About the WUMM Modelling Concepts of a TRIZ Ontology

Hans-Gert Gräbe

March 30, 2021

1 Background

In 2019, a group of TRIZ specialists around A.G. Kuryan and M.S. Rubin launched the *TRIZ Developer Summit Ontology Project* (TOP), to achieve a review of the status quo and a more accurate ontological mapping of the TRIZ theory corpus. The work is a natural continuation of earlier efforts by other authors [11, 12] to outline a *TRIZ Body of Knowledge*. While the latter focused on a guide through the literature, TOP is concerned with the identification of essential concepts and essential relationships between these concepts using a modern semantic approach. The status of TOP was presented at the TRIZ Developer Summits in 2019 and 2020 and fixed in two publications [9, 10]. In a webinar series[1] first approaches of a detailed modelling of several sub-areas of TRIZ were presented. The project operates its own website `https://triz-summit.ru/Onto_TRIZ/` on which consolidated results are published.

The main results so far have been a mapping of the continents of the TRIZ world as a *Top Level Ontology* as well as a (still developing) division of that world into *Ontomaps* as specifically defined areas, which are to be modelled in more detail. Moreover, a *thesaurus* of about 500 terms as essential TRIZ concepts has been identified, which are to be defined more precisely. The glossary [15] by V. Souchkov in its version 1.2 serves as basis for this work. In the meantime a first list of 100 terms [18] has been published on the TOP website.

The efforts differ significantly from earlier approaches to the development of a TRIZ ontology [3, 4, 5, 6, 27, 28]. In those earlier works, the focus was rather on the modelling of concrete TRIZ analysis steps with the aim to incorporate the modelling into corresponding tools, e.g. [5], or even on the modelling of processual elements in flow charts, e.g. [4].

The basis for these and the more recent modelling in the TOP project is the OWL ontology. However, the formal descriptions of logical relationships that are possible with it – such as limits for the cardinality of attribute values of a predicate, which are required to implement of a web interface – are expressed in more recent developments of the Semantic Web on the basis of SHACL, the inference possibilities of OWL that go beyond this are hardly used in practice, since OWL-Full leads in sufficient generality to provably undecidable problems, but the modelling restrictions of weaker OWL variants do not meet the requirements of real-world modelling even of *structural* relationships in TRIZ.

Our approach therefore returns to modelling based on RDF and consistently relies on the SKOS ontology as a lightweight framework for modelling structural relationships in conceptual

[1] See `https://wumm-project.github.io/OntologyWebinar` for links to the presentations and an English summary of the talks and discussions.

systems. On this basis we model *structural* aspects and relationships between TRIZ concepts and tools. *Processual* relationships as in [4] or questions of an implementation of web interfaces as in [5] are initially not covered, although especially for the second question comprehensive experience from other application areas with the use of SHACL is available. Such a restriction seems reasonable to us in view of general insights into the development of conceptual systems [20] for a first stage of ontological modelling.

The main disadvantage of the TOP approach so far is the inconsistent use of semantic means. Such means are used in the background and in the internal processes of the TOP team, but even a clear namespace concept for URIs[2], the public availability of the results in an RDF store or at least as files in a relevant format – all this is missing, not to mention a SPARQL endpoint for querying the concepts.

However, such an infrastructure was developed and set up in the context of the WUMM project [22] and used for the representation of actors and activities of a *TRIZ Social Network* `https://wumm-project.github.io/TSN.html` (persons, conference reports, presentations, certificates). The data is publicly available in our github repo `RDFData` at [21] and forms the basis for a prototypical presentation platform [23] that uses simple semantic tools[3] to present different facets of the data. Via a SPARQL endpoint [25] experts can make their own complex queries to the dataset.

This technical basis is the starting point for remodelling parts of the TOP outcome in the course of a *WUMM TRIZ Ontology Companion Project* (WOP) [26]. This 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[4] infrastructure on this basis,

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

In addition to our own modelling (so far of the TRIZ Principles, the TRIZ Inventive Standards and the TRIZ Business Standards), the *Top Level Ontology* and the division into *Ontomaps* are available in this format. The work on a *thesaurus* as well as the presentation of different approaches to a common glossary is actively accompanied. In the WOP approach the differing definitions of different TRIZ schools can coexist more clearly side by side than it is conceptually possible (and is probably not aimed at) in the TOP approach. This aspect, together with the focus on multilinguality, for which individual translation projects can easily be delimited based on the relevant RDF concepts, represent the essential additional contributions of the WOP approach.

The aim of this paper is to explain the basic modelling and semantic assumptions, concepts and settings of the WOP approach in more detail.

[2]URI – Unique Resource Identifier, one of the basic RDF concepts. This string is the *digital identity* of a concept and allows to add independently information about ≪the same thing≫ in a distributed environment.

[3]PHP and bootstrap using the EasyRdf PHP library – the code is publicly available in the github repo `web` at [21] for study and reuse in own platforms.

[4]LOD is the abbreviation for *Linked Open Data*, a world of interlinked data and ≪worlds of concepts≫ steadily growing during the last 15 years. See `https://lod-cloud.net/`.

2 Modelling a TRIZ ontology

2.1 TRIZ and the World of (Technical) Systems

All TRIZ concepts revolve around the central notion of a *system*, its planning, creation, operation, maintenance, further development, etc. Following the widely accepted understanding of that concept in the TRIZ community, TOP defines

> A *system* is a set of elements in relationship and connection with each other, which forms a certain integrity, unity. The need to use the term ≪system≫ arises when it is necessary to emphasize that something is large, complex, not fully immediately understandable, yet whole, unified. In contrast to the notions of ≪set≫ and ≪aggregate≫, the concept of a system emphasises order, integrity, regularities of construction, functioning and development. The notion of system is part of the system and functional approach, and is used in the system operator.

Usually, however, the definition of a system refers to the concept of a *component*, as in Souchkov's glossary [15]:

> *Technical System:* A number of components (material objects) that were consciously combined to a system by establishing specific interactions between the components. A technical system is designed, developed, manufactured, and assigned to perform a controllable main useful function or a number of functions within a particular context. A technical system can include subsystems which can be considered as separate technical systems.
>
> *Component:* A material object (substance, field, or substance-field combination) that constitutes a part of a technical system or its supersystem. A component might represent both a single object and a group of objects.

We thus conclude that a system is essentially a collection of components that interact in a specific way to produce the characteristic functionalities of the system. The subsystems referred to as components provide own functions for this, but the functionalities of the system do not result from a simple addition of the component functions, but as an emergent system property from their interaction. For the modelling of systems, their structural organisation (the ≪machine≫ in the sense of [16]) and their workflow organisation (≪how the machine works≫, ibid.) are equally important. The systemic approach is thus self-similar and fractal; the terms ≪system≫ and ≪component≫ are largely used synonymously depending on the respective *modelling focus*.

In TRIZ, an *engineering problem* is always conceptualised as the design of a new system or the improvement of an existing one. We regard the design of a new system as a special case of further development, since in this case, concepts of a model of the ≪system as it is≫ do exist, how vague they may be.

The delimitation of meaningful systems as modelling units has many facets and points of view, see for example [17, section 8]. In the TRIZ concepts, a certain functional completeness plays a major role in this delimitation, even if a defined throughput of energy, material and information is required for its operation. For a system, its *design* and *operation* have to be

distinguished, as explained in more detail in [7]. This also applies to *components* of a system. In the white-box analysis of a system, its components are considered as working black-boxes, which are characterised in the design dimension by a *specification of their functionality* and in the operation dimension by the *guaranteed specification compliant operation*, provided that the operating conditions (in particular the throughput of material, energy and information required for its operation) are ensured within the system. The description of these operating conditions is part of the specification, which thus consists of an input and an output part (also referred to as import and export interfaces).

The components thus constitute a *world of technical systems* in the sense of the explanations in [7], to which we refer for further details of this conceptualisation.

2.2 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.

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 real-world system*. In the modelling of a real-world system with its *core and cross-cutting concerns* (as known from Software Engineering), the worlds of several conceptual systems often come together. In addition to the methodological dimension of a TRIZ ontology, these are regularly the conceptual world of a technical ontology and possibly other conceptual worlds such as a company-internal compliance etc.
   The ontologies provide 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 (TRIZ) 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.

2.3 The TOP Concept of a System

A central concept in TOP modelling is the distinction between the stages of

(1) the system as it is,
(2) the TRIZ model of the system as it is,
(3) the TRIZ model of the system as required, and
(4) the system as required.

The TOP glossary [18] explains the differences as follows

(1) The *system as is* is a system in its original state before it is analysed and transformed into a new ≪system as is≫.
(2) The *TRIZ model of the system as is* is formed from the ≪system as is≫ by means of various TRIZ models: component-structural and functional models, su-field or ele-field models, description of contradictions or of typical conflict schemes, etc. Depending on the chosen model type, the model will be transformed into the ≪TRIZ model of the system as required≫.
(3) The *TRIZ model of the system as required* is formed from the ≪TRIZ model of the system as is≫ by procedures which correspond to the selected model transformation method (functional, su-field, ele-field, solution of the contradictions in requirements and properties, etc.). The transition is performed along the line

≪System as is≫ → ≪TRIZ model of the system as is≫
→ ≪TRIZ model of the system as required≫ → ≪System as required≫

in accordance with the scheme of a TRIZ Model.
(4) The *system as required* is a system derived from the ≪system as is≫ through a transformation, based on the ≪model of the system as required≫.

It is clear that ≪system≫ here can only mean a *model of the system* in which, in addition to the ontology of the TRIZ methodology, a domain-specific ontology plays a central role, because a system is only accessible in descriptive terms via its model, as the schematisation in Figure 1 also suggests.

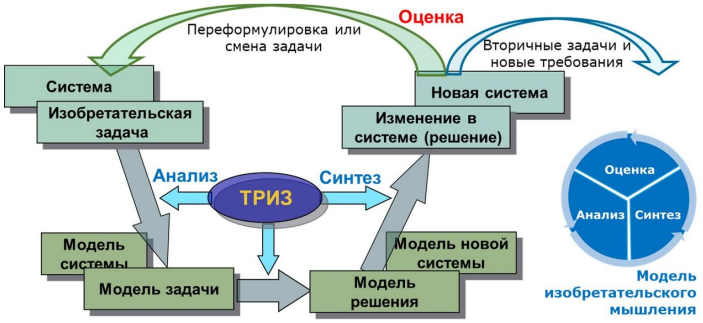

Figure 1. Visualisation of the TOP TRIZ model.
`https://triz-summit.ru/onto_triz/mod`

The ≪TRIZ model of the system as is≫ emerges from this through application of specific-structural TRIZ concepts and instruments. How is this to be understood? Is the (model of the) ≪system as required≫ initially a domain-specific modelling that is to be enriched by an appropriate TRIZ model in this phase (2)? Such an understanding would contradict TRIZ modelling practices, which methodically are to be applied already in the creation of the

domain model, for example with the schema of a *minimal technical system* to be filled in as a template in the domain modelling. In the sense of the hill schema, in phase (2) rather the specific TRIZ structure of the modelling is to be determined and on this basis the domain-specific modelling from phase (1) is to be strengthened in a targeted manner at points to be identified (operative zone and operative time). This *TRIZ model* as a prototypical abstraction of the modelling of the real-world problem at the same time determines the *abstract TRIZ tools* to be used and thus provides the context for the transition to the solution model ≪on the top of the hill≫, which in the end only has to be ≪rolled down≫ to the (model of the) ≪system as required≫. The TRIZ model is thus a *context* for all four phases of real-world modelling. In this way the notion of TRIZ model is also explained in [18]:

> A *TRIZ model* is a schematic notation of a gradual transition from the problem to TRIZ model of the problem, then to TRIZ model of the solution and then to the solution itself; or from the system to TRIZ model of the system, then to TRIZ model of the new system and then to actual change of the system (≪system as required≫). The TRIZ model includes the basic components of inventive thinking: analysis, synthesis, evaluation.

Hence a *TRIZ model* is the common (developing during the phases) abstract TRIZ context of the four model stages described above, including the modelling process itself. However, these four stages all refer to level 1 models of a real-world system; no distinction is made between application of concepts from level 2 of a *TRIZ ontology of tools* (present at level 1 as concept instances) and level 3 of a *TRIZ ontology of methods* (present at level 1 only in a methodological-processual way).

What does this mean for the scope of a TRIZ ontology? The modelling of any system starts with the modelling of the ≪system as it is≫ on the basis of domain-specific concepts. If the modelling is done on the methodological basis of TRIZ principles, the domain-specific system of concepts must be enriched with TRIZ methodological concepts such as MPV, conflicting pairs, operative zone and operative time, etc. The second step requires a special abstraction from domain-specific concepts for the extraction of abstract TRIZ patterns as ≪TRIZ model of the system as it is≫ (TRIZ task model) according to the hill schema. Hence the modelling of the real-world system requires that the domain-specific concepts are compatible with the requirements of TRIZ modelling. The two ontologies present in this modelling – the domain-specific and the TRIZ ontology – have a similar relationship of the specific to the general and thus stand in a relationship of mutual complementation of the modelling languages in the special modelling application.

It is clear, however, that the ≪model of the system as it is≫ (MSI) enriched with elements of a domain-specific ontology, the abstract ≪TRIZ model of the system as it is≫ (TSI) extracted from it, the resulting ≪TRIZ model of the system as required≫ (TRIZ solution model, TSO) and finally the ≪model of the system as required≫ (MSO), which is again enriched in a domain-specific way, call up largely the same language constructs from the point of view of a TRIZ ontology and are thus four instances of the (developing through the four phases) real-world system model, which are related as follows:

- **MSI → TSI:** Consolidation and refinement of TRIZ-relevant concepts in the MSI.
- **TSI → TSO:** Description of an abstract transformation and execution of the parts of the transformation that are possible at this level, i.e. without interaction with the

domain-specific modelling.

- **TSO → MSO:** Detailing of the model, completion and execution of the domain-specific part of the transformation.

For the Level 2 ontology (it answers the question ≪Which TRIZ tools are available and how do they relate to each other?≫), the distinction between these four system models is therefore not relevant. Corresponding language tools are only needed at level 3, when it comes to the terminology of the *application* of the TRIZ methodology itself.

Concerning the balance between the new and the old, as suggested by relevant methodologies for the further development of conceptual systems, we see the need to clearly distinguish between these two levels of ontologisation and limit our ontological modelling to level 2.

3 Basics of the WUMM Ontology Project

3.1 SKOS Basics

The SKOS ontology allows to express concepts and their relations in a lightweight way. The class `skos:Concept` and the predicates `skos:narrower`, `skos:broader` and `skos:related` are used for this purpose. The first two predicates describe hierarchical relationships between concepts[5], the third is used for non-hierarchical relationships.

Relationships between concepts can be of very different structure. Hierarchical relationships, for example, can model (transitive) subconcept relationships in taxonomies as well as whole-part relationships, which are inherently non-transitive when concepts of different qualities are related. Both types of conceptualisation have an intentional as well as an extensional aspect – the new units of meaning, especially their emergent properties, can neither be adequately described by mere enumeration of their subconcepts nor by the ≪legitimate interpretation of sense≫ of the purposes of their constitution in the sense of [2]. In the SKOS primer [14] these modelling aspects are described in more detail, especially the modelling of class-instance and whole-part relationships. We follow the recommendation in [14, Sect 4.7] and introduce subpredicates of the generic SKOS predicates listed above for different modelling contexts. More detailed modelling rules for such contexts are described and discussed below.

Since this project is about modelling a unified space of TRIZ concepts, further SKOS aggregation concepts such as `skos:ConceptScheme`, `skos:Collection`, `skos:OrderedCollection` etc. are not used. The aggregation of different concepts in collections (assignment to TRIZ generations [10, Table 1] or in concept classes `Basic`, `Model`, `Rule` and `TermGroup` [10, Fig. 4]) is realised via special predicates.

We use the SKOS ontology [13] with the concepts (K)

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

[5] From the SKOS Primer [14]: ≪The subject of a `skos:broader` statement is the more specific concept involved in the assertion and its object is the more generic one≫, i.e. `A skos:broader B` expresses that A is a subconcept of B. `skos:narrower` is the inverse property to `skos:broader`.

SKOS provides an initial descriptive framework for conceptualisations. For the meaning of the individual concepts, we refer to [13] and the explanations below.

3.2 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 TOP sources are identified solely by their labels. The OSA platform[6] 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.

One of the first decisions that must be made for the allocation of URIs is the *definition of namespaces* that correspond to the different modelling contexts. Since an *ontology modelling* basically has the purpose of being applied in *modellings of real-world systems*, at least these two modelling contexts have to be distinguished. The modelling context of a (prototypical) real-world system will usually only play a role in examples in which the effect of ontology modelling decisions is practically demonstrated. At the level of ontology modelling, we further distinguish between the parts of the concepts that are largely uncontroversial[7] and the parts of the concept for which special conceptual approaches have been developed within the WUMM Ontology Project (WOP). For these different abstraction layers we use the following namespaces:

- `ex:` – the namespace of a special system to be modelled.
- `tc:` – the namespace of the TRIZ concepts (RDF subjects).
- `od:` – the namespace of WUMM's own concepts (RDF predicates, general concepts).

During the computer based transformation of the datasets into a valid RDF format, a first suggestion for URIs in the namespace `tc:` was automatically generated and then further consolidated in several steps. An essential task still to be done is to finish this final consolidation of URIs, i.e. to merge URIs generated from different sources that refer to the same concept.

3.3 Provenance of Explanations

Another 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 address the ≪worlds≫ of the individual TRIZ schools separately. The same applies to the use of provenance-dependent subclasses of `skos:Concept`.

Such notational variations are for example

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

[6]The OSA platform is used as an TOP internal ontology editor, see `https://wumm-project.github.io/TOP` for more information about the platform, its odds and evens.

[7]These are mainly the *concepts* to be included in a glossary. We assign URIs of a `skos:Concept` to them and model their names as `skos:prefLabel`.

etc. Here `GSAThesaurus` stands for the thesaurus published on the Altshuller website [1], `VDIGlossary` for the VDI glossary [19] and `SouchkovDefinition` for the glossary [15] by V. Souchkov. All these data were available or provided to the WUMM project in a machine-readable format, transformed by us into suitable RDF formats and are available as open source both as files in the github repo `RDFData` at [21] and in our RDF Store [24]. See the RDF data itself, which can also be queried via our SPARQL endpoint [25].

This can be used to build a *combined glossary* where definitions from different TRIZ schools of the same concept co-exist. This is implemented prototypically[8] im such a way, that for each concept represented by a URI, a link displays all RDF triples in which this concept occurs as a subject or object. Further links in this representation can be used to navigate in the entire RDF graph (more precisely: in its respective connected component).

3.4 Modelling Systems and TOP TRIZ Models

Since TOP system models in (1) and (4) come with additional modelling information based on a great variety of domain-specific ontologies the TRIZ ontology is an add-on only and (domain-specific) ontology integration engineering approaches are required anyway. It is thus justified to concentrate solely on the concerns of modelling the TRIZ-relevant aspects in all four modelling stages.

In the TOP approach, the distinction of these modelling stages is consistently introduced for all system-relevant concepts at the level of subconcepts. However, since the transition from one modelling stage to the next is carried out jointly for all concepts related to the system, it is sufficient to assign the modelling stage to the respective model of the system as a property. Separate sub-concepts at all levels, as introduced in TOP modelling, are not necessary, since the stage can be inferred via the model of the system in which a concept is ≪built in≫. Hence the WOP approach takes a different route here and models this connection as a predicate `od:belongsTo` with value from the `tc:StageValue` range

```
tc:SystemAsIs, tc:SystemAsRequired, tc:ModelAsIs, tc:ModelAsRequired
```

to assign to a concept *instance* in a real-world modelling one of the modelling stages as discussed above. This also makes it easy to extend the `rdfs:range` of this property if, for example, stages of earlier or later system versions are to be included in the modelling of a special system in the context of an application of the system operator.

This significantly reduces the number of concepts to be distinguished, which is also indicated by reasons of homogeneity, since the different stages of this system model must be structurally similar in order to be able to capture structural continuities within its development, and thus must be modelled by a single uniform concept.

[8] Seee `http://wumm.uni-leipzig.de/ontology.php`.

4 Typical modelling situations

4.1 Class-instance relation

In OO programming classes are usually extensionally conceptualised as a concept of member functions and attributes that are common to all instances of that class. We do not consider here the special possibility to define also static attributes and functions for a class. In this sense the class concept generalized the concept of its members. A special way to conceptualize classes with a finite number of instances are *morphological tables*.

Within the WOP approach this relation is modelled using the predicates `od:allowedValues` (a subproperty of `skos:narrower`) and `od:valueOf` (a subproperty of `skos:broader`). The class concept belongs to the WOP category `od:PropertyDomain`. No distinction is made between attribute (left column of a morphological table) and the attribute value range (set of values in the right column of a morphological table).

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

```
ex:MeiersCar a ex:Car; od:hasColour tc:green .

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

tc:Colour a skos:Concept, od:AdditionalConcept ;
  skos:prefLabel "Colour"@en, "Farbe"@de ;
  od:WOPCategory od:PropertyDomain ;
  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 .

...
```

4.2 Class hierarchy relation

In the WOP approach class hierarchies are modelled with the predicate `od:hasSubConcept` that refines `skos:narrower` and `od:subConceptOf` as its inverse predicate.

Example: A flow has several static components (source, channel, receiver, control unit). A flow itself is a component of a system and hence belongs to the WOP category `od:Component` as its subcomponents do.

```
ex:MeiersFlow od:hasStaticFlowComponent ex:SpecialPumpX32 .
ex:SpecialPumpX32 a tc:Pump .

od:hasStaticFlowComponent a rdfs:Property;
  rdfs:domain tc:Flow;
  rdfs:range tc:StaticFlowComponent .

tc:StaticFlowComponent
  od:hasSubConcept tc:ControlUnit, tc:Receiver, tc:Source, tc:Channel ;
```

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

4.3 Classification of transformations

The concept of *transformation* is one of the central TRIZ concepts and means that a real-world system or parts of it is transformed from a ≪system as it is≫ into a ≪system as required≫ according to predefined principles oriented at an *ideal end result* (IER). As explained above in more detail, the expected outcome of the planned transformation (of the ≪system as required≫) is to be distinguished from the real outcome of the execution of the transformation (of the ≪system as it became≫), which gives rise to the execution of further transformations, iterative transformation approaches and more elaborated transformation concepts such as versions of system generations. The concept of transfomation is central to such further elaborations, which, however, are beyond the scope of our current ontological modelling.

A transformation is associated with a change of state of the system under investigation and thus occurs at level 1 – the modelling of the real-world system – as an RDF predicate. The transformation pattern applied here are concepts on level 2 of the modelling – the ontological modelling – and describe the RDF predicate more precisely. In this ontology modelling, the RDF predicate appears as RDF subject to which the corresponding transformation patterns are assigned.

This is demonstrated by an example from [8, Fig. 4.20], in which the evolutionary lines of boats are described. Here the transformation from a tree trunk to a row boat is explained.

```
ex:TreeTrunk ex:addPaddles ex:Rowboat .
ex:addPaddles a rdf:Property, skos:Concept;
  od:usesPattern tc:NewPrinciplePattern, tc:IncreasingControllabilityPattern;
  skos:note """The non-controllable floating tree trunk on the river is
    provided with oars with which the boat can be propelled by muscle power
    and also steered"""@en .
```

The RDF statements are part of a (level 1) description of the real-world evolution process of boats and use (in the second sentence) the (level 2) TRIZ development pattern

`tc:NewPrinciplePattern` and `tc:IncreasingControllabilityPattern` that are bound to the level 1 predicate `ex:addPaddles` (being a concept not of TRIZ but of the special evolution process) by the level 3 predicate `od:usesPattern` (a concept of TRIZ methodology).

4.4 Concept Categories

The WOP approach distinguishes (at the moment) the concept categories `od:Component` for describing the *structural* organisation of a system and its parts, and `od:PropertyDomain` for describing the structure of *property domains* of a system and its parts. Both are used as values of the predicate `od:WOPCategory`, following a modelling approach of categories of V. Souchkov in his glossary.

References

[1] Altshuller Web Site. Basic TRIZ terms.
`https://www.altshuller.ru/thesaur/thesaur.asp`

[2] Peter L. Berger, Thomas Luckmann. The Social Construction of Reality: A Treatise in the Sociology of Knowledge. Anchor Books, 1966. ISBN 978-0-385-05898-8.

[3] Alexis Bultey, F. de Bertrand de Beuvron, François Rousselot (2007). A substance-field ontology to support the TRIZ thinking approach. International Journal of Computer Applications in Technology 30 (1), pp. 113-124.
`https://doi.org/10.1504/IJCAT.2007.015702`.

[4] Alexis Bultey, Wei Yan, Cécilia Zanni (2015). A Proposal of a Systematic and Consistent Substance-field Analysis. Procedia Engineering 131, pp. 701-710.
`https://doi.org/10.1016/j.proeng.2015.12.357`.

[5] Denis Cavallucci, François Rousselot, Cécilia Zanni (2011). An ontology for TRIZ. Proc. TRIZ Future Conference 2009. Procedia Engineering 9, 251–260.
`https://doi.org/10.1016/j.proeng.2011.03.116`.

[6] Sébastien Dubois, Philippe Lutz, François Rousselot, Gérard Vieux (2007). A model for problem representation at various generic levels to assist inventive design. International Journal of Computer Applications in Technology 30 (1), pp. 105-112.
`https://doi.org/10.1504/IJCAT.2007.015701`.

[7] Hans-Gert Gräbe. Human and their technical systems. In Proceedings of the TRIZ Future Conference 2020, p. 399-410. An enlarged German version is available as `http://dx.doi.org/10.14625/graebe_20200519`.

[8] Karl Koltze, Valeri Souchkov (2017). Systematische Innovationsmethoden (in German). Hanser, Munich. ISBN 978-3-446-45127-8.

[9] Andrej Kuryan, Valeri Souchkov, Dmitri Kucharavy. Towards ontology of TRIZ. Proceedings of TRIZ Developers Summit 2019 Conference, Minsk, 2019.
`https://wumm-project.github.io/Ontology.html`

[10] Andrej Kuryan, Michail Rubin, Nikolaj Shchedrin, Olga Eckardt, Natalja Rubina. TRIZ Ontology. Current State and Perspectives. TDS 2020 (in Russian). https://wumm-project.github.io/Ontology.html

[11] Simon Litvin, Vladimir Petrov, Michail Rubin (2007). TRIZ Body of Knowledge. https://triz-summit.ru/en/203941.

[12] Simon Litvin, Vladimir Petrov, Michail Rubin, Victor Fey (2012). TRIZ Body of Knowledge. https://matriz.org/wp-content/uploads/2012/07/TRIZ-Body-of-Knowledge-final.pdf

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

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

[15] Valeri Souchkov. Glossary of TRIZ and TRIZ-related terms, version 1.2. The International TRIZ Association, MATRIZ 2018. https://matriz.org/wp-content/uploads/2016/11/TRIZGlossaryVersion1_2.pdf

[16] Nikolai Shpakovski et al. The TRIZ Trainer. https://triztrainer.ru, https://triz-trainer.com.

[17] Clemens Szyperski (2002). Component Software: Beyond Object-Oriented Programming. ISBN: 978-0-321-75302-1.

[18] TRIZ 100 Glossary. A short glossary of key TRIZ concepts and terms (in Russian). https://triz-summit.ru/onto_triz/100/.

[19] VDI-Norm 4521 Blatt 1. Erfinderisches Problemlösen mit TRIZ – Grundlagen und Begriffe (Inventive problem solving with TRIZ – Basics and terminology). April 2016.

[20] Lev S. Vygotsky (1934). Denken und Sprechen. Akademie-Verlag, Berlin 1964.

[21] The WUMM github repositories. https://github.com/wumm-project.

[22] The github pages of the WUMM Project. https://wumm-project.github.io/.

[23] The web demonstration pages of the WUMM Project. http://wumm.uni-leipzig.de.

[24] The RDF store of the WUMM Project. http://wumm.uni-leipzig.de/rdf.

[25] The SPARQL endpoint of the WUMM Project. http://wumm.uni-leipzig.de:8891/sparql.

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

[27] Cécilia Zanni-Merk, François Rousselot, Denis Cavallucci (2009). An Ontological Basis for Inventive Design. Computers in Industry, 60 (8), pp. 563-574.

[28] Cécilia Zanni-Merk, François de Bertrand de Beuvron, François Rousselot, Wei Yan (2013). A formal ontology for a generalized inventive design methodology. Applied Ontology, 8 (4), pp. 231-273. https://doi.org/10.3233/AO-140128.