Upper ontologies and design patterns

Miroslav Blaško

miroslav.blasko@fel.cvut.cz

November 12, 2017

Outline

Motivation – reuse of ontological resources

- Types of ontologies:
  - top-level (upper) ontologies
  - domain ontologies and task ontologies
  - application ontologies
- Ways to reuse ontological resources:
  - ontologies as wholes
  - syntactic/semantic ontology modules
  - ontology design patterns
  - ontology statements

Ontology design patterns

Basics

Why should we use ontology design patterns ?

- It is hard to extract *only useful pieces* of comprehensive higher level ontologies (e.g. foundational ontologies)
- There is need for small ontologies to address each design issue separately
- The ontology should be accompanied with explicit documentation of its design rationales and best reengineering practices
- Therefore, in analogy to software design patterns there are **ontology design patterns**

Ontology design pattern catalogues

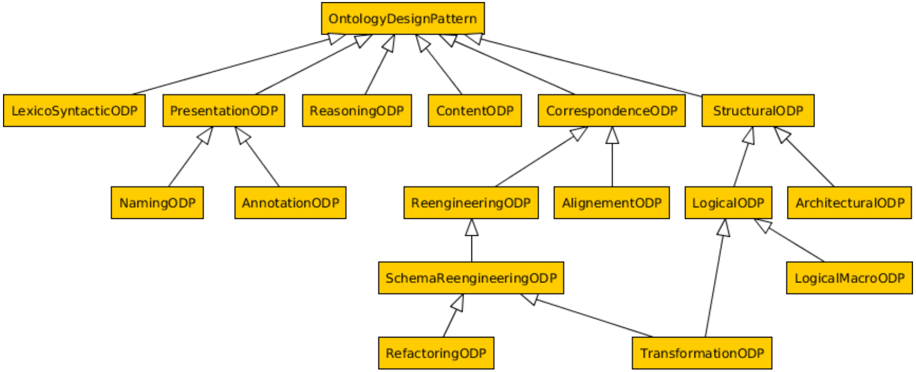
Overview of ontology design pattern catalogues

Most known public ODP catalogues are :

- **ODPs from W3C Semantic Web Best Practices and Deployment Working Group** – contains 4 pattens i.e. n-ary relations, classes as property values, value partitions/sets, simple part-whole relations.
  (`http://www.w3.org/2001/sw/BestPractices`)
- **ODPs from the University of Manchester** – contains 17 patterns devided into groups *extension ODPs* (solutions to bypass the limitations of OWL such as n-ary relations), *good practice ODPs* (making robust and cleaner design e.g. value partitions), *domain modelling ODPs* (solutions for concrete modeling problems in biology). (`http://www.gong.manchester.ac.uk/odp/html`)
- **ODPs from ontologydesignpatterns.org** – contains over 100 patterns categorized into 6 groups of patterns hosted on Semantic Web portal dedicated to ODPs providing review process for creation of certified patterns. (`http://ontologydesignpatterns.org`)

Types of ontology design patterns

Classification of ODPs (1)



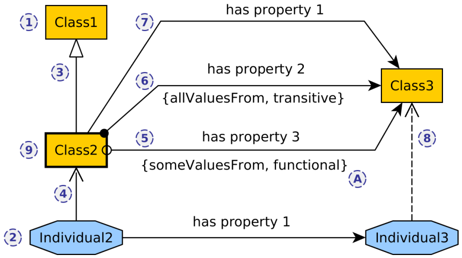
Classification of ODPs according to ontologydesignpatterns.org portal (ODP portal)

Classification of ODPs (2)

- **Content ODP** – represents domain-specific pattern
- **Structural ODP** – is structure to solve architectural and logical issues of OWL ontologies
- **Correspondence ODP** – is used for reengineering and mappings
- **Reasoning ODP** – is typical reasoning procedure
- **Presentation ODP** – relates to usability of ontology from user perspective
- **Lexico-Syntactic ODP** – is linguistic structure/schema that allow to generalize and extract some conclusions about the meaning they express

Selected ontology design patterns

Diagramming conventions within selected ODPs



The figure shows diagramming conventions that will be used in subsequent slides for selected ODPs. *Squares* (1,9) – classes; *octagons* (2) – individuals; *closed hollow arrows* (3) – `rdfs:subClassOf` or `rdfs:subPropertyOf` relations; *opened arrows* (4) – `rdf:type` relations; *semi-closed solid arrows* (5,6,7) having origin of the arrow decorated by: a) *hollow blob* (5) – existential restrictions of the class at the origin of the arrow, b) *solid blob* (6) – universal restriction of the class at the origin of the arrow, c) *no decoration* (7) – domain and range axioms of the property if used with classes, facts if used with individuals; *solid/dashed arrows* (3,4,5,6,7)/(8) – asserted/inferred axioms, respectively; *normal/bold edges of a square* (1)/(9) – the classes represented by the square are defined partially/completely with restrictions and other relevant axioms defined in the figure, respectively; *texts within* {} *brackets* (A) – additional information about restriction or property represented by the arrow, someValuesFrom/allValuesFrom information is already represented with arrow having solid/hollow decoration of the origin, thus may be omitted.

N-ary relations – general patterns

N-ary relations ODP [**Noy:06:DNR**] is logical ODPs that solves issue of representing n-ary relations in OWL which has native support only for binary relations.

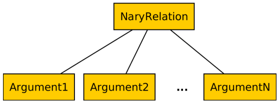


Figure: Pattern 1

- most common representation of n-ary relations
- possible restrictions per argument (e.g. type for each argument, cardinality of each argument type)
- possibility to define required/optional arguments of the relation
- determining owner of relations by direction of the object properties

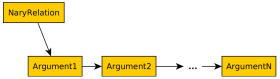
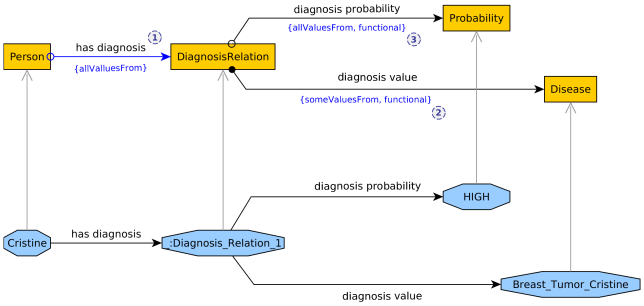


Figure: Pattern 2

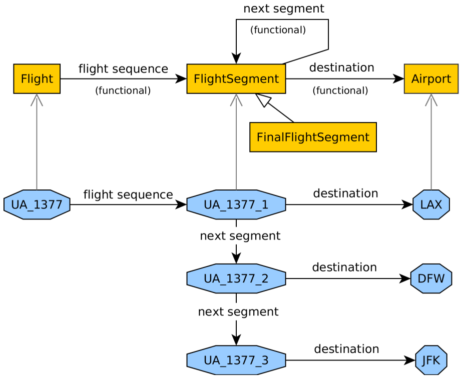
- ordering of dynamic number of arguments
- argument types are content specific instead of generic ones as it is in case of generic list ODP

N-ary relations – pattern 1 example



The figure demonstrate use of *nary-relations ODP pattern 1* for representation of ternary relation – medical diagnosis of disease (expressed by class `DiagnosisRelation`). The ownership of the relation is captured by direction of `has_diagnosis` (1). Each diagnosis is obliged to have some diagnosis value (2), while the diagnosis probability is understood as additional parameter of the relation which is only obliged to have correct type (3) if the value exists. Similarly to diagnosis probability, `Person` is not obliged to have some diagnosis (1).

N-ary relations – pattern 2 example



The figure demonstrate use of *nary-relations ODP pattern 2* for representation of n-ary relation that have dynamic number of parameters where ordering of the parameters matters. It represents flight as ordered sequence of flight segments that point to airport destinations.

Value partitions and value sets – general patterns

Value partition and value set ODP [**Rector:05:RSV**] is able to represent a *feature* of some entity (sometimes also referred as “quality”, “attribute”, “characteristic”, or “modifier” of the entity). There are two ways basic ways to represent the feature:

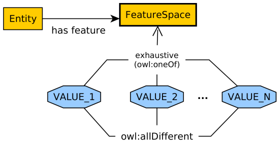


Figure: Pattern 1

- *values as sets of individuals*
- no possibility of further sub-partitioning
- no alternative partitioning of same feature space
- straightforward with database matching

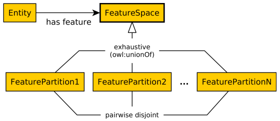
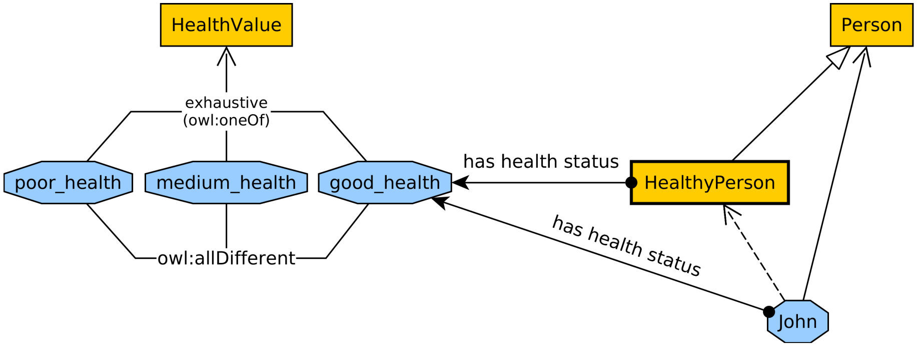


Figure: Pattern 2

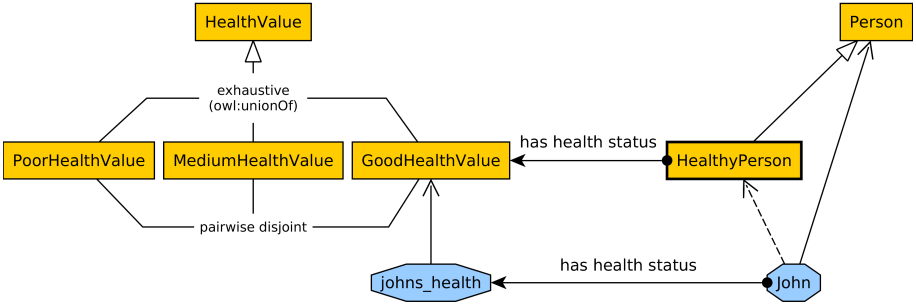
- *values as subclasses partitioning a “feature”*
- possible sub-partitioning and alternative partitioning
- some people consider it less intuitive

Value partitions and value sets – pattern 1 example



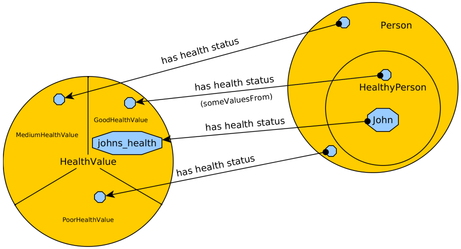
The figure represents feature "health status of a person" by using feature space `HealthValue` as set of concrete values `poor_health`, `medium_health`, `good_health`.

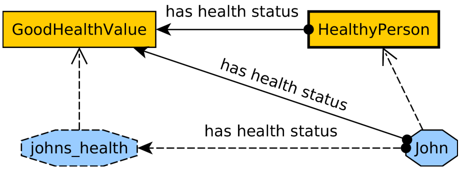
Value partitions and value sets – pattern 2 example



The figure represents feature “health status of a person” by partitioning feature space `HealthValue` into sub-partitions `PoorHealthValue`, `MediumHealthValue`, `GoodHealthValue`.

Value partitions and value sets – pattern 2 example

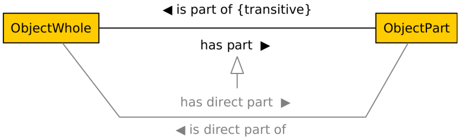




The figure on the left depict previous example in adapted Venn diagram as an alternative diagrammatic format to show partitioning more explicitly. The right part of the figure shows alternative representation of John's health status which is not expressed explicitly but inferred from other axioms

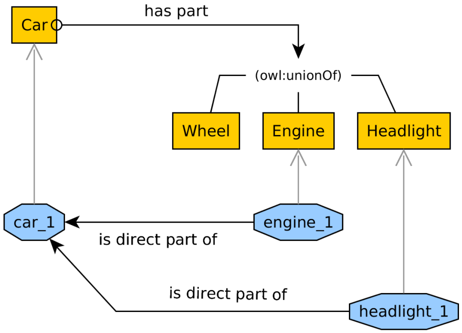
Part-whole relations – general pattern

Part-whole relation ODP [**Rector:05:partwhole**] provide us way to represent objects called wholes and their parts.



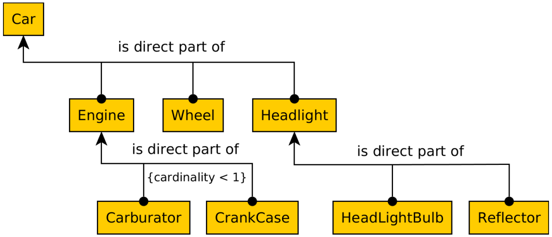
The figure depicts general schema for part-whole relations.

Part-whole relations – inventory of parts example



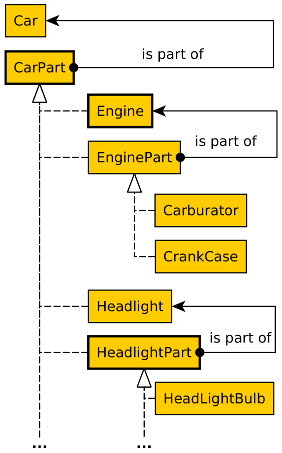
The figure shows how to represent inventory of parts (i.e. parts of concrete objects).

Part-whole relations – hierarchy of parts example



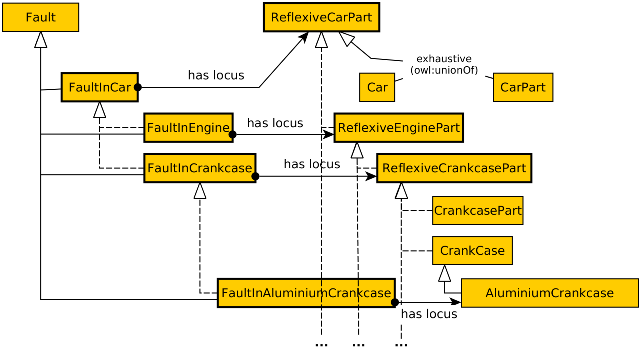
The figure shows how to represent hierarchy of hypothetical parts of wholes.

Part-whole relations – classes for parts example



The figure shows how to represent classes for parts, so the correct hierarchy of parts is inferred.

Part-whole relations – faults in parts example



The figure shows how to represent faults in parts using `has_locus` property.

Upper ontologies

Basics

What are upper ontologies ?

- **Upper ontologies** (sometimes also called *top-level* or *foundational* ontologies) describe very general concepts that are independent of particular problem or domain.
- They provide categories of kinds of things and relations that can provide a basic structure for “lower-level“ ontologies such as domain ontology.

Why should we use upper ontologies ?

- Pros:
  - ”top-down approach“ and modelling guidance for ontology development
  - basic categories and relations that we don’t need to reinvent again
  - interoperability among ontologies
- Cons:
  - a lot of effort needed to understand
  - too abstract

Basic ontological commitments

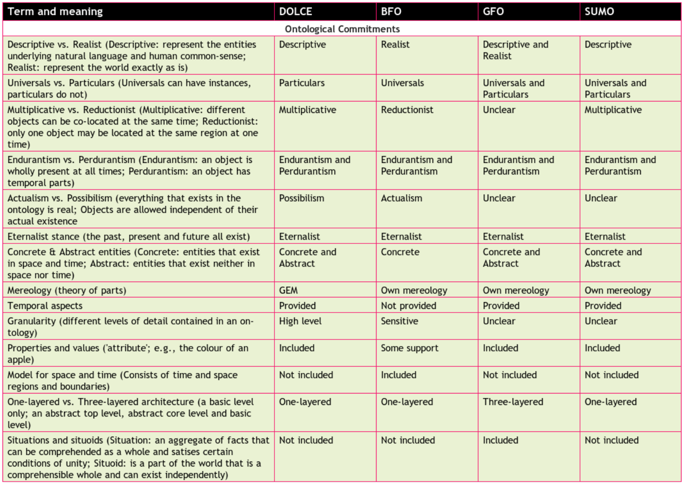
- Universals vs. Particulars – Universals can have instances, while Particulars don't
- Descriptive vs. Realist – represent world using natural language and common sense vs. represent it as is
- Multiplicative vs. Reductionist – different objects can be co-located at the same time vs. only one object may be located at the same region at one time
- Endurantism vs. Perdurantism – an object is wholly present at all times vs. an object has temporal parts
- Actualism vs. Possibilism – everything that exists in the ontology is real vs. objects are allowed independent of their actual existence
- Concrete & Abstract entities – entities that exist in space and time & entities that exist neither in space nor time

Overview of upper ontologies

Existing upper ontologies

- UFO (Unified Foundational Ontology)
- BFO (Basic Formal Ontology)
- DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering)
- SUMO (Suggested Upper Merged Ontology)
- YOMATO (Yet Another More Advanced Top-level Ontology)
- GFO (General Formal Ontology)
- PROTON (PROTo ONtology)
- Cyc
- ?WordNet

Comparison of ontological commitments



Term and meaning	DOLCE	BFO	GFO	SUMO
Ontological Commitments				
Descriptive vs. Realist (Descriptive: represent the entities underlying natural language and human common-sense; Realist: represent the world exactly as is)	Descriptive	Realist	Descriptive and Realist	Descriptive
Universals vs. Particulars (Universals can have instances, particulars do not)	Particulars	Universals	Universals and Particulars	Universals and Particulars
Multiplicative vs. Reductionist (Multiplicative: different objects can be co-located at the same time; Reductionist: only one object may be located at the same region at one time)	Multiplicative	Reductionist	Unclear	Multiplicative
Endurantism vs. Perdurantism (Endurantism: an object is wholly present at all times; Perdurantism: an object has temporal parts)	Endurantism and Perdurantism	Endurantism and Perdurantism	Endurantism and Perdurantism	Endurantism and Perdurantism
Actualism vs. Possibilism (everything that exists in the ontology is real; Objects are allowed independent of their actual existence	Possibilism	Actualism	Unclear	Unclear
Eternalist stance (the past, present and future all exist)	Eternalist	Eternalist	Eternalist	Eternalist
Concrete & Abstract entities (Concrete: entities that exist in space and time; Abstract: entities that exist neither in space nor time)	Concrete and Abstract	Concrete	Concrete and Abstract	Concrete and Abstract
Mereology (theory of parts)	GEM	Own mereology	Own mereology	Own mereology
Temporal aspects	Provided	Not provided	Provided	Provided
Granularity (different levels of detail contained in an ontology)	High level	Sensitive	Unclear	Unclear
Properties and values ('attribute'; e.g., the colour of an apple)	Included	Some support	Included	Included
Model for space and time (Consists of time and space regions and boundaries)	Not included	Included	Not included	Not included
One-layered vs. Three-layered architecture (a basic level only; an abstract top level, abstract core level and basic level)	One-layered	One-layered	Three-layered	One-layered
Situations and situoids (Situation: an aggregate of facts that can be comprehended as a whole and satises certain conditions of unity; Situoid: is a part of the world that is a comprehensible whole and can exist independently)	Not included	Not included	Included	Not included

Comparison of ontological commitments within selected upper ontologies taken from

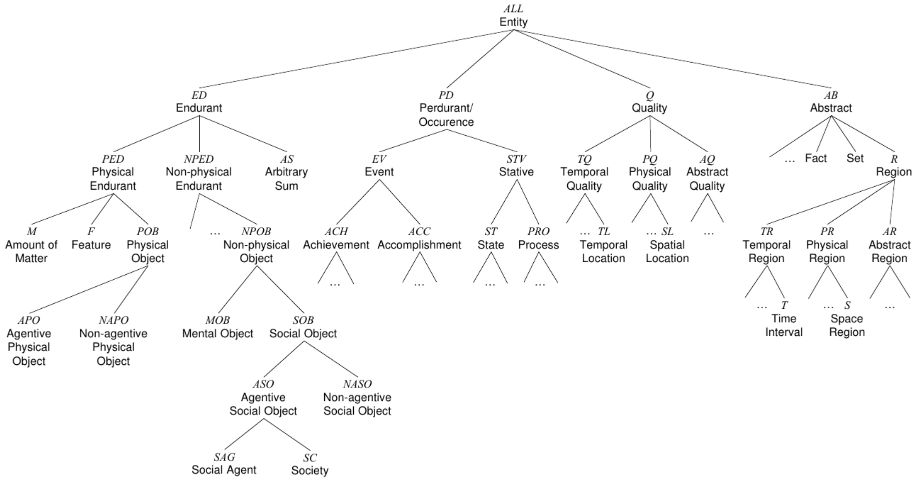
`http://www.thezfiles.co.za/ROMULUS/ontologicalCommitments.html`

DOLCE overview

- **D**escriptive **O**ntology for **L**inguistic and **C**ognitive **E**ngineering[1]
- developed by researchers from the Laboratory of Applied Ontology, headed by N. Guarino
- first module of the WonderWeb Foundational Ontologies Library
- ontology of particulars, multiplicative, possibilism
- strong cognitive/linguistic bias – descriptive attitude with categories mirroring cognition, common sense, and the lexical structure of natural language

[1]Home page – `http://www.loa.istc.cnr.it/old/DOLCE.html`, online term search – `https://www.w3.org/2001/sw/BestPractices/WNET/DLP3941_daml.html`

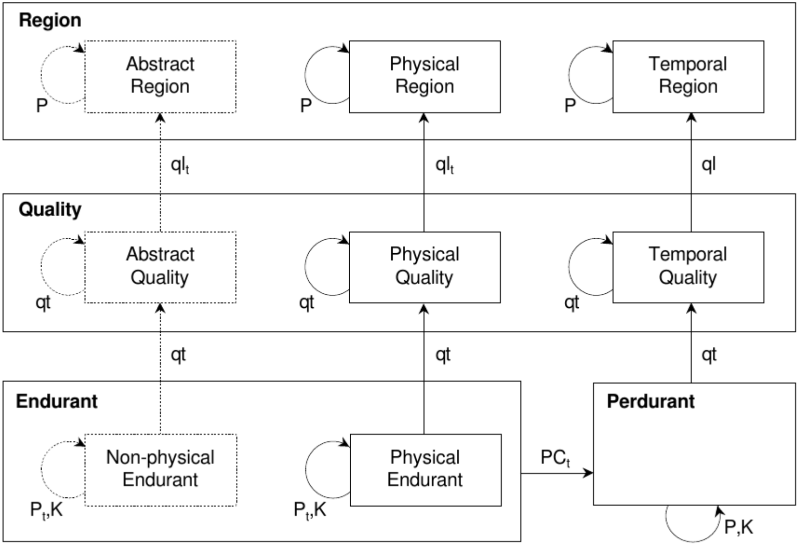
DOLCE's taxonomy of basic categories



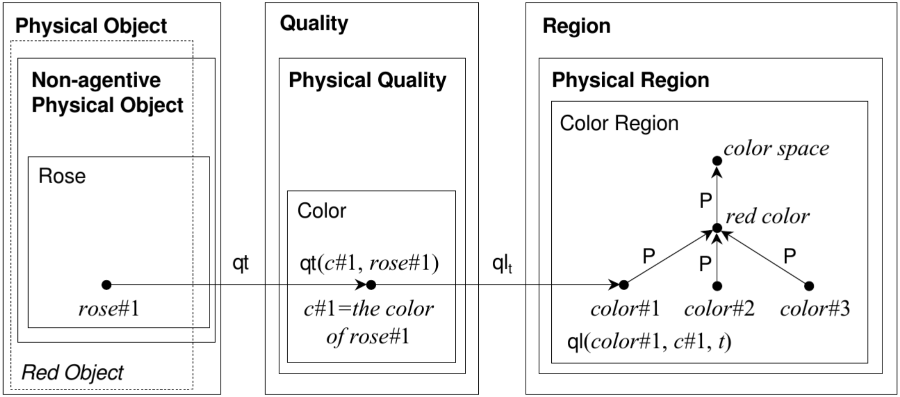
DOLCE basic relations

- parthood (immediate and temporary)
- constitution
- participation
- representation
- specific/generic constant dependence
- inherence (between a quality and its host)
- quale (immediate and temporary)

DOLCE's primitive relations between basic categories



DOLCE's relations about qualities

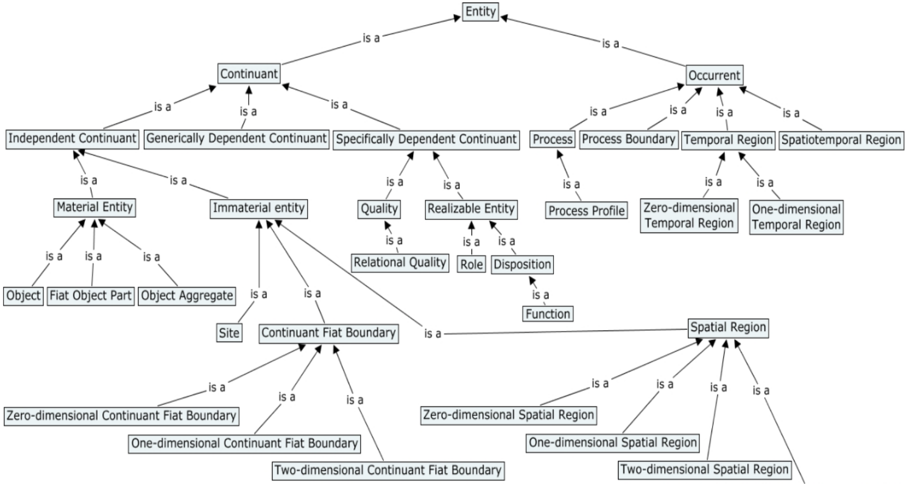


BFO overview

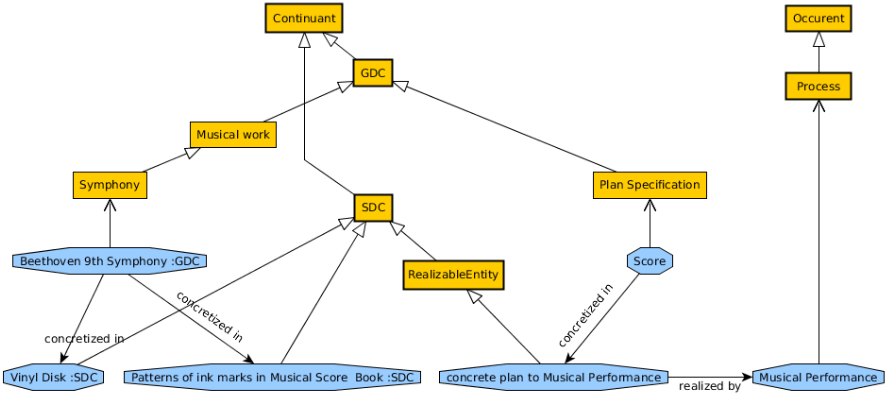
- **B**asic **F**ormal **O**ntology[2]
- developed in Saarland University mainly by B.Smith, P.Grenon
- designed for use in supporting information retrieval, analysis and integration in scientific and other domains
- realistic and reductionist view of the word, actualism
- limited granularity
- contains both SNAP (endurants) and SPAN (perdurants) sub-ontologies

[2] `http://ifomis.uni-saarland.de/bfo/`

BFO's taxonomy of basic categories


Entity
is a
Continuant
Occurrent
Independent Continuant
Generically Dependent Continuant
Specifically Dependent Continuant
Process
Process Boundary
Temporal Region
Spatiotemporal Region
Material Entity
Immaterial entity
Quality
Realizable Entity
Process Profile
Zero-dimensional Temporal Region
One-dimensional Temporal Region
Relational Quality
Role
Disposition
Function
Object
Fiat Object Part
Object Aggregate
Site
Continuant Fiat Boundary
Spatial Region
Zero-dimensional Continuant Fiat Boundary
One-dimensional Continuant Fiat Boundary
Two-dimensional Continuant Fiat Boundary
Zero-dimensional Spatial Region
One-dimensional Spatial Region
Two-dimensional Spatial Region
Three-dimensional Spatial Region

BFO’s realizable entity example



Other interesting upper ontology resources

- ONSET: the foundational ONtology Selection and Explanation Tool – `http://www.meteck.org/files/onset`
- Ontology browser integrated mainly through BFO – `http://www.ontobee.org`
- SUMO Concept hierarchy and search – `http://virtual.cvut.cz/kifb/en/toc/root.html`
- YOMATO ontology description – `http://download.hozo.jp/onto_library/YAMATO101216.pdf`
- UFO related community portal – `https://ontouml.org`