Annotation guidelines August – September 2024

For each property

1. Determine if the property is, ontologically, **atomic**, or expresses a **chain** in which the first part is a relationship, and the second part is to be further processed. The chain is sometimes reflected by the presence of more than one content terms in the name of the property (including the term “type” or similar, which would reflect the “instance of” relationship).
    a. If the DP is atomic then continue to step 2.
    b. If the DP expresses a chain then write the (estimated) type of object to the “mediating object” column, and consider the “rest” of the data property semantics as if its domain were the class of such “mediating” objects, in step 2.
2. Depending on the DP type, consider the options below for the given type.
    a. If you consider the DP as corresponding to one of the semantic categories listed then fill the category to the “DP category” column.
    b. Otherwise fill “New” to the “DP category” column, and describe the nature of the new category in the “New category description” column.

String-valued

- **Id - Identifier.** For example, DOI for a publication, or ORCID for a researcher. Sometimes an identifier also plays an additional role, e.g., an e-mail address can not only identify a person but also be decoded wrt. to the server to which a message is to be sent.
    - In PURO: identifiers as such do not matter yet when an ontology sketch is being designed: a PURO object is implicitly assumed to *have its identity*, the details of which can be specified at a later phase. As regards more complex handling of IDs, it is possible to lift them to the level of *objects* (to which further info can be associated) - this holds for OWL and PURO likewise.
- **Name.** A name need not be totally unique as an Id, but also refers to a term under which the entity is known.
    - In PURO, the name is directly used as the label of the entity, i.e., it is not linked to it by a dedicated construct
- **Tp - (Code of) Type.** The string value belongs to a nomenclature of types that are, through a data property assigned to the source object. (A kind of anti-pattern, as the type would always be better expressed by a class or object property.)
    - Example: :MyVolvo v:carType “Sedan” .
    - In PURO: the source object can be assigned a (level-1) *type*.
- **Obj - (Code of) Object.** The string value denotes an object (particular), e.g., an organization the source object is related to. (A kind of anti-pattern, as the relation to an object would always be better expressed by an object property.)
    - Example: :MyVolvo v:manufacturedIn “Stockholm” .
    - In PURO: the source object is associated with another *object* via a *relationship*.
- **Qual - Quality.** The string value expresses some quality, e.g., color, of the object.

    - In PURO: we do not have an explicit construct for expressing qualities. However, it is roughly possible to approximate qualities as objects (particulars), as they do not have “obvious” instances. Thus, it similar to Obj, although we explicitly distinguish it.
- **Msg - Human-readable message.** When it comes to fine-grained ontologies related to software systems, some KG individuals may have associated human-readable messages. However, the essence of the semantic meaning of such messages can be expressed via structured data.
    - In PURO: if the message is important, its structure should be modeled in a *structured* manner (e.g., a warning can express that there is an object of type “Risk” associated by relation “incurred by” with an object of type “User action”). Otherwise the message *need not be modeled* in the sketch at all.

Integer-valued

- **Quant - Inherently quantitative, continuous-valued feature.** Such as size, speed or price. Integer-valued only due to rounding-up.
    - Example: :MyVolvo v:maximumSpeed 210 .
    - In PURO: *valuation*.
- **Count - Counting property.** Expresses the fact that there is a finite number of relationships of the same kind.
    - Example: :JohnDoe v:numberOfCitizenships 2 .
    - In PURO: *relationship* (typically, two of the same type, to indicate multiplicity).

Float

- **Quant - Inherently quantitative, continuous-valued feature.**
    - In PURO: *valuation*.

Date

- **Event - Date/time of event.** The source object is an event with granularity corresponding to the date/time value, which specifies when this event happened. E.g., the date of enrollment.
    - :myEnrollmentToVSE v:date … .
    - In PURO: *valuation*.
- **(RVChain) - Relationship+valuation chain**, namely, date/time of an event associated with an object. The date specifies the time of an instantaneous event to which the source object (which is possibly not an event, although it may an event, too) is somehow related. E.g., the birthdate of a person.
    - In PURO: *relationship* to an (event) *object*, which is then equipped with a (date/time-valued) *valuation*. The first object would be the person, the 2nd object would be his/her birth, and the valuation would provide the date for the birth.

Boolean

- **Rel - (Existentially quantified) relationship.** The 0/1 values indicate whether there is or not a particular relationship to an object - either a *specific one*, or one of *certain kind* (existential quantification).
    - :myVolvo v:hasInsuranceContract true .
    - In PURO: *relationship* to an *object* (equipped with its *type*).
- **Tp - Type.** The 0/1 values indicate whether the source object is instance of a specific type (i.e., a specific subtype of the source object's type).
    - :myVolvo v:isLuxuryCar false .
    - In PURO: *instantiation* wrt. a *type*.

Relationship chain

- **(RRChain) - chain of 2 relationships.** The property expresses what is, in reality, a chain of 2 or more relationships, the types of which correspond to the previous taxonomy.
    - Example: :MyVolvo v:hasTypeOfEngine "Diesel" .
    - In PURO: the individual constituent relationships should better be used explicitly (in the example: relationship between my Volvo and its engine, and the instantiation of the engine to the type Diesel).

Scenarios of guidelines use: with/out chain

Scenario without a chain

The input table (for annotation)

Source (Domain)	DP	Range	DP category	Mediating object (optional)	DP new category suggestion	Comment
Car	car type	xsd:string				

1. Is the dataproperty “car type” a chain?
    a. No
2. What is the range of the property?
    a. String
3. Based on the answer in the second question, look at the categories that could be used for strings. What category is the most suitable?
    a. Tp (Type)

The output:

Source (Domain)	DP	Range	DP category	Mediating object (optional)	DP new category suggestion	Comment
Car	car type	xsd:string	Tp			

Scenario with a chain

The input table

Source (Domain)	DP	Range	DP category	Mediating object (optional)	DP new category suggestion	Comment
Person	birth date	xsd:date				

1. Is the dataproperty “birth date” a chain?
    a. Yes
    b. Imagine the chain: Person - was born - Birth Event - birth date - date
2. What category would be the second part of the chain (Birth Event - birth date - date)?
    a. Event -> fill in the table under DP category
3. What is a mediating object in this chain?
    a. Event - Birth of a person or Birth Event -> fill in the table under mediating object

4. Optionally you can specify in comment section kind of chain, in this case it could be RVChain

The output table

Source (Domain)	DP	Range	DP category	Mediating object (optional)	DP new category suggestion	Comment
Person	birth date	xsd:date	event	Birth Event (Birth of a person)		RVChain