First remarks on the WUMM modelling of a TRIZ ontology

Hans-Gert Gräbe

March 5, 2021

1 Background

As already stated in the *Handreichung zu den Seminararbeiten*, the modelling to be carried out follows the concepts of the *TRIZ Summit Ontology Project* (TOP) [3] and should fit in the framework of the *WUMM Ontology Companion Project* [5]. In contrast to the ≪large-scale ontologisations≫ within TOP, which models the basic interrelationships of different parts of TRIZ theory, the themes of the seminar papers are concerned with semantic modelling of different TRIZ sub-areas in more detail.

2 Abstraction levels of modelling

An ontology is about ≪modelling of models≫, because the clarification of terms and concepts aimed at with an ontology is intended to be practically used in real-world modelling contexts. This ≪modelling of models≫ references a typical engineering context, in which the *modelling* of systems plays a central role and serves as basis of further planned action (including project planning, implementation, operation, maintenance, further development of the system).

In this process, *several levels of abstraction* are to be distinguished, which in TOP is not sufficiently worked out. These are

0. The level of the *real world system* to which the engineering task refers. This level is only *practically* accessible. The model to be developed at level 1 must be appropriate to cover all problems arising in the process of development and use of the system and express the inherent contradictoriness of the system.
   This contradictory nature of the system can be formulated only in language form, i.e. on the model level and *applying* the concepts available there. These concepts must therefore not only be able to describe the system itself, but also cover a description of the necessary aspects of its operation.
1. The *level of modelling* the system. The ontology provides the language means, concepts (RDF subjects) and properties (RDF predicates), which are to be *applied* at this level. This level is also the *level of methodological practice*.
2. The *level of the meta-model* as the actual ontology level on which the systemic concepts are *defined*. This definition is processed *applying* the methodological concepts whose linguistic means are made available on meta-level 2.
3. The *modelling meta-level 2* at which the methodological concepts are defined.

In the examples below, three namespaces play a role:

- `ex:` as the namespace of a special model (level 1).
- `tc:` as the namespace of TRIZ concepts (level 2 subjects).
- `od:` as the namespace of WUMM's own concepts (methodological subjects, level 2 predicates).

3 Basics of the WUMM Ontology Project

The WUMM ontology project accompanies the TOP activities in order

1. to carry out a remodelling according to semantic standards,
2. to enhance the material multilingually and
3. to build an LOD infrastructure on this basis,

and thus to improve the basis for the necessary social coordination processes.

For this purpose, the SKOS ontology [1] is used with the concepts (K)

- `skos:Concept`, `skos:prefLabel`, `skos:altLabel` – concept naming
- `skos:definition`, `skos:example`, `skos:note` – concept properties
- `skos:narrower`, `skos:broader` – concept relations.

It provides an initial descriptive framework for conceptualisations. For the meaning of the individual concepts, please refer to [1].

3.1 URIs and Namespaces

One of the central problems of transferring the existing data stocks on TRIZ concepts is the allocation of meaningful URIs, since the individual glossary entries in the existing sources are identified solely by their labels. The OSA platform is no exception to this since the URIs assigned there (both for the nodes and the edges of the constructed RDF graph) are not publicly visible.

During the computer based transformation of the datasets into a valid RDF format, URIs were automatically generated for all concepts, which are located in the namespace `tc:` (like *TRIZ Concepts*). An essential task still to be done is the unification of these URIs, i.e. merging different URIs that refer to the same concept.

3.2 Provenance of Explanations

A further problem of this ontological modelling is the representation of the provenance of the individual explanations. For this purpose the SKOS concepts listed under (K) were replaced for each individual source by notations from the namespace `od:` in order to first identify the ≪worlds≫ of the individual authors and TRIZ schools separately.

`od:` is the namespace used by the WUMM project to develop its own concepts. A more detailed description of this namespace in a separate RDF file is still pending, the concepts represented by the URIs have so far only been agreed verbally.

Corresponding notation variations are for example

- `skos:Concept` → `od:GSAThesaurusEntry`, `od:VDIGlossaryEntry` ...
- `skos:definition` → `od:SouchkovDefinition`, `od:VDIGlossaryDefinition` ...
- `skos:example` → `od:VDIGlossaryExample` ...

etc. See the RDF data itself, which can be accessed via the SPARQL endpoint

`http://wumm.uni-leipzig.de:8891/sparql`

of the WUMM project.

The *use of different predicate identifiers* allows easily to indicate the origin – e.g. of definitions – to different sources at the level already of triples:

(term X) – (is defined in source A as) – (definition of X in source A)
(term X) – (is defined in source B as) – (definition of X in source B)

If `skos:definition` were used consistently, more complex constructs

$$\text{(term X)} - \texttt{skos:definition} - \left[\begin{array}{l}\text{(source A)} - \text{(definition of X in source A)} \\ \text{(source B)} - \text{(definition of X in source B)}\end{array}\right]$$

had to be used. Although this is the final goal of the exercise, in the current state of the modelling, the simpler form will be used, even though this inflates the number of predicates. A consolidation towards the ≪ideal final result≫ can be achieved later on by a simple model transformation.

The same applies to the use of provenance-dependent subclasses of `skos:Concept`. Since the matching of the URIs assigned to concepts must be carried out across subclasses, the instances of each subclass are also marked as `skos:Concept`, although this could also be done via a general sentence such as

```
od:VDIGlossaryEntry rdfs:subClassOf skos:Concept
```

and then determined by inference. We proceed this way since inferencing belongs to the more complex semantic OWL concepts and is not supported by simple RDF stores. Again, the data can be consolidated later on by a simple model transformation.

4 Typical modelling situations

4.1 Morphological box

A Morphological box describes a concept that can take a finite number of fixed values (a enum data type in Java).

Example: Colour (red, green, yellow, blue).

```
ex:MeiersCar od:hasColour tc:green .

od:hasColour a rdfs:Property;
  rdfs:domain tc:Flow;
  rdfs:range tc:Colour .

tc:Colour a skos:Concept, od:AdditionalConcept ;
  skos:prefLabel "Color"@en, "Farbe"@de ;
  od:allowedValues tc:red, tc:green, tc:yellow, tc:blue .

tc:green a skos:Concept, od:AdditionalConcept ;
  od:valueOf tc:Colour ;
  skos:prefLabel "green"@en, "grün"@de .

...
```

Considered as datatypes `tc:green` is an instance of `tc:Colour`. Since both `tc:Colour` and `tc:green` are modeled as `skos:Concept` the properties `od:allowedValues` and `od:valueOf` are subproperties of `skos:narrower` and `skos:broader`[1] and used to indicate the relation between attribute type and attribute value type

In the TOP model, properties of classes are modelled only sparsely. For example, a flow can be a *discrete flow*. This is a value of a property of this flow. To model such a relationship, the *property itself*, the *range of values of the property*, and the *possible values* of this range must be modelled.

```
ex:MeiersFlow od:hasFlowType tc:discreteFlow .

od:hasFlowType a rdfs:Property;
  rdfs:domain tc:Flow;
  rdfs:range tc:FlowType .

tc:FlowType a skos:Concept, od:AdditionalConcept ;
  od:allowedValues tc:complexFlow, tc:discreteFlow, tc:continousFlow ;
  skos:prefLabel "Type of flow"@en, "Flussart"@de ;
  skos:altLabel "Art der Strömung"@de .

tc:discreteFlow a skos:Concept, od:AdditionalConcept;
  od:valueOf tc:FlowType ;
  skos:prefLabel "discrete flow"@en, "diskreter Fluss"@de .

...
```

All this is hard to distinguish from a class hierarchy, which should be modelled with the predicate refinement `od:hasSubConcept` of `skos:narrower`. `od:subConceptOf` as the inverse predicate is also used for reasons of clarity; however, it can also be added later automatically.

```
ex:MeiersFlow od:hasStaticFlowComponent tc:Pump .

od:hasStaticFlowComponent a rdfs:Property;
  rdfs:domain tc:Flow;
```

[1] From [2]: The subject of a `skos:broader` statement is the more specific concept involved in the assertion and its object is the more generic one.

```
  rdfs:range tc:StaticFlowComponent .
tc:StaticFlowComponent
  od:hasSubConcept tc:ControlUnit, tc:Receiver, tc:Source, tc:Channel ;
  a skos:Concept, od:AdditionalConcept ;
  skos:prefLabel "static components of the flow"@en,
    "statische Flusskomponenten"@de .
tc:ControlUnit
  od:subConceptOf tc:StaticFlowComponent ;
  od:hasSubConcept tc:Pump, tc:Valve ;
  skos:prefLabel "Control Unit"@en, "Steuerungssystem"@de ;
  skos:altLabel "Management System"@en, "Managementsystem"@de .
tc:Pump
  od:subConceptOf tc:ControlUnit ;
  skos:prefLabel "pump"@en, "Pumpe"@de ;
  a skos:Concept, od:AdditionalConcept .
```

First of all, it would have to be decided whether `tc:Pump` should be included in the set of concepts or whether the term is too specific. This is of course a question of taste.

Modelling via class hierarchies has the advantage that one can use as object of the predicate `od:hasStaticFlowComponent` instances of the entire class hierarchy. But its exact type can then only be determined by a second query to the model. Alternatively, in this example one could have used predicates `od:hasStaticFlowComponent` and `od:hasControlUnit` in order to distinguish the object type already from the predicate identifier.

References

[1] SKOS – The Simple Knowledge Organization System. `https://www.w3.org/TR/skos-reference/`.

[2] SKOS Simple Knowledge Organization System Primer. `https://www.w3.org/TR/2009/NOTE-skos-primer-20090818/`.

[3] The TRIZ Ontology Project (TOP) Онтология ТРИЗ of the TRIZ Developer Summit. `https://triz-summit.ru/Onto_TRIZ/`.

[4] The WUMM Project. `https://wumm-project.github.io/`

[5] The WUMM TOP Companion Project. `https://wumm-project.github.io/Ontology.html`