
- Knowledge teleontologies
- Teleologies
- Example

# Natural language Lexicon

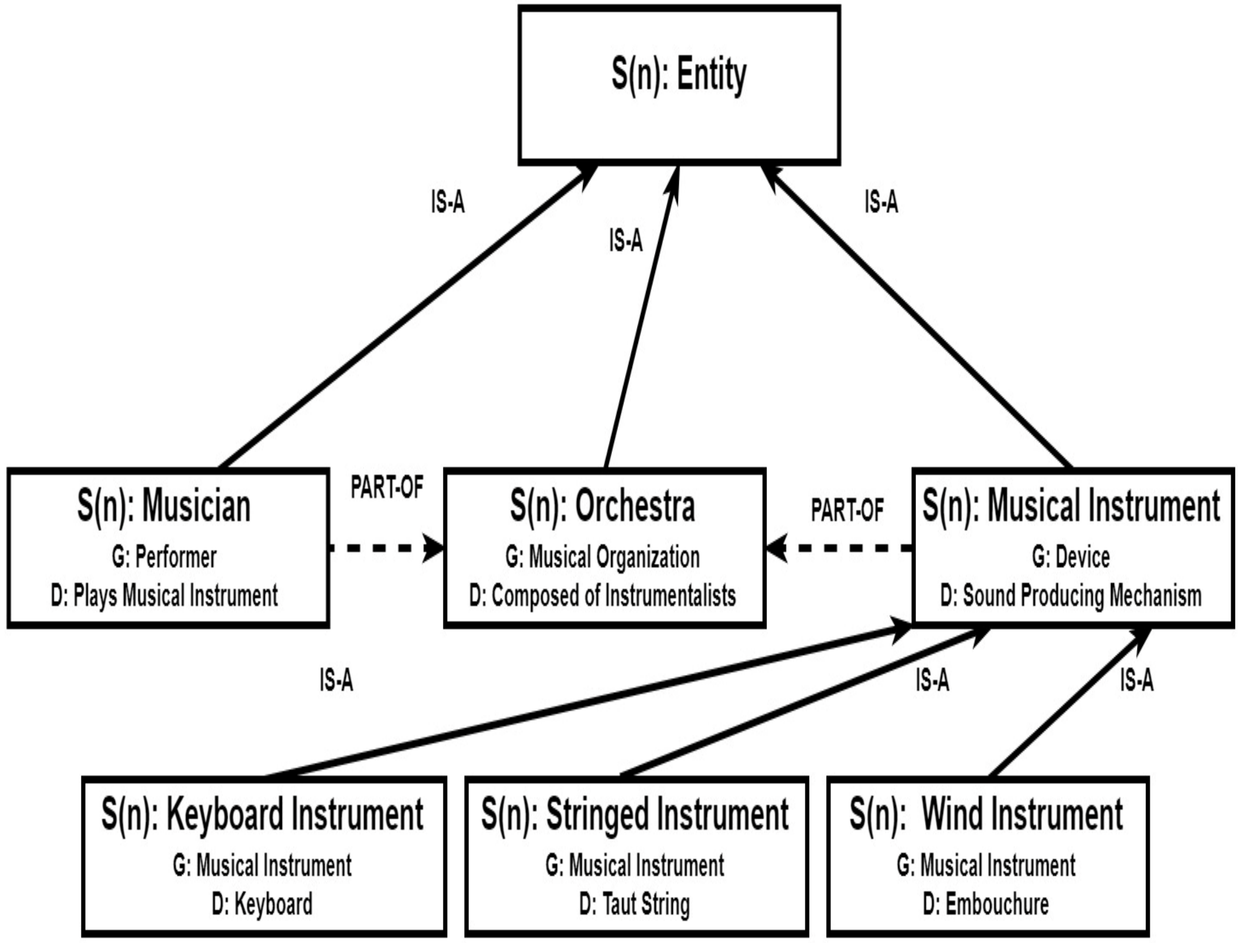
Lexicons are *informal hierarchies* which encode natural language(s) via Genus-Differentia: an ISA hierarchy of synsets.

Sense? One of the many concepts denoted by a polysemous word (e.g., *car* stands for *automobile* and *railway car*).

Synsets? sets of synonyms, e.g., words having same or similar meaning for the same sense (e.g., *car*, *automobile*)

Genus? set of properties which define the scope of a sense, e.g., *musical instrument (G)* for *stringed instrument.*

Differentia? Set of properties which qualify / differentiate senses with the same genus, e.g., *Taut String (D)* for *Stringed Instrument.*



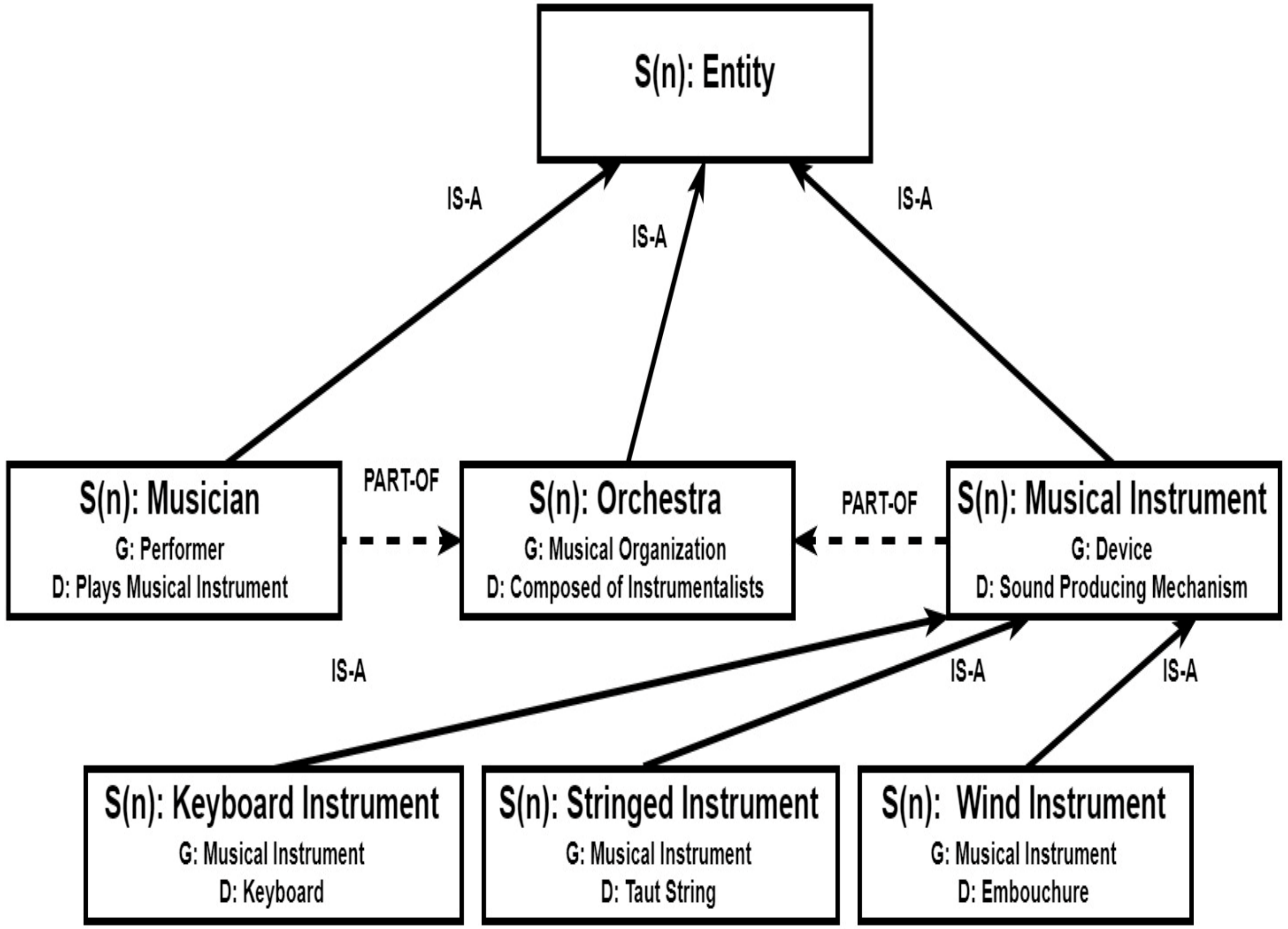
# Natural language Lexicon (continued)

The IS-A hierarchy semantically models superclass - subclass relations between senses based on genus-differentia

Each sense is identified via a unique identifier named *GID*, e.g., *588967* for *Musician* (not visible to user)

Each word is (implicitly) defined via a universal quantification over its differentia, e.g., Stringed Instrument’s differentia with respect to Musical instrument is ∀*D.TautStrings*.

Lexicons are built by adhering to quality principles for modelling, e.g., how to differentiate a node into children nodes, etc.



# Natural language Lexicon (continued)

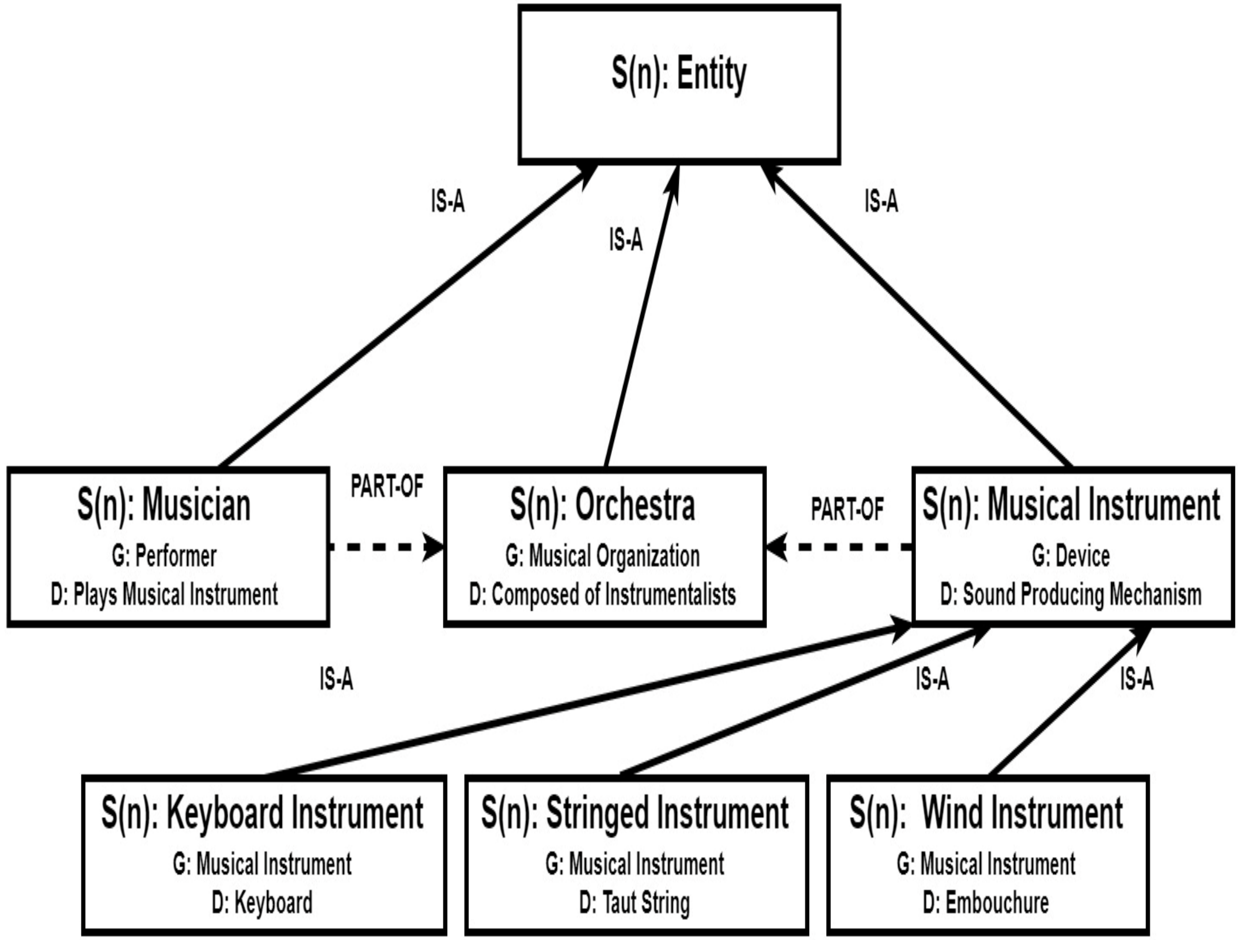
In addition to the IS-A hierarchy, lexicons are also organized according to a PART-OF hierarchy.

All concepts have *parts*. For any part there is a "bigger" *whole* which somehow "contains" it. For instance Musicians and Musical Instruments are part-of Orchestras. Musicians have parts, Musicians have parts, …, and so on, down to materials.

Part-of links model the part-whole relation which exists between a whole (the Unity) and possibly multiple diverse parts.

The whole defines the spatial context within whose boundaries the EG is built.

The PART-OF hierarchy defines the relevant component parts of the whole, namely those which will ultimately be considered in an ETG/EG (as, e.g. selected in ER/EER models)



# Natural language Lexicon (continued)

The IS-A and PART-OF hierarchies are independent orthogonal hierarchies

The PART-OF hierarchy models containment. Space containment with objects, Time containment with events.

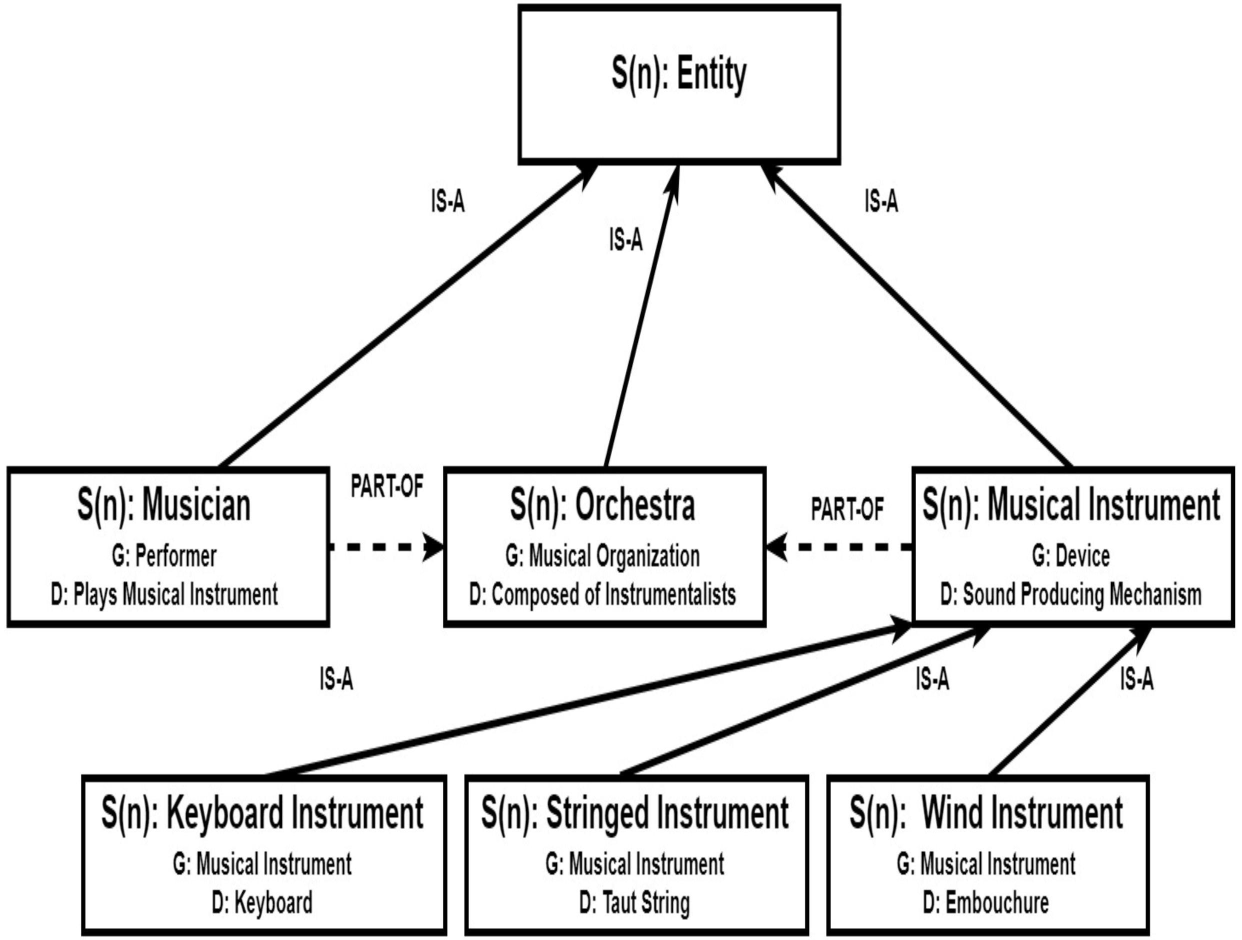
The IS-A hierarchy models the behavior of entities, that is how objects specialize in their properties (i.e., their functions and actions).

*Entity* (=*anything*), the top concept of the IS-A has no properties and no parts. But it is PART-OF everything.

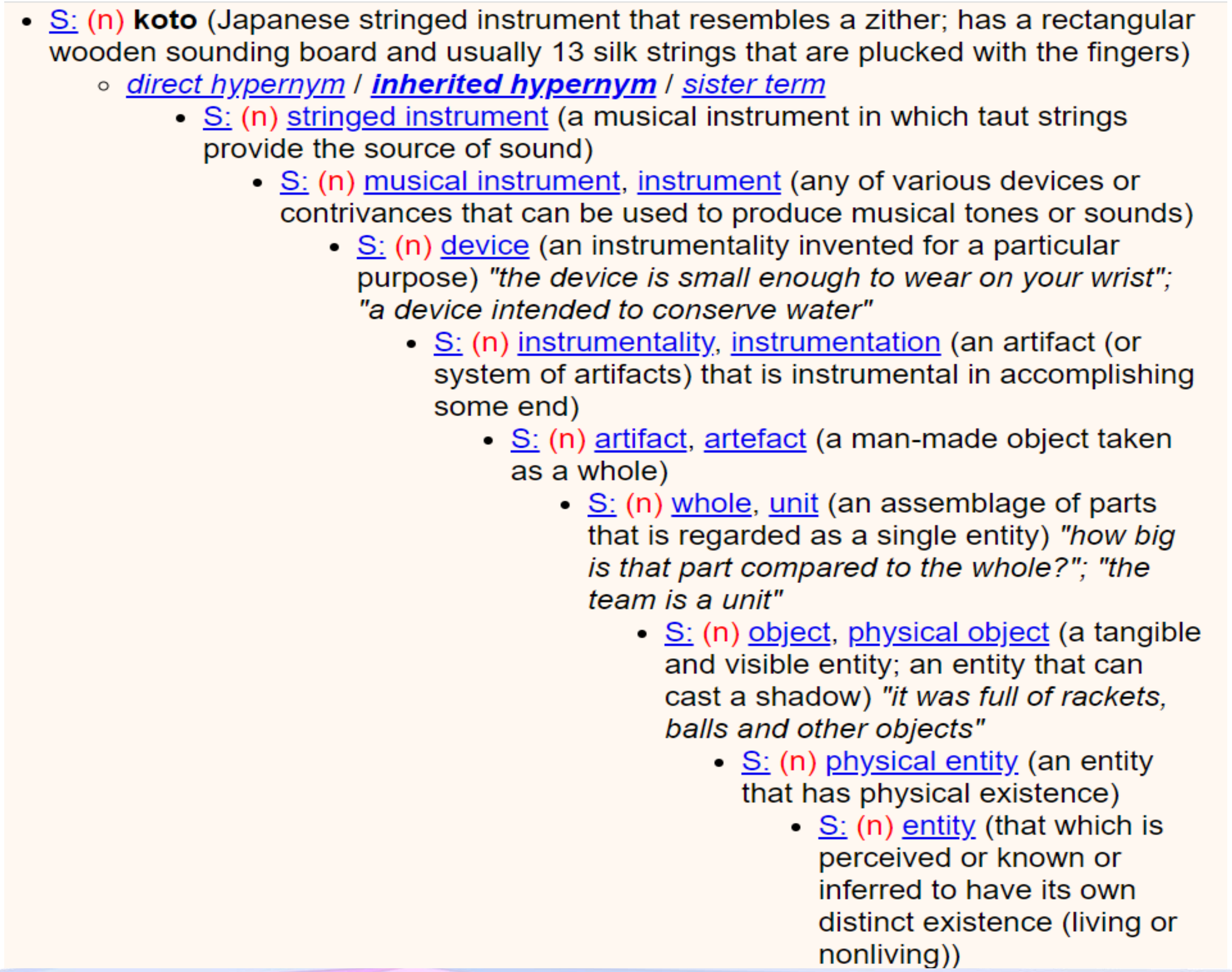
*Everything*, the top concept of the PART-OF hierarchy contains all parts and therefore has all properties.

**If** *PART-OF(part, whole)* **then**
*Property(part, P) ≡ Part-Property (whole,P)*

The IS-A and PART-OF hierarchies form a *lattice.*



# Natural language Lexicon – example (WordNet - Koto)



- S: (n) **koto** (Japanese stringed instrument that resembles a zither; has a rectangular wooden sounding board and usually 13 silk strings that are plucked with the fingers)
  - *direct hypernym* / ***inherited hypernym*** / *sister term*
    - S: (n) stringed instrument (a musical instrument in which taut strings provide the source of sound)
      - S: (n) musical instrument, instrument (any of various devices or contrivances that can be used to produce musical tones or sounds)
        - S: (n) device (an instrumentality invented for a particular purpose) *"the device is small enough to wear on your wrist"; "a device intended to conserve water"*
          - S: (n) instrumentality, instrumentation (an artifact (or system of artifacts) that is instrumental in accomplishing some end)
            - S: (n) artifact, artefact (a man-made object taken as a whole)
              - S: (n) whole, unit (an assemblage of parts that is regarded as a single entity) *"how big is that part compared to the whole?"; "the team is a unit"*
                - S: (n) object, physical object (a tangible and visible entity; an entity that can cast a shadow) *"it was full of rackets, balls and other objects"*
                  - S: (n) physical entity (an entity that has physical existence)
                    - S: (n) entity (that which is perceived or known or inferred to have its own distinct existence (living or nonliving))

On the left hand side is the WordNet lexical hierarchy generalizing the concept for Koto. See Princeton WordNet

S(n) indicates a synset associated to a word (here Koto) (one of the possibly many) of synonymous nouns. Here Koto has no synonyms

Hyponym/ Hypernym indicate subclass and superclass IS-A relationship

Meronym indicates the part-of relation.

Each synset is described by a gloss (an definition, most of the time incomplete, provided informally) and an example (between quotes in the figure on the left).

# Domain Lexicon

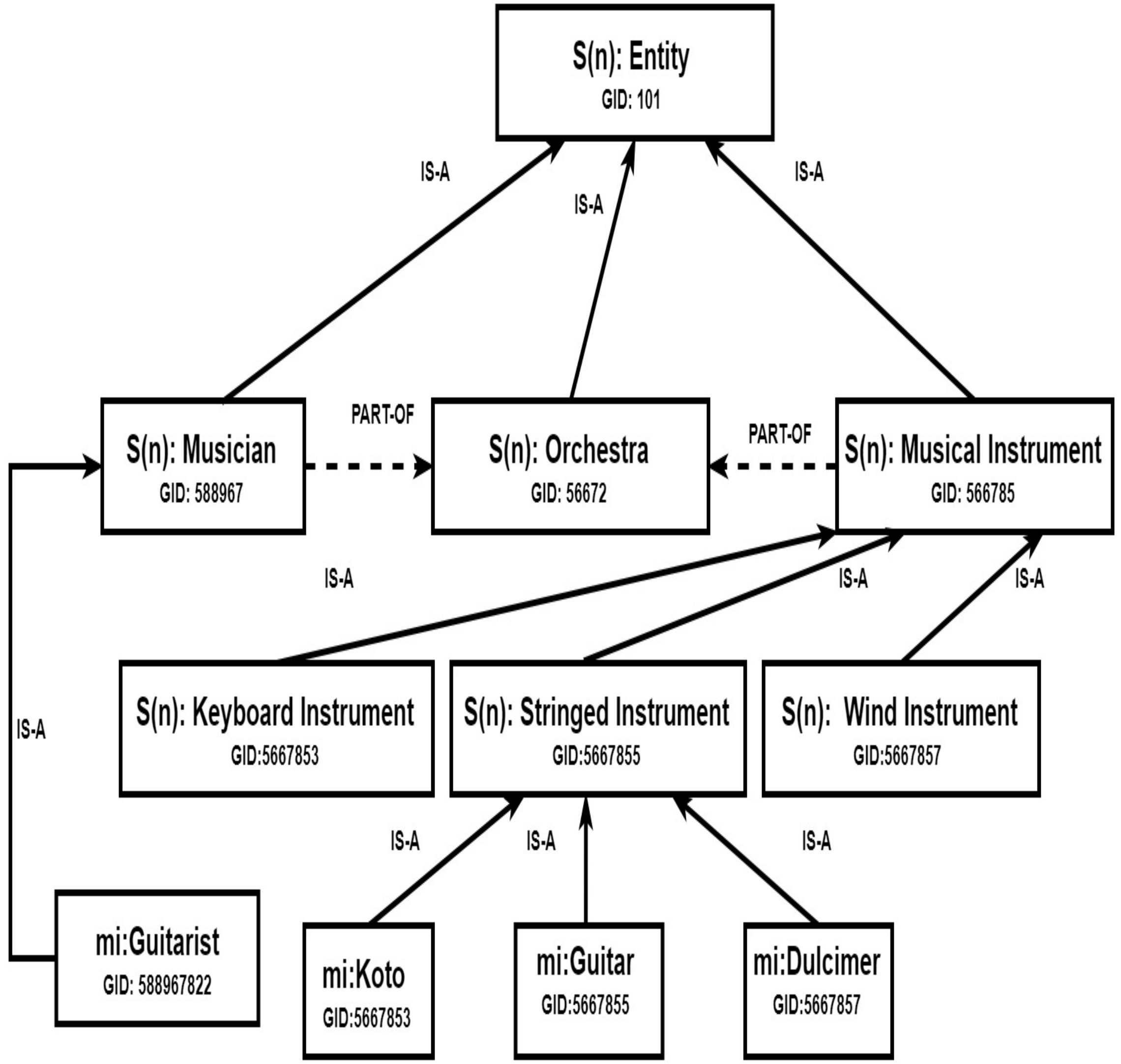
Domain Lexicons (XML name spaces) are informal hierarchies which encode domain language(s)

They extend natural language lexicons with domain-specific terminology. They are therefore language aware.

Words in domain languages follow the same rules as those in natural languages with one exception: they are NOT polysemous, i.e., they have only one sense.

Each word has a prefix, e.g., *mi*: (e.g., for musical instruments) to indicate the domain language/ name space to which the words is associated.

As with lexicons, each sense is identified via a unique identifier named *GID*, e.g., *5667853* for *mi:Koto,* denoting the single sense of that word.



## LOD – Lexicon formalization (IS-A Hierarchy)

$$\text{Label} \equiv \text{Genus} \sqcap \text{Differentia}$$

As from the example above on musical instruments:

- $\text{KeyboardInstr} \equiv \text{MusicalInstr} \sqcap$
  $\exists\text{meansOfSoundProduction.Keyboard} \sqcap \forall\text{meansOfSoundProduction.Keyboard} \dots$
- $\text{KeyboadInstr} \sqsubseteq \neg\, \text{StringedInstr}$
- $\text{KeyboadInstr} \sqsubseteq \neg\, \text{WindInstr}$

# LOD – Lexicon formalization (IS-A Hierarchy ) (cont)

**Observation 1**: only one etype: concept (etype *concept* and not dtype *string* as we need to define object properties)

**Observation 2:** The label appears only once in the left of a definition and on the right of all its children

**Observation 3**: In a definition only conjuncts (as many as needed), where each conjunct consists only of exists and forall quantification

**Observation 4**: The subsumption relation holds between a label and its genus. For instance

$$\text{KeyboardInstr} \sqsubseteq \text{MusicalInstr}$$

**Observation 5:** A formalized lexicon is a subsumption hierarchy

# LOD – Lexicon formalization (IS-A Hierarchy ) (cont)

**Observation 6:** For each definition as many pairwise disjointness constraints as there are siblings. For instance (the left hand side is a complex assertion)

$$\text{KeyboadInstr} \sqcap \text{StringedInstr} \sqsubseteq \bot$$

**Observation 7**: Disjointness constraints can be formalized in two ways (check yourself in Venn Diagram)

$$\text{KeyboadInstr} \sqsubseteq \neg\, \text{StringedInstr}$$

$$\text{StringedInstr} \sqsubseteq \neg\, \text{KeyboadInstr}$$

In this formulation, the left hand side is a concept and not a complex assertion

**Observation 8:** Negation applies <u>only</u> to assertions

**Observation 9:** The relation between Genus and differentia could also be stated as

$$\text{Genus} \equiv \text{Label1} \sqcap \text{Label2} \sqcap \ldots \sqcap \text{UnknownLabel}$$

with UnknownLabel optional and defined as

$$\text{UnknownLabel} \equiv \text{Genus} \sqcap \text{UnknownDifferentia}$$

**Observation 10:** The IS-A hierarchy of a formalized lexicon is a terminology

# LOD – Lexicon formalization (part-of hierarchy)

Part-of(part, whole)

with inverse relation

Whole-of(label2,label1)

**Observation 1**: A hierarchy but with no property inheritance

**Observation 2**: Whole defines the physical (space) boundaries within which the parts are located

**Observation 3**: Whole provides reference coordinate system, parts provide functionalities

**Observation 4:** Part-of hierarchy formalized mucs less frequently

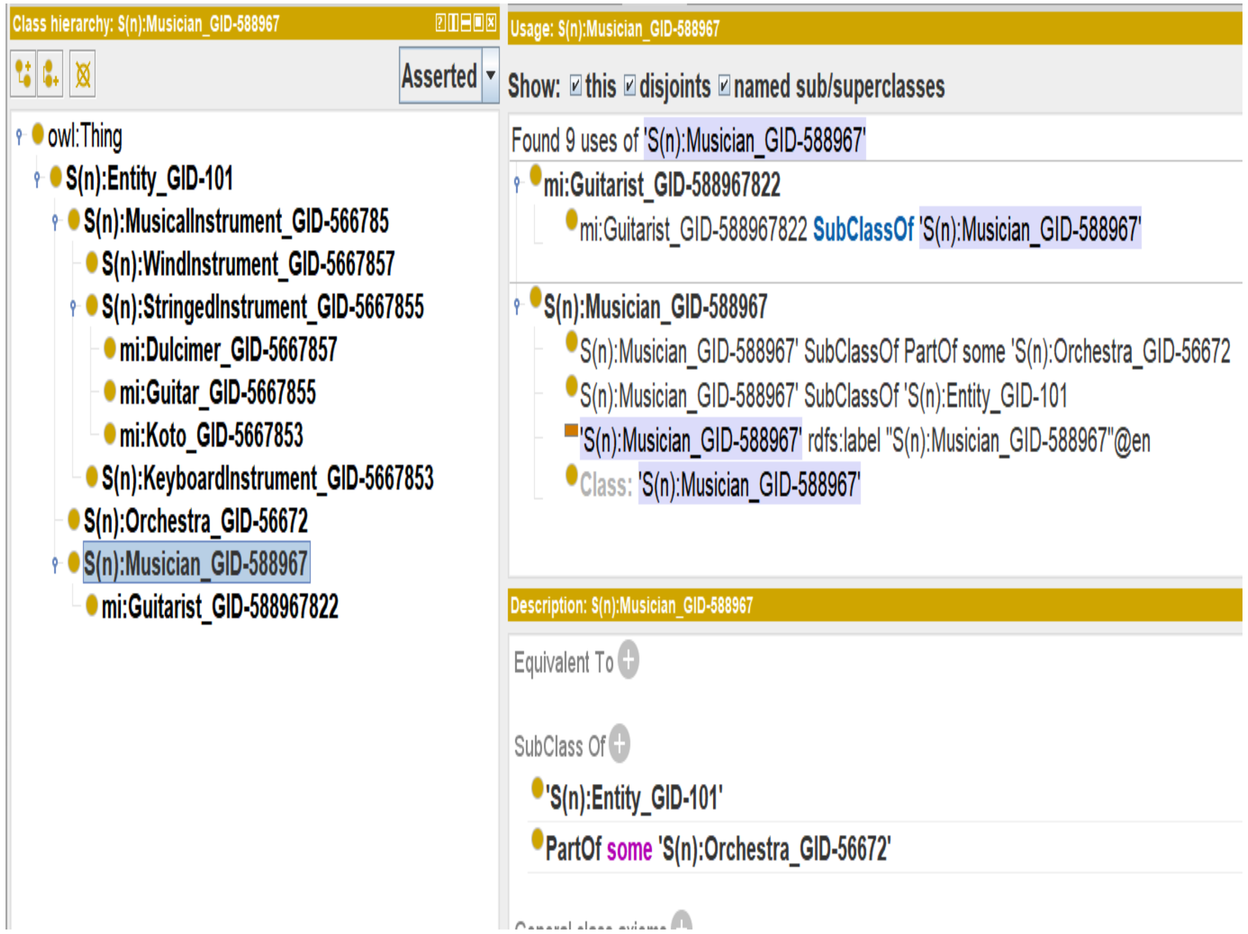
# LOD - Protégé (General Example)

Left: concept IS-A hierarchy

- owl:Thing (predefined root)

Right: concept property specification

- Class name (Class)
- IS-A hierarchy dependency (SubClassOf)
- Data property with its codomain (Domain)
- How and in which language a concept is named (rdfs:label)
- some ~ keyword for existential quantification.



Class hierarchy: S(n):Musician_GID-588967

Asserted

- owl:Thing
  - S(n):Entity_GID-101
    - S(n):MusicalInstrument_GID-566785
      - S(n):WindInstrument_GID-5667857
      - S(n):StringedInstrument_GID-5667855
        - mi:Dulcimer_GID-5667857
        - mi:Guitar_GID-5667855
        - mi:Koto_GID-5667853
      - S(n):KeyboardInstrument_GID-5667853
    - S(n):Orchestra_GID-56672
    - S(n):Musician_GID-588967
      - mi:Guitarist_GID-588967822

Usage: S(n):Musician_GID-588967

Show: ☑ this ☑ disjoints ☑ named sub/superclasses

Found 9 uses of 'S(n):Musician_GID-588967'

- mi:Guitarist_GID-588967822
  - mi:Guitarist_GID-588967822 SubClassOf 'S(n):Musician_GID-588967'
- S(n):Musician_GID-588967
  - S(n):Musician_GID-588967' SubClassOf PartOf some 'S(n):Orchestra_GID-56672
  - S(n):Musician_GID-588967' SubClassOf 'S(n):Entity_GID-101
  - 'S(n):Musician_GID-588967' rdfs:label "S(n):Musician_GID-588967"@en
  - Class: 'S(n):Musician_GID-588967'

Description: S(n):Musician_GID-588967

Equivalent To

SubClass Of

- 'S(n):Entity_GID-101'
- PartOf some 'S(n):Orchestra_GID-56672'

# LOD – Lexicon example (Protégé)

The snippet on the right side shows the domain lexicon formalized via the Protégé ontology editor.

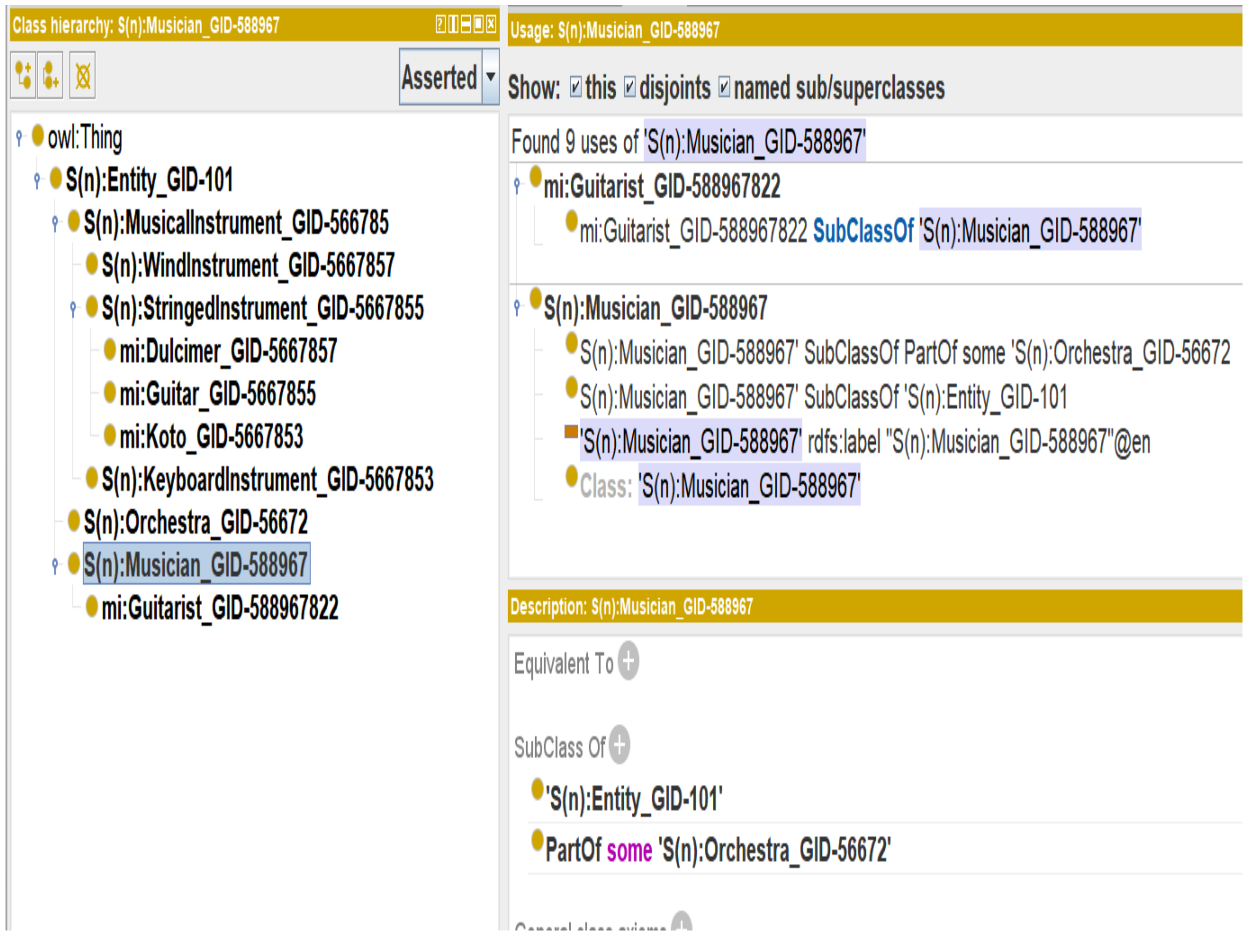
You can see the entire class hierarchy starting from Entity downwards depicting the concepts with their unique GIDs.

Notice mi:Koto, mi:Guitar etc, belong to domain lexicon and not natural language lexicon.

You can also see (partial) visualization of LOD formalization of the example domain lexicon, e.g.,

Musician is - <u>PartOf</u> - (some) Orchestra

**Observation:** Formalization language: OWL/ RDF



Class hierarchy: S(n):Musician_GID-588967

Asserted

- owl:Thing
  - S(n):Entity_GID-101
    - S(n):MusicalInstrument_GID-566785
      - S(n):WindInstrument_GID-5667857
      - S(n):StringedInstrument_GID-5667855
        - mi:Dulcimer_GID-5667857
        - mi:Guitar_GID-5667855
        - mi:Koto_GID-5667853
      - S(n):KeyboardInstrument_GID-5667853
    - S(n):Orchestra_GID-56672
    - S(n):Musician_GID-588967
      - mi:Guitarist_GID-588967822

Usage: S(n):Musician_GID-588967

Show: ☑ this ☑ disjoints ☑ named sub/superclasses

Found 9 uses of 'S(n):Musician_GID-588967'

- mi:Guitarist_GID-588967822
  - mi:Guitarist_GID-588967822 SubClassOf 'S(n):Musician_GID-588967'
- S(n):Musician_GID-588967
  - S(n):Musician_GID-588967' SubClassOf PartOf some 'S(n):Orchestra_GID-56672
  - S(n):Musician_GID-588967' SubClassOf 'S(n):Entity_GID-101
  - 'S(n):Musician_GID-588967' rdfs:label "S(n):Musician_GID-588967"@en
  - Class: 'S(n):Musician_GID-588967'

Description: S(n):Musician_GID-588967

Equivalent To

SubClass Of

- 'S(n):Entity_GID-101'
- PartOf some 'S(n):Orchestra_GID-56672'

General class axioms

## LOD theories

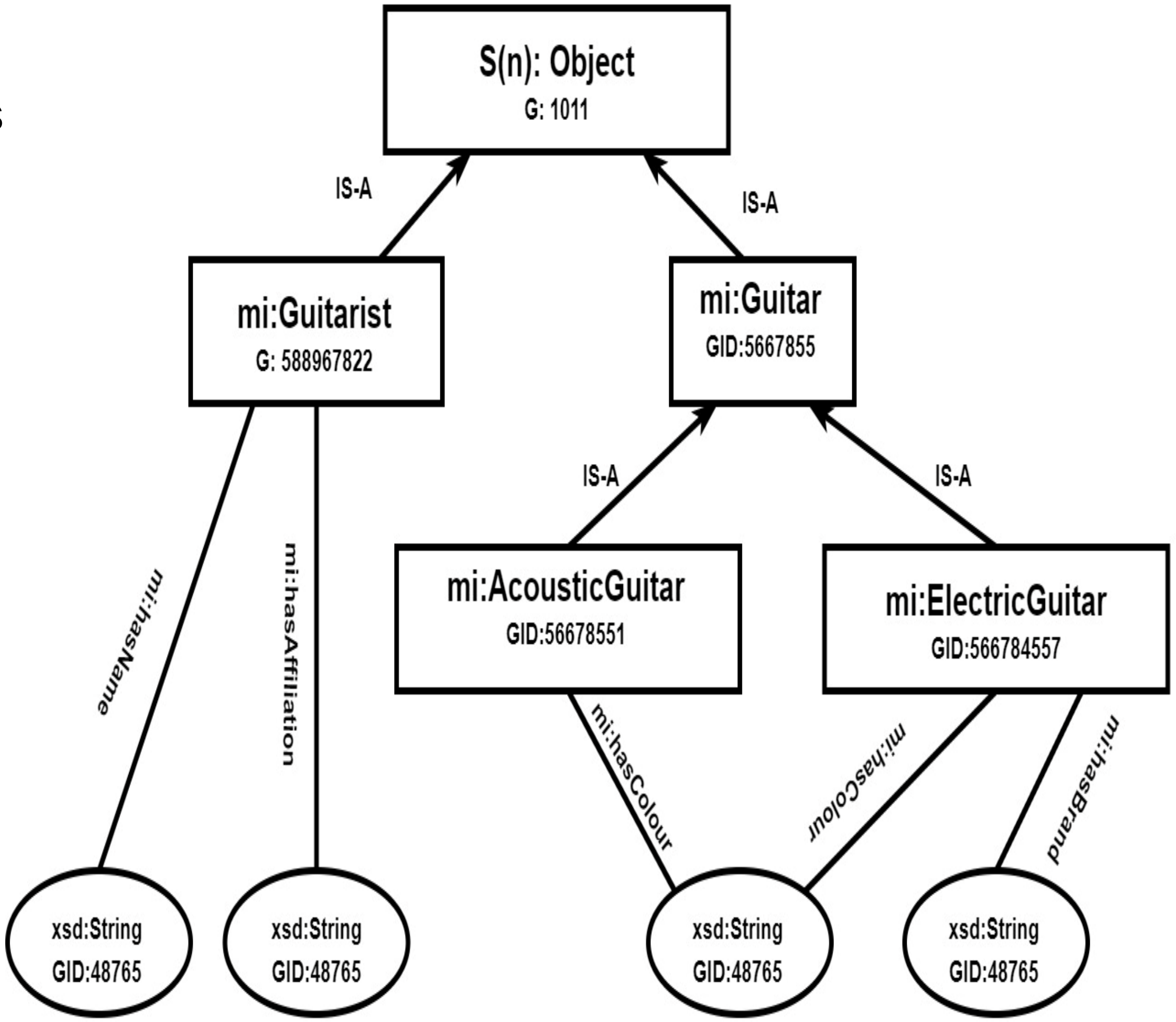
- Lexicons
- Lexical teleontologies
- Knowledge teleontologies
- Teleologies
- Example

# Lexical Teleontology

Lexical Teleontologies are specialized lexicons which focus on a purpose-specific (application dependent) part of a lexicon.

Lexical teleontologies are obtained from lexicons by:

- Identifying the root concept, as from the purpose. This is a whole defining the reference space or time containment
- Keeping the relevant concepts among those which are part-of the root in the PART-OF hierarchy
- Keeping the relevant concepts which specialize, via the IS-A and PART-OF hierarchies, the concepts from previous step
- Keeping the part-of relations as needed
- Dropping irrelevant concepts (below/ above the whole)
- Substituting the root concept with most specific concept in the IS-A hierarchy which subsumes all the parts (e.g., from Orchestra to Object



# LOD – Lexical teleontology formalization

A lexical teleontology is obtained from a formalized lexicon according to the process described in the previous slide, that is:

- Eliminate the irrelevant concepts
- Define the root concept
- Add in the metadata information about the whole being formalized.

# LOD - Lexical teleontology Example (Protégé)

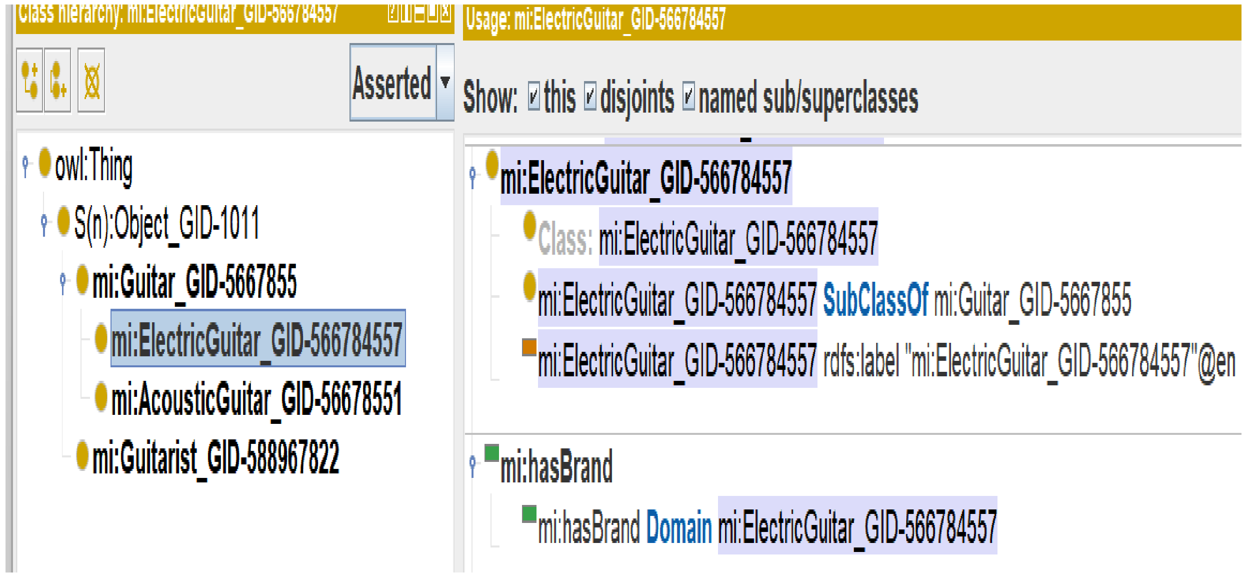
The snippet on the right side shows the lexical teleontology formalized via the Protégé ontology editor.

You can see the entire class hierarchy starting from Object downwards depicting the lexical teleontlogy concepts, with their unique GIDs. Irrelevant concepts such as Koto, Dulcimer, etc. have been eliminated.

You can see (partially) data properties, e.g., ElectricGuitar - mi:hasBrand - String

You can also see (partially) visualization of LOD formalization of lexical teleontology,

e.g., Electric Guitar is: SubClassOf Guitar.



Class hierarchy: mi:ElectricGuitar_GID-566784557

Asserted

- owl:Thing
  - S(n):Object_GID-1011
    - mi:Guitar_GID-5667855
      - mi:ElectricGuitar_GID-566784557
      - mi:AcousticGuitar_GID-56678551
    - mi:Guitarist_GID-588967822

Usage: mi:ElectricGuitar_GID-566784557

Show: ☑ this ☑ disjoints ☑ named sub/superclasses

- mi:ElectricGuitar_GID-566784557
  - Class: mi:ElectricGuitar_GID-566784557
  - mi:ElectricGuitar_GID-566784557 SubClassOf mi:Guitar_GID-5667855
  - mi:ElectricGuitar_GID-566784557 rdfs:label "mi:ElectricGuitar_GID-566784557"@en
- mi:hasBrand
  - mi:hasBrand Domain mi:ElectricGuitar_GID-566784557

## LOD theories

- Lexicons
- Lexical teleontologies
- Knowledge teleontologies
- Teleologies
- Example

## Knowledge teleontology

Knowledge Teleontologies extend lexical teleontologies by

- Adding new additional concepts (as etypes) and dtypes as needed by the application
- For each new or already existing etype, add new object and data properties which describe the specific aspects which are relevant to the selected context (the whole and parts selected in the lexical teleontology)

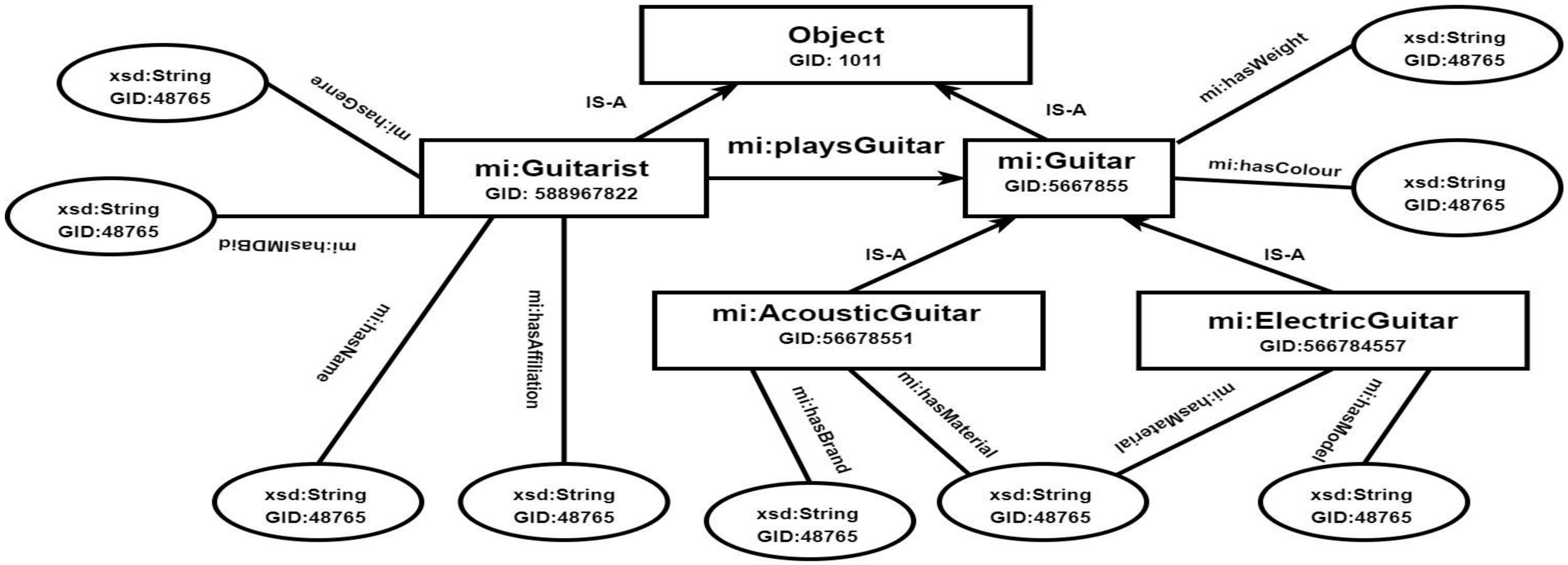
**Observation 1:** lexical teleontologies/ lexicons define linguistic concepts modeling the elements of the world.

**Observation 2:** Knowledge teleontologies describe knowledge concepts by providing relevant local description properties

**Observation 3:** Knowledge teleontologies "are" formalized EER models

**Terminology:** We drop the attribute lexical/ knowledge when the context makes clear the meaning

# Knowledge teleontology (example)



## LOD – teleontology formalization

*Lexical teleontology*: Definition from the above example of musical instruments

$\text{ElectricGuitar} \equiv \text{Guitar} \sqcap \forall \text{hasSoundAmplification.withInputJack}$

Plus disjointness axioms

- $\text{ElectricGuitar} \sqsubseteq \neg\, \text{AcousticGuitar}$
- *… more disjointness axioms*

*Knowledge Teleontology*: Description of the concept define in a lexical ontology:

$\text{ElectricGuitar\#1} \equiv \text{ElectricGuitar} \sqcap \exists \text{hasColour.String} \sqcap \exists \text{hasBrand.String}$

## LOD – teleontology formalization (cont)

**Observation 1:** ElectricGuitar#1 $\sqsubseteq$ ElectricGuitar

**Observation 2**: A description is a definition enriched with data properties (only conjuncts)

**Observation 3**: A description (like a definition)contains only conjuncts

**Observation 4**: The same definition can be associated multiple diverse descriptions

**Observation 5**: when building a description constraints (e.g., no disjointness constraints) on the selected properties used only if needed as part of the description

**Observation 6**: no changes to the PART-OF hierarchy (additional etypes are only specializations generalizations based on properties)

# Teleontology - Example

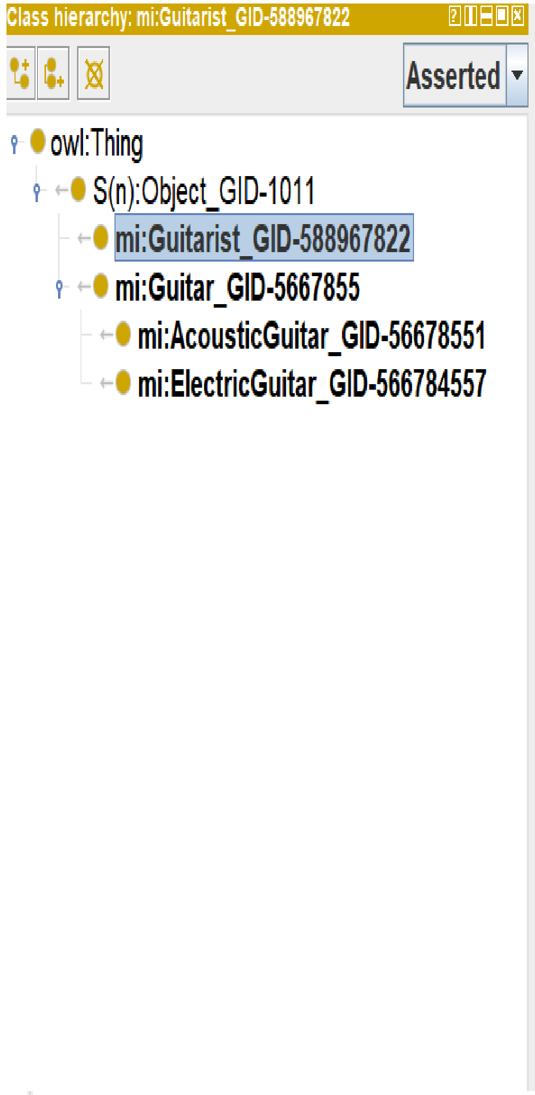
The snippets on the right hand side shows the knowledge teleontology formalized via the Protégé ontology editor.

We add object properties (e.g., playsGuitar) and additional data properties (e.g., hasIMDBid) here.

You can see some (partial) visualization of LOD formalization for, e.g.,

e.g., *mi:Guitarist - mi:PlaysGuitar - mi:Guitar* is an object property-based assertion which indicates that a guitarist plays a guitar.

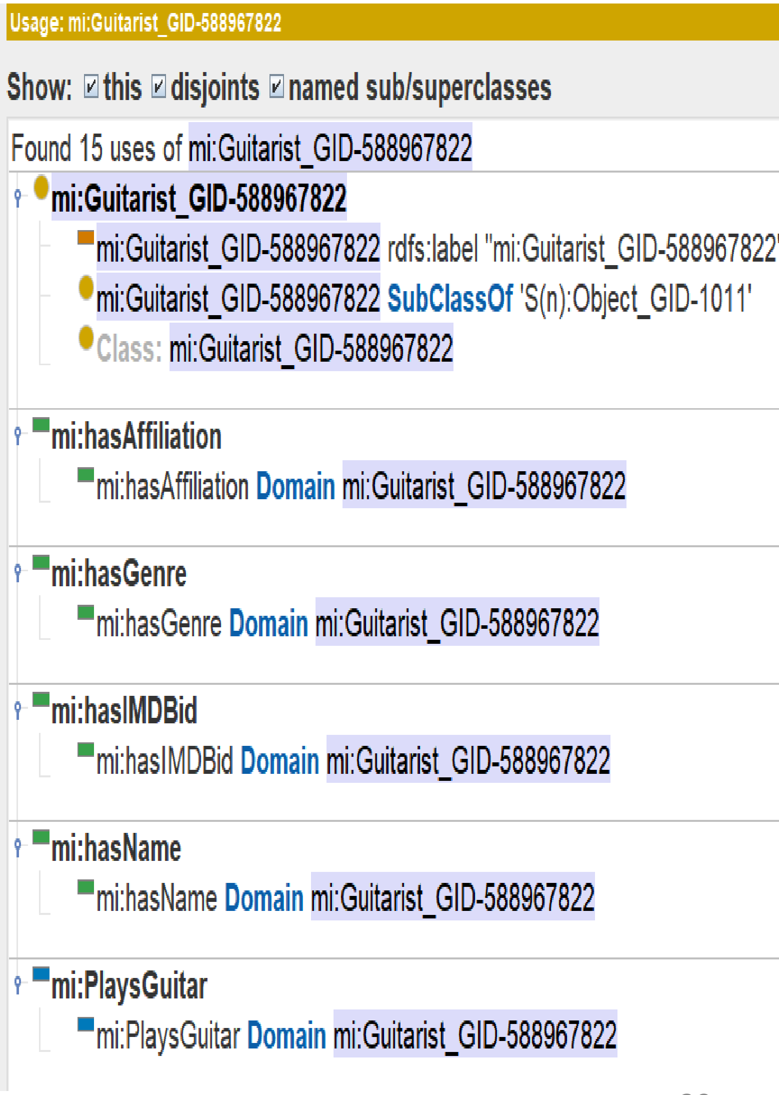
*mi:Guitarist - mi:hasIMDBid - xsd:String* is a data property-based assertion which indicates that a guitarist has an IMDB id encoded as a string.



Class hierarchy: mi:Guitarist_GID-588967822

Asserted

- owl:Thing
  - S(n):Object_GID-1011
    - mi:Guitarist_GID-588967822
    - mi:Guitar_GID-5667855
      - mi:AcousticGuitar_GID-56678551
      - mi:ElectricGuitar_GID-566784557



Usage: mi:Guitarist_GID-588967822

Show: ☑ this ☑ disjoints ☑ named sub/superclasses

Found 15 uses of mi:Guitarist_GID-588967822

- mi:Guitarist_GID-588967822
  - mi:Guitarist_GID-588967822 rdfs:label "mi:Guitarist_GID-588967822"
  - mi:Guitarist_GID-588967822 SubClassOf 'S(n):Object_GID-1011'
  - Class: mi:Guitarist_GID-588967822
- mi:hasAffiliation
  - mi:hasAffiliation Domain mi:Guitarist_GID-588967822
- mi:hasGenre
  - mi:hasGenre Domain mi:Guitarist_GID-588967822
- mi:hasIMDBid
  - mi:hasIMDBid Domain mi:Guitarist_GID-588967822
- mi:hasName
  - mi:hasName Domain mi:Guitarist_GID-588967822
- mi:PlaysGuitar
  - mi:PlaysGuitar Domain mi:Guitarist_GID-588967822

## LOD theories

- Lexicons
- Lexical teleontologies
- Knowledge teleontologies
- Teleologies
- Example

## Teleology / ETG

Teleologies are flattened teleontologies.

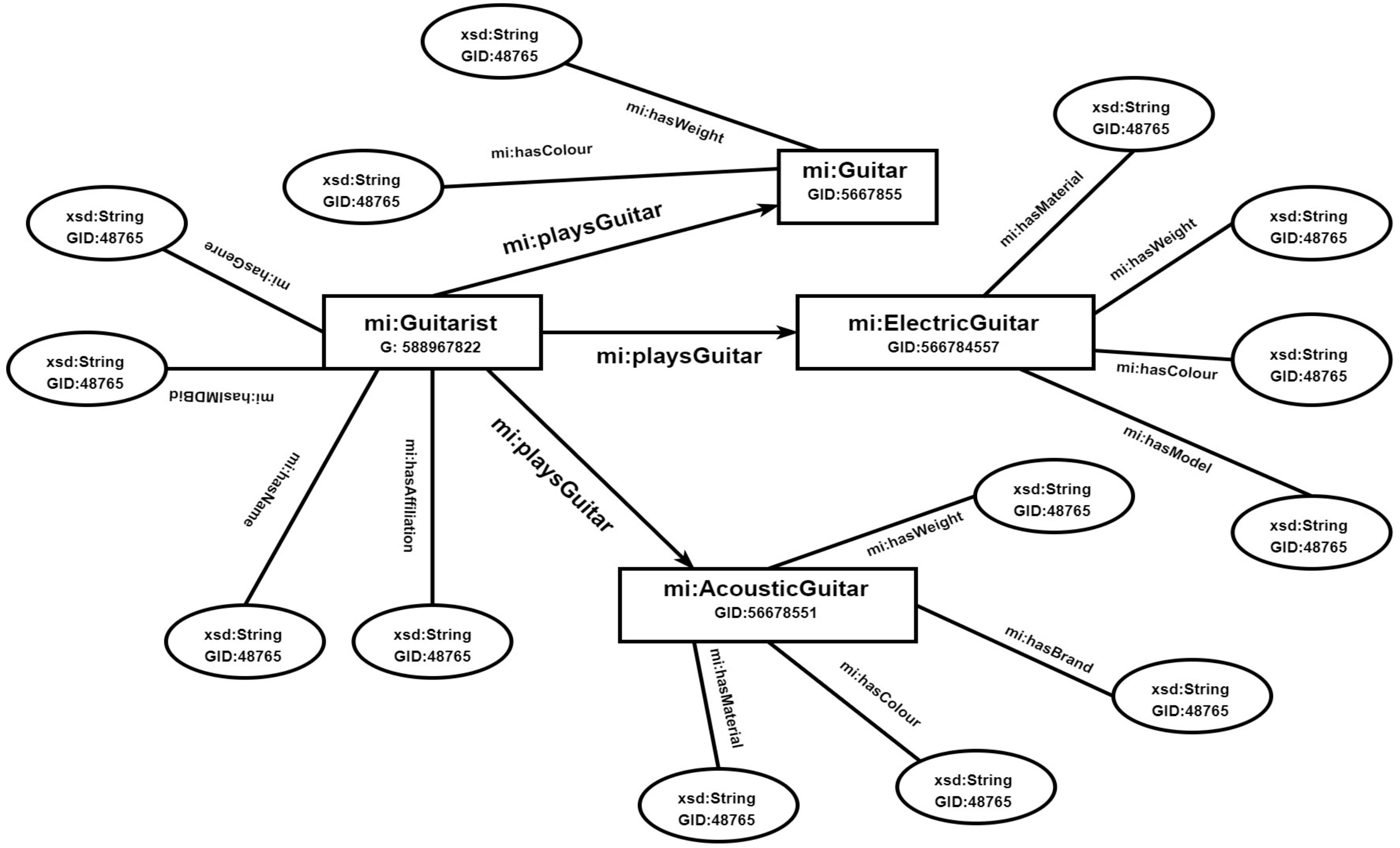
Flattening Process:

- Starting from the root, each concept is defined in terms of the concept one level above in the hierarchy
- Remove the dependence from the more general concept by expanding the definition
- Iterate the process till the leaf nodes

**Observation 1**: Teleologies are formalizations of ER models

**Observation 2:** Teleologies are formalisations of ETGs

## Teleology


mi:Guitarist
G: 588967822
mi:Guitar
GID:5667855
mi:ElectricGuitar
GID:566784557
mi:AcousticGuitar
GID:56678551
mi:playsGuitar
mi:playsGuitar
mi:playsGuitar
mi:hasWeight
mi:hasColour
mi:hasGenre
mi:hasIMDBid
mi:hasName
mi:hasAffiliation
mi:hasMaterial
mi:hasWeight
mi:hasColour
mi:hasModel
mi:hasWeight
mi:hasBrand
mi:hasColour
mi:hasMaterial
xsd:String
GID:48765

# LOD – Teleology formalization

ISA-Hierarchy: From the above example of musical instruments teleontology :

ElectricGuitar#1 ⊑ ElectricGuitar ⊓∃hasColour.String ⊓∃hasBrand.String

-- [description = definition enriched with data properties]

AcousticGuitar#1 ⊑ AcousticGuitar ⊓∃hasMaterial.String ⊓∃hasModel.String

-- [description = definition enriched with data properties]

Guitarist (leaf in teleology) ≡

Musician (implicit in teleology) ⊓∃hasName.String ⊓∃hasAffiliation.String

PART-OF-Hierarchy: From the above example of musical instruments teleontology

## Teleology - Example

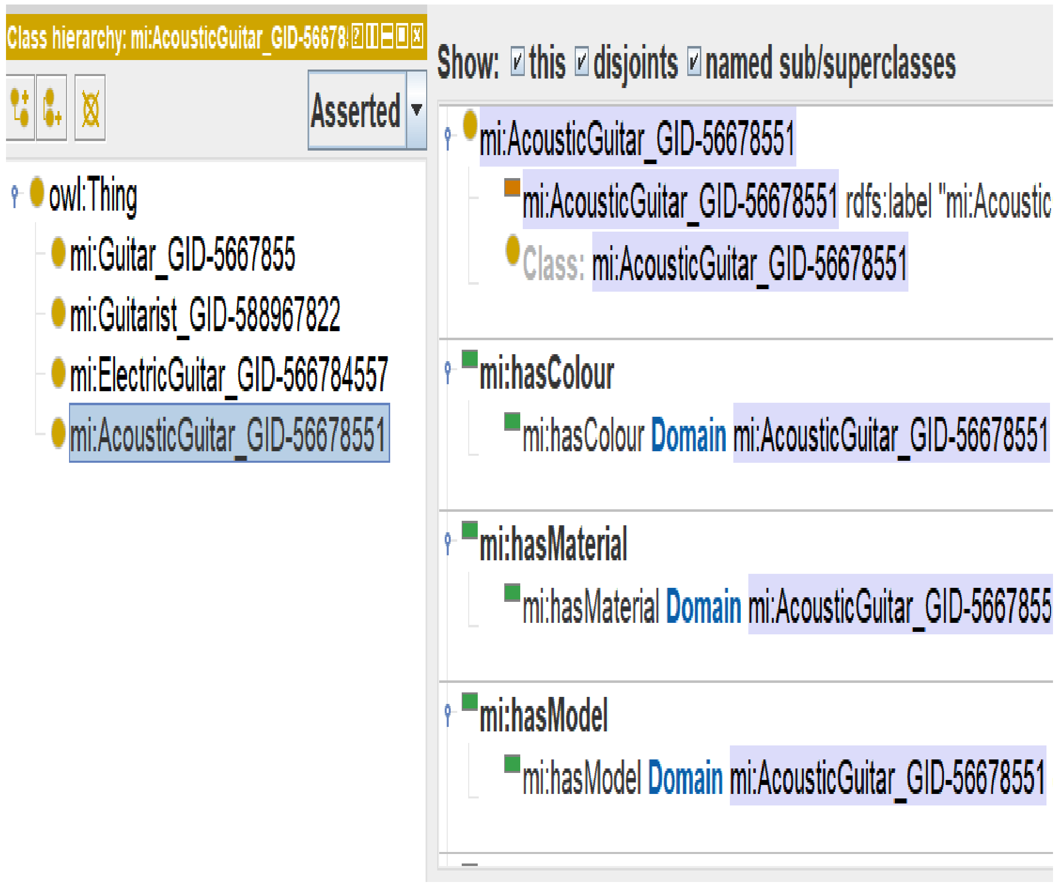
The snippet on the right hand side shows (partially) the teleology formalized via the Protégé ontology editor.

Notice that the class hierarchy is completely flattened, i.e., there are no IS-A links asserting superclass-subclass subsumption relationships.

You can see some (partial) visualization of LOD formalization for, e.g.,

e.g., *mi:AcousticGuitar - mi:hasColour - xsd:String* is a data property-based assertion which indicates that an acoustic guitar has a color which is encoded as a String.

e.g., *mi:AcousticGuitar - mi:hasModel - xsd:String* is a data property-based assertion which indicates that an acoustic guitar is of a specific model spcification encoded as a String.

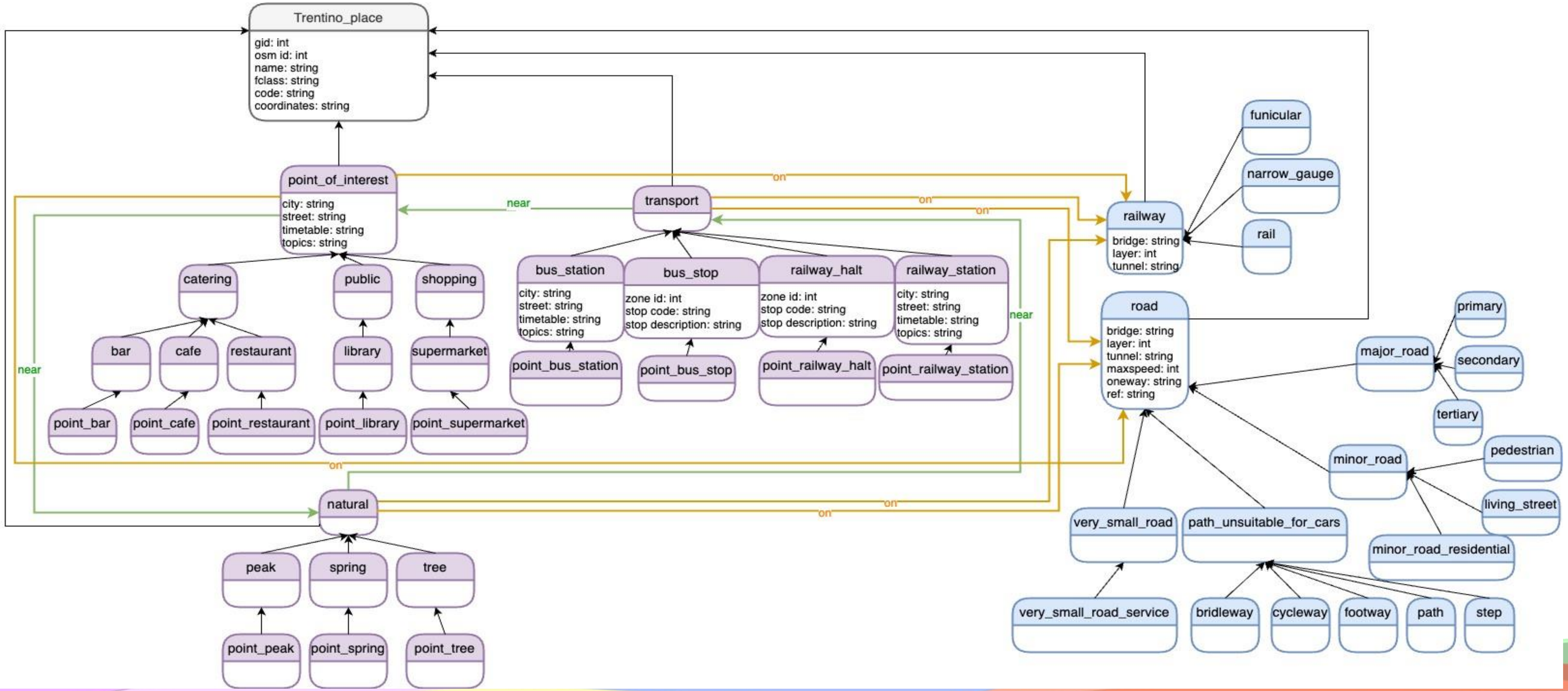

Class hierarchy: mi:AcousticGuitar_GID-56678
Asserted
owl:Thing
mi:Guitar_GID-5667855
mi:Guitarist_GID-588967822
mi:ElectricGuitar_GID-566784557
mi:AcousticGuitar_GID-56678551

Show: ☑ this ☑ disjoints ☑ named sub/superclasses
mi:AcousticGuitar_GID-56678551
mi:AcousticGuitar_GID-56678551 rdfs:label "mi:Acoustic
Class: mi:AcousticGuitar_GID-56678551
mi:hasColour
mi:hasColour Domain mi:AcousticGuitar_GID-56678551
mi:hasMaterial
mi:hasMaterial Domain mi:AcousticGuitar_GID-5667855
mi:hasModel
mi:hasModel Domain mi:AcousticGuitar_GID-56678551

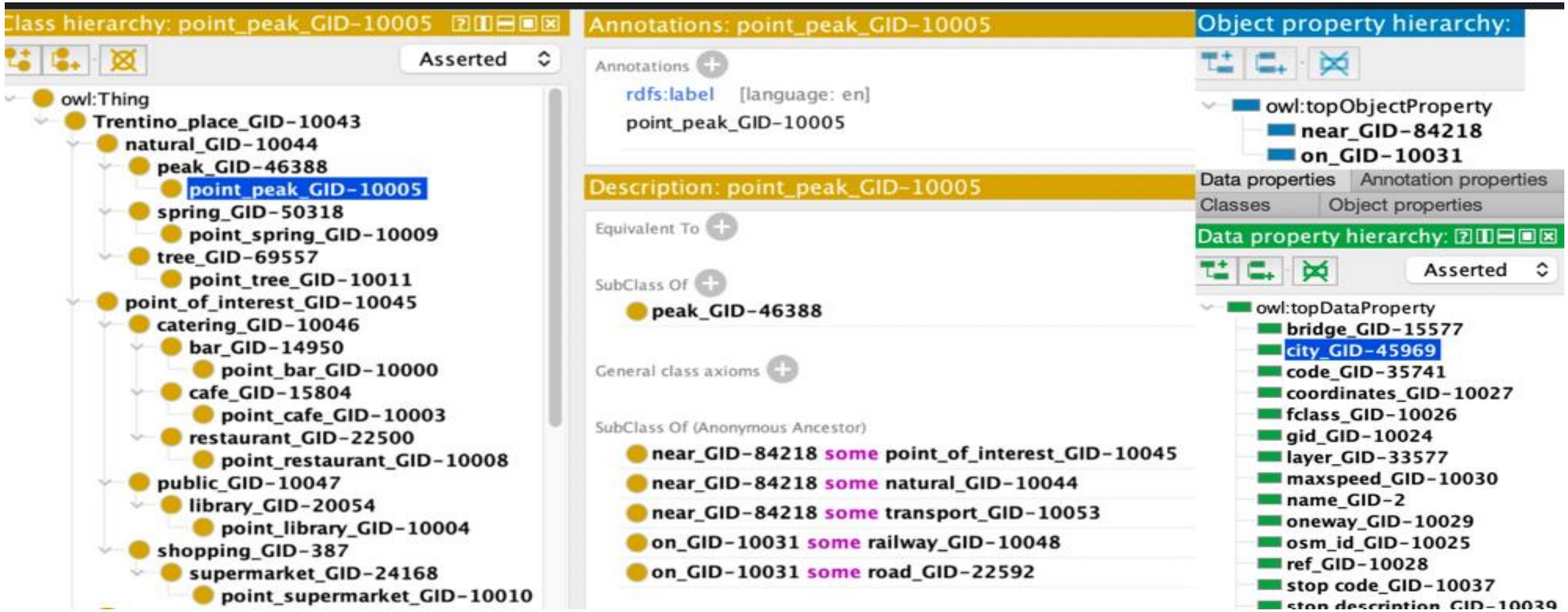
## LOD theories

- Lexicons
- Lexical teleontologies
- Knowledge teleontologies
- Teleologies
- Example

# EER diagram of Open Street Maps data



# Teleontology of Open Street Maps data



Class hierarchy: point_peak_GID-10005

Asserted

- owl:Thing
  - Trentino_place_GID-10043
    - natural_GID-10044
      - peak_GID-46388
        - point_peak_GID-10005
      - spring_GID-50318
        - point_spring_GID-10009
      - tree_GID-69557
        - point_tree_GID-10011
    - point_of_interest_GID-10045
      - catering_GID-10046
        - bar_GID-14950
          - point_bar_GID-10000
        - cafe_GID-15804
          - point_cafe_GID-10003
        - restaurant_GID-22500
          - point_restaurant_GID-10008
      - public_GID-10047
        - library_GID-20054
          - point_library_GID-10004
      - shopping_GID-387
        - supermarket_GID-24168
          - point_supermarket_GID-10010

Annotations: point_peak_GID-10005

Annotations

rdfs:label [language: en]
point_peak_GID-10005

Description: point_peak_GID-10005

Equivalent To

SubClass Of
- peak_GID-46388

General class axioms

SubClass Of (Anonymous Ancestor)
- near_GID-84218 some point_of_interest_GID-10045
- near_GID-84218 some natural_GID-10044
- near_GID-84218 some transport_GID-10053
- on_GID-10031 some railway_GID-10048
- on_GID-10031 some road_GID-22592

Object property hierarchy:

- owl:topObjectProperty
  - near_GID-84218
  - on_GID-10031

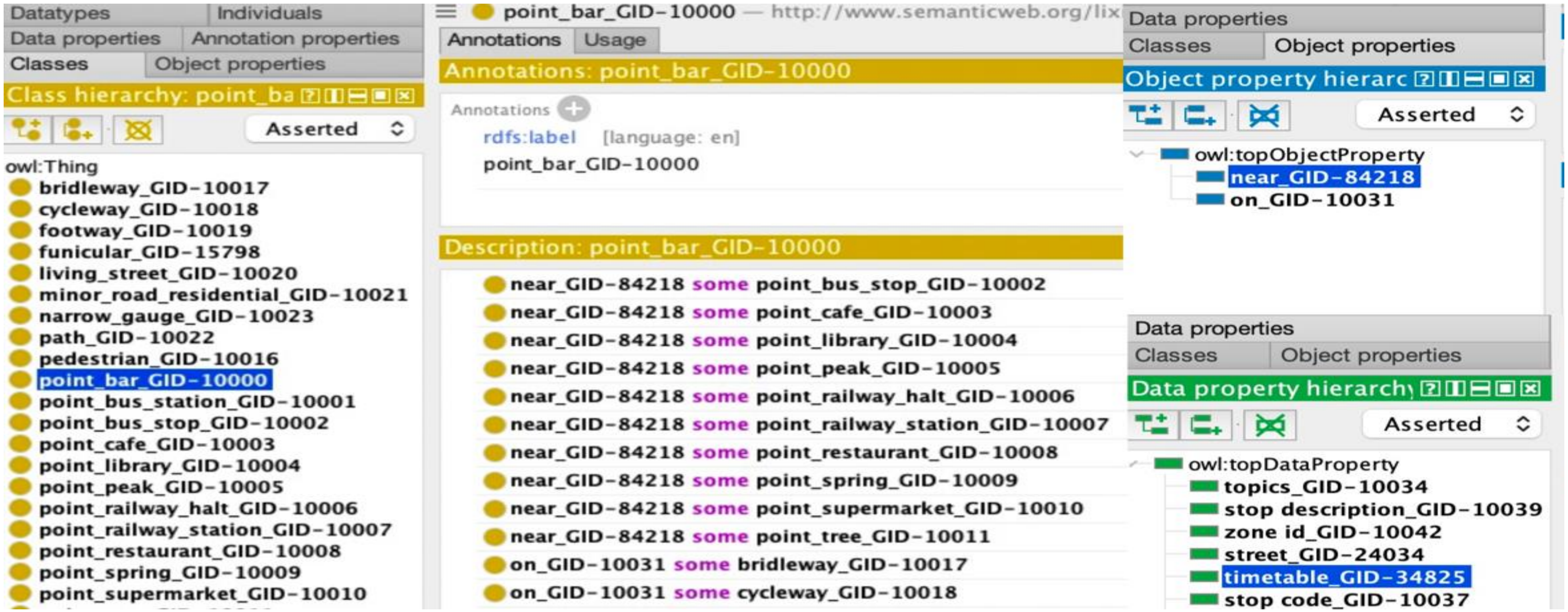
Data properties | Annotation properties
Classes | Object properties

Data property hierarchy:

Asserted

- owl:topDataProperty
  - bridge_GID-15577
  - city_GID-45969
  - code_GID-35741
  - coordinates_GID-10027
  - fclass_GID-10026
  - gid_GID-10024
  - layer_GID-33577
  - maxspeed_GID-10030
  - name_GID-2
  - oneway_GID-10029
  - osm_id_GID-10025
  - ref_GID-10028
  - stop code_GID-10037

# Teleology of Open Street Maps data



owl:Thing
- bridleway_GID-10017
- cycleway_GID-10018
- footway_GID-10019
- funicular_GID-15798
- living_street_GID-10020
- minor_road_residential_GID-10021
- narrow_gauge_GID-10023
- path_GID-10022
- pedestrian_GID-10016
- point_bar_GID-10000
- point_bus_station_GID-10001
- point_bus_stop_GID-10002
- point_cafe_GID-10003
- point_library_GID-10004
- point_peak_GID-10005
- point_railway_halt_GID-10006
- point_railway_station_GID-10007
- point_restaurant_GID-10008
- point_spring_GID-10009
- point_supermarket_GID-10010

point_bar_GID-10000 — http://www.semanticweb.org/lix

Annotations: point_bar_GID-10000

Annotations
rdfs:label [language: en]
point_bar_GID-10000

Description: point_bar_GID-10000

- near_GID-84218 some point_bus_stop_GID-10002
- near_GID-84218 some point_cafe_GID-10003
- near_GID-84218 some point_library_GID-10004
- near_GID-84218 some point_peak_GID-10005
- near_GID-84218 some point_railway_halt_GID-10006
- near_GID-84218 some point_railway_station_GID-10007
- near_GID-84218 some point_restaurant_GID-10008
- near_GID-84218 some point_spring_GID-10009
- near_GID-84218 some point_supermarket_GID-10010
- near_GID-84218 some point_tree_GID-10011
- on_GID-10031 some bridleway_GID-10017
- on_GID-10031 some cycleway_GID-10018

owl:topObjectProperty
- near_GID-84218
- on_GID-10031

owl:topDataProperty
- topics_GID-10034
- stop description_GID-10039
- zone id_GID-10042
- street_GID-24034
- timetable_GID-34825
- stop code_GID-10037

## LOD - The logic of Descriptions

- TBoxes and terminologies
- LOD theories
- Unfolding
- Reasoning

## Unfolding a Concept (notion)

**Definition (Definiendum, definiens):** The left hand side of a definition A ≡ C (that is, A) is called *definiendum,* the right hand side (that is, C) is called the *definiens.*

**Definition (Defined and primitive concept):** Given a TBox, a *defined* concept is a concept which appears on the left of a definition of the TBox. A *primitive* concept is a concept which only appears on the right of the definitions.

**Definition (Concept unfolding)** A defined concept is *unfolded* if all the defined concepts occurring in its definiens are substituted with their definition.

**Observation (unfolded concept):** The definition of an unfolded concept contains only primitive concepts

**Example.** *From*:

ElectricGuitar ≡ Guitar ⊓ ∀hasSoundAmplification.withInputJack

ElectricGuitar#1 ≡ ElectricGuitar ⊓ ∃hasColour.String ⊓ ∃hasBrand.String

*To*:

ElectricGuitar#1 ≡ Guitar ⊓ ∀hasSoundAmplification.withInputJack ⊓ ∃hasColour.String ⊓ ∃hasBrand.String

**Remark:** In an acyclic terminology the process of concept unfolding can be applied recursively up to any level, with the possibility to primitive concepts (etypes).

## Unfolding a TBox (notion)

**Observation (Defined and primitive concept):** We restrict to the case where a defined concept appears once on the left or a definition. Defined and primitive concepts can appear on the right of definitions any number of times.

**Definition (TBox unfolding).** A definitional TBox T can be unfolded into a Tbox T′ by (recursively) unfolding all its defined concepts.

**Theorem:** Let T be a terminology . Let T' the result of unfolding T. Then M is a model of T if and only if it is a model of T'.

## Complexity of unfolding

TBox definitions are like macros that can be expanded into primitive concepts

The size of the unfolded TBox grows exponentially with the depth of the TBox induced subsumption hierarchy. For instance, from

$$A2 \equiv \exists A3 \sqcap \forall A3$$
$$A1 \equiv \exists A2 \sqcap \forall A2$$
$$A0 \equiv \exists A1 \sqcap \forall A1$$

We obtain

$A1 \equiv \exists(\exists A3 \sqcap \forall A3) \sqcap \forall(\exists A3 \sqcap \forall A3)$ *(2 times)*

$A0 \equiv \exists(\exists(\exists A3 \sqcap \forall A3) \sqcap \forall(\exists A3 \sqcap \forall A3)) \sqcap \forall(\exists(\exists A3 \sqcap \forall A3) \sqcap \forall(\exists A3 \sqcap \forall A3))$ *(4 times)*

## TBox to unfold (example 1 (reprise))

- ~~Person ≡ ∃hasname.String ⊓ ∀HasJob.Organization~~
- Woman ≡ Person ⊓ Female
- Man ≡ Person ⊓ ¬ Woman
- Mother ≡ Woman ⊓ ∃hasChild.Person
- Father ≡ Man ⊓ ∃hasChild.Person
- Parent ≡ Father ⊔ Mother

**Observation 1**: The deletion of the first definition makes Person a primitive concept. The choice of the primitive terms is up to the modeler

**Observation 2**: definition of Man not minimalistic (Woman instead of Female)

**Observation 3**: definition of Parent redundant (IsParent ≡ HasChild)

$$\mathcal{T}' = \begin{cases} Woman \equiv Person \sqcap Female \\ Man \equiv Person \sqcap \neg(Person \sqcap Female) \\ Mother \equiv (Person \sqcap Female) \sqcap \exists hasChild.Person \\ Father \equiv (Person \sqcap \neg(Person \sqcap Female)) \sqcap \exists hasChild.Person \\ Parent \equiv (Person \sqcap \neg(Person \sqcap Female)) \sqcap \\ \qquad \exists hasChild.Person \sqcup (Person \sqcap Female) \sqcap \exists hasChild.Person \end{cases}$$

**Observation**: Unfolding generates disjunctions. In fact (check Venn Diagram

$$\neg(A \sqcap B) \equiv \neg A \sqcup \neg B$$

## TBox to unfold (example 2 (example 1 revised))

Woman ≡ Person ⊓ Female
Man ≡ Person ⊓ ¬ Female
Mother ≡ Woman ⊓ ∃HasChild.Person
Father ≡ Man ⊓ ∃HasChild.Person
~~isParent ≡ HasChild~~

**Observation 1**: The dependence of Man on Woman has been eliminated

**Observation 2**: Parent has been formalized for what it is, that is the inverse relation of HasChild (IsParent ≡ HasChild)

**Observation 3**: The TBox has now become a teleontology

## TBox unfolded (example 2)

$$\begin{aligned}
\text{Woman} &\equiv \text{Person} \sqcap \text{Female} \\
\text{Man} &\equiv \text{Person} \sqcap \neg\, \text{Female} \\
\text{Mother} &\equiv \text{Person} \sqcap \text{Female} \sqcap \exists \text{HasChild.Person} \\
\text{Father} &\equiv \text{Person} \sqcap \neg\, \text{Female} \sqcap \exists \text{HasChild.Person}
\end{aligned}$$

## TBox unfolded (example 2 (cont))

**Observation 1**: The flattened TBox contains only conjuncts, no disjuncts

**Observation 2**: Female ≡ ∃Female.T means that there are no restrictions on the codomain of Female

**Observation 3:** ¬ Female ≡ ∀Female.⊥ means that there are no females (coherently with the definition of Man)

**Observation 4**: The flattened TBox is a teleology constructed by flattening a teleontology (no dijuncts generated as negation applied only to atomic assertions)

**Theorem:** The unfolding of a teleontology always results into a teleology where definitions contain only coniunctions

## TBox to unfold (example 3 (reprise))

- ~~Undergraduate $\sqsubseteq \neg$ Teach~~
- Bachelor $\equiv$ Student $\sqcap$ Undergraduate
- Master $\equiv$ Student $\sqcap \neg$ Undergraduate
- PhD $\equiv$ Master $\sqcap$ Research
- Assistant $\equiv$ PhD $\sqcap$ Teach

**Observation 1**: The inclusion is more a description rather than a definition

**Observation 2**: The TBox resulting from the deletion is a teleontology

**Observation 3**: The TBox has now become a lexical teleontology

## TBox unfolded (example 3 (cont))

- Bachelor $\equiv$ Student $\sqcap$ Undergraduate
- Master $\equiv$ Student $\sqcap \neg$ Undergraduate
- PhD $\equiv$ Student $\sqcap \neg$ Undergraduate $\sqcap$ Research
- Assistant $\equiv$ Student $\sqcap \neg$ Undergraduate $\sqcap$ Research $\sqcap$ Teach

**Observation**: The inclusion

$$\text{Undergraduate} \sqsubseteq \neg\, \text{Teach}$$

was not needed as Teach applies only to students who are not undergraduate

## Complexity of unfolding of a teleontology

Teleontologies are like macros that can be expanded into teleologies

The size of the unfolded definition grows polinomially with the depth of the teleontology. For instance, from

$$L2 \equiv G2 \sqcap D2$$
$$L1 \equiv L2 \sqcap D1$$
$$L0 \equiv L1 \sqcap D0$$

we obtain

$$L1 \equiv G2 \sqcap D2 \sqcap D1 \quad \text{(times 2+1 Differentia)}$$
$$L0 \equiv G2 \sqcap D2 \sqcap D1 \sqcap D0 \quad \text{(times 2 +2 Differentia)}$$

**Observation 1**: The number of paths grows exponentialy with the number of childrens for the same node

**Observation 2**: The growth is local to the subtree and not global

# LOD - The logic of Descriptions

- TBoxes and terminologies
- LOD theories
- Unfolding
- Reasoning

## Entailment (reprise)

**Definition (Entailment |=)**

1. M $\models w1 \sqsubseteq w2$ iff $I(w1) \subseteq I(w2)$
2. M $\models w1 \equiv w2$ iff $I(w1) = I(w2)$
   iff $I(w1) \subseteq I(w2)$ and $I(w2) \subseteq I(w1)$
3. M $\models w1 \sqsubseteq \neg w2$ iff $I(w1) \cap I(w2) \subseteq \emptyset$

**with**

- $w1, w2 \in L$;
- $w1 \sqsupseteq w2$ a notational variant of $w2 \sqsubseteq w1$;
- $w1 \perp w2$ a notational variant of $w1 \sqsubseteq \neg w2$

## Entailment with teleontologies

**Observation 1 (reprise)**: We are in presence of TBoxes with only conjuncts

**Observation 2**: Teleontologies are nested subsumption hierarchies. Teleologies / ETGs are constructed from teleontologies via unfolding

**Observation 3**: All the reasoning problems for a teleontology can be solved in a Teleology (which contains only primitive concepts)

## Entailment with teleontologies )

**Theorem 1**: M |= $w1 \sqsubseteq w2$ iff, after unfolding, the conjuncts in $w1$ and in $w2$ are a superset or equal to those of $w2$

**Theorem 2**: M |= $w1 \equiv w2$ iff, after unfolding, the conjunts in $w1$ and in $w2$ are exactly the same as those of $w2$

**Theorem 3** : M |= $w1 \sqsubseteq \neg\, w2$ iff, after unfolding, one of the conjunts occurs negated in $w1$ and not negated in $w2$, or vice versa

**Observation:** all the LOD reasoning problems for teleontologies can be resolved byunflding them into teleologies

## Satisfiability with teleontologies (reprise)

- Bachelor $\equiv$ Student $\sqcap$ Undergraduate
- Master $\equiv$ Student $\sqcap \neg$ Undergraduate
- PhD $\equiv$ Student $\sqcap \neg$ Undergraduate $\sqcap$ Research
- Assistant $\equiv$ Student $\sqcap \neg$ Undergraduate $\sqcap$ Research $\sqcap$ Teach

Is

Bachelor $\sqcap$ PhD

satisfiable? NO!

## Subsumption with teleontologies (reprise)

- Bachelor $\equiv$ Student $\sqcap$ Undergraduate
- Master $\equiv$ Student $\sqcap \neg$ Undergraduate
- PhD $\equiv$ Student $\sqcap \neg$ Undergraduate $\sqcap$ Research
- Assistant $\equiv$ Student $\sqcap \neg$ Undergraduate $\sqcap$ Research $\sqcap$ Teach

Is

$$\text{PhD} \sqsubseteq \text{Student}$$

satisfiable? YES!

## Equivalence with teleontologies (reprise)

- Bachelor $\equiv$ Student $\sqcap$ Undergraduate
- Master $\equiv$ Student $\sqcap \neg$ Undergraduate
- PhD $\equiv$ Student $\sqcap \neg$ Undergraduate $\sqcap$ Research
- Assistant I($A$ $\sqcup$ $\equiv$ Student $\sqcap \neg$ Undergraduate $\sqcap$ Research $\sqcap$ Teach

Is (assuming extended query language)

$$\text{Student} \equiv \text{Bachelor} \sqcup \text{Master}$$

satisfiable? YES!

## Disjointness with teleontologies (reprise)

- Bachelor $\equiv$ Student $\sqcap$ Undergraduate
- Master $\equiv$ Student $\sqcap \neg$ Undergraduate
- PhD $\equiv$ Student $\sqcap \neg$ Undergraduate $\sqcap$ Research
- Assistant $\equiv$ Student $\sqcap \neg$ Undergraduate $\sqcap$ Research $\sqcap$ Teach

Is

$$\text{Assistant} \sqsubseteq \neg\, \text{Undergraduate}$$

satisfiable? YES!

# LOD – The logic of Descriptions

## Theories