0.1 Upper ontologies and ontology matching

0.1.1 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

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

Comparison of ontological commitments

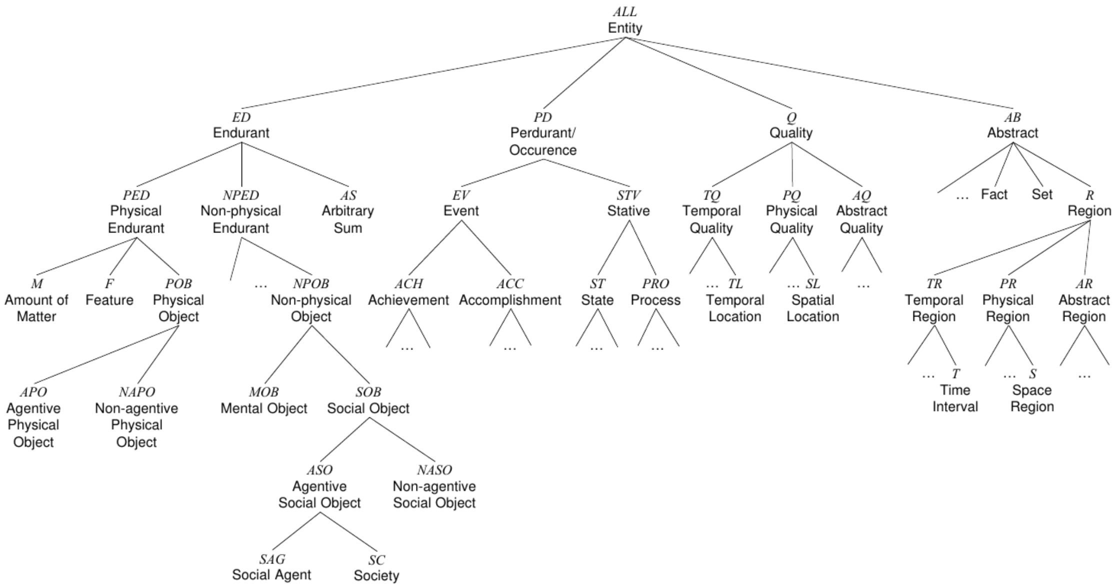
Term and meaning	DOLCE	BFO	GFO	SUMO
Ontological Commitments				
Descriptive vs. Realist (Descriptive: represent the entities underlying natural language and human common-sense; Realist: represent the world exactly as is)	Descriptive	Realist	Descriptive and Realist	Descriptive
Universals vs. Particulars (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
- 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

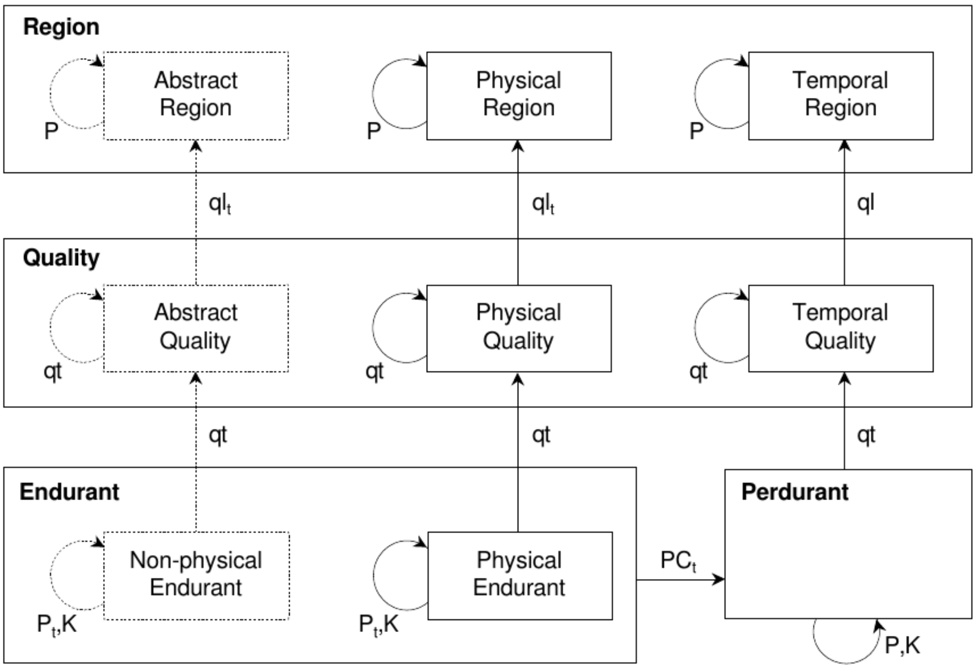
DOLCE's taxonomy of basic categories



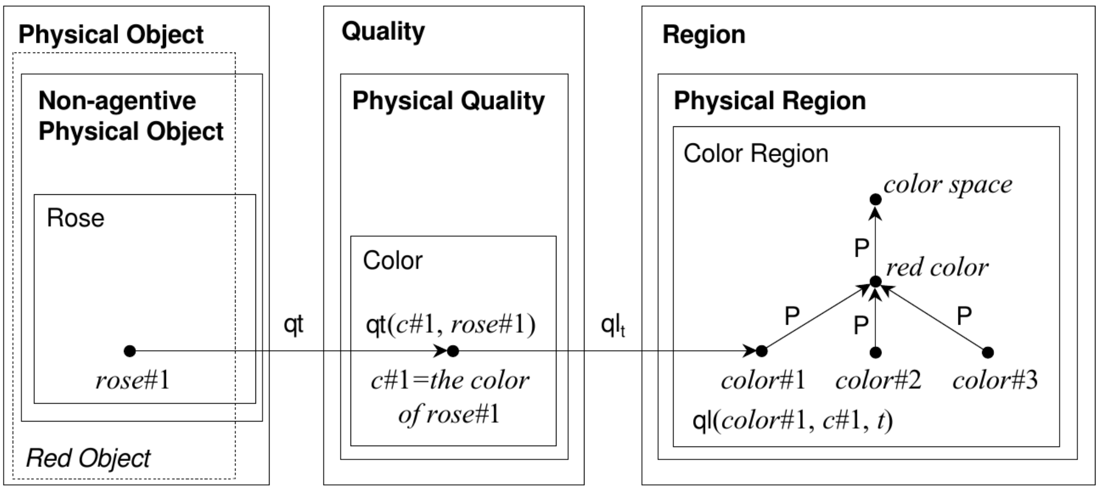
DOLCE basic relations

- 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

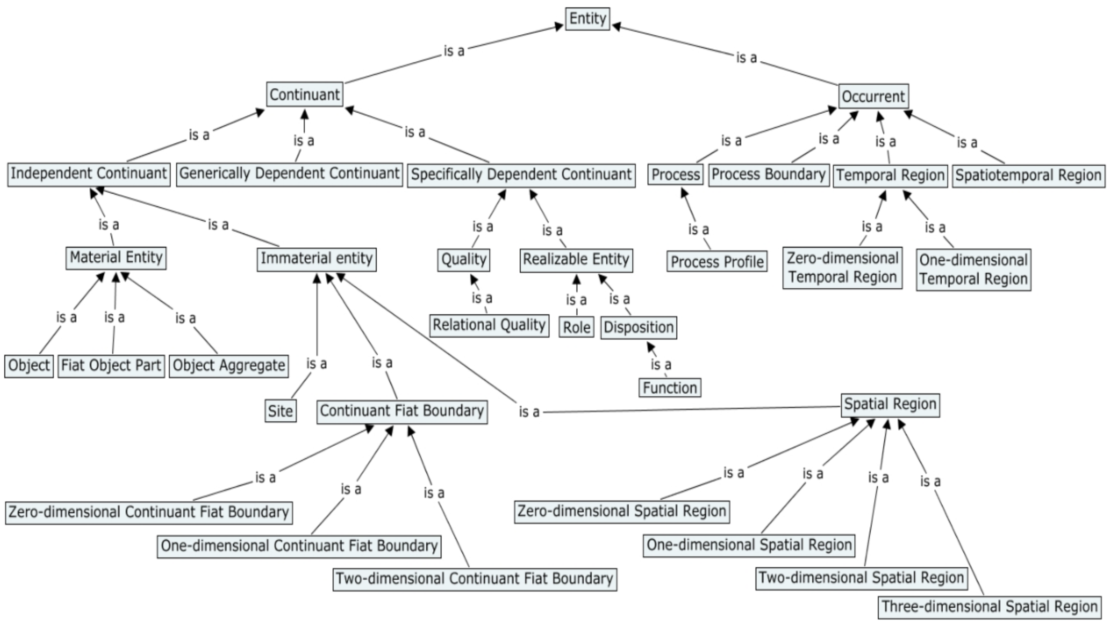


BFO overview

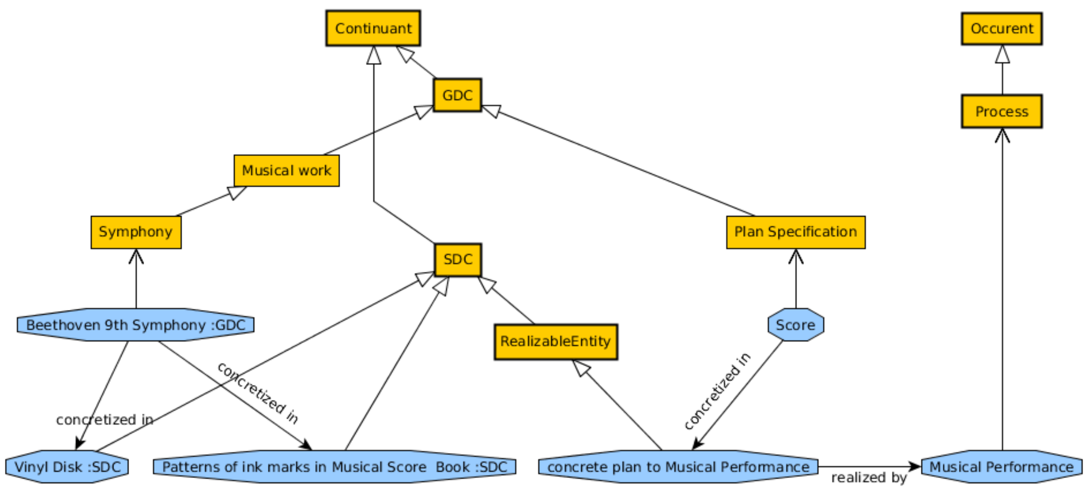
- **B**asic **F**ormal **O**ntology
- developed in Saarland University mainly by B.Smith, P.Grenon

- 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

BFO's taxonomy of basic categories



BFO's realizable entity example

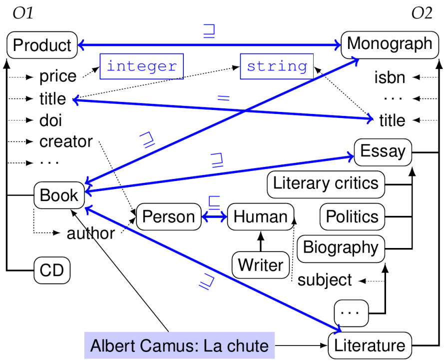


0.1.2 Ontology matching

Why do we need ontology matching ?

- One of the main challenges of *semantic technologies* is to solve problem of managing *semantic heterogeneity* among various information sources
- This *heterogeneity* introduces variations in meaning as well as ambiguity in entity interpretations
- *Ontology matching* is semantic technology that focus to solve this issue by automating *integration* of distributed information sources

Motivating example



Simple example of matching, taken from [1], between two different information sources : 1) ontology *O1* on the left side, 2) ontology *O2* on the right side. Classes are shown in rectangles with round corners (e.g. *Book* being subclass of *Product*), while relations are not bordered (e.g. *price* is an integer data type, while *creator* is an object property). *Albert Camus: Lachute* is a shared instance between the ontologies. Correspondence between entities of the ontologies are shown by thick blue arrows annotated with the relation that express it (e.g. *Person* in *O1* is subclass of *Human* from *O2*, which is marked by $\sqsubseteq$ symbol).

Basics

Definitions of core concepts (1)

- Given 2 ontologies, a **correspondence** is a 4-tuple $\langle id, e_1, e_2, r\rangle$, where *id* is identifier for correspondence, e_1 and e_2 are entities, e.g. classes and properties of the first and second ontology respectively, r is a relation holding between e_1 and e_2, e.g. equivalence, generalization/specialization, disjointness. An example of correspondence expressing that *Book* in *O1* is more general that *Monograph* in *O2* would be tuple $\langle id_{3,4}, Book, Monograph, \sqsupseteq\rangle$.
- Correspondence can have some associated *metadata* such as author name or confidence in correspondence typically expressed by a number within the range [0,1].

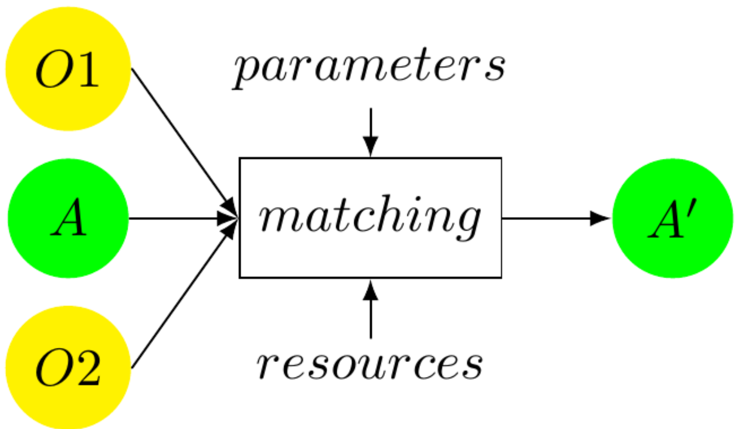
- **Alignment** is set of correspondences between entities belonging to the examined ontologies. Alignments are of cardinality *1:1*, *n:1* or *n:m*.

Definitions of core concepts (2)

- **Matching** is an operation that determines an *alignment A'* for a pair of ontologies *O1* and *O2*.

- Within context of *matching* it is usually referred to terms:
  - *matching operation* – focusing on the input and the result
  - *matching task* – focusing on the goal and the insertion of the task in a wider context
  - *matching process* – focusing on the internals of the task.

- *Matching operation* can be defined by use of: (i) an input *alignment A*, which is to be extended, (ii) the matching *parameters*, such as weights, or thresholds, and (iii) external *resources*, such as common knowledge or thesauri.



Inputs and result of *matching operation*

taken from [1].

Overview of matching systems

Comparison of matching systems

System	Input	Output	GUI	Operation	Terminological	Structural	Extensional	Semantic
SAMBO §4.1	OWL	1:1 alignments	Yes	Ontology merging	n-gram, edit distance, UMLS, WordNet	Iterative structural similarity based on *is-a*, *part-of* hierarchies	Naive Bayes over documents	-
Falcon §4.2	RDFS, OWL	1:1 alignments	-	-	I-SUB, Virtual documents	Structural proximities, clustering, GMO	Object similarity	-
DSsim §4.3	OWL, SKOS	1:1 alignments	AQUA Q/A [31]	Question answering	Tokenization, Monger-Elkan, Jaccard, WordNet	Graph similarity based on leaves	-	Rule-based fuzzy inference
RiMOM §4.4	OWL	1:1 alignments	-	-	Edit distance, vector distance, WordNet	Similarity propagation	Vector distance	-
ASMOV §4.5	OWL	n:m alignments	-	-	Tokenization, string equality, Levenstein distance, WordNet, UMLS	Iterative fix point computation, hierarchical, restriction similarities	Object similarity	Rule-based inference
Anchor-Flood §4.6	RDFS, OWL	1:1 alignments	-	-	Tokenization, string equality, Winkler-based sim., WordNet	Internal, external similarities; iterative anchor-based similarity propagation	-	-
AgreementMaker §4.7	XML, RDFS, OWL, N3	n:m alignments	Yes	-	TF·IDF, edit distance, substrings, WordNet	Descendant, sibling similarities	-	-

Analytical comparison of matching systems taken from [1]. The first half of the table shows the system name, the input format, cardinality of output alignments, weather GUI is provided, the ways in which system can process alignments, respectively. The second half of the table shows the available matching methods classified by kind of data that the algorithm work with.

Applications

Applications of matching systems

The ontology matching systems are used within two scenarios :

- **design-time matching**
  - used in traditional applications, characterized by heterogeneous models (e.g ontologies or database schemas), such as ontology evolution, ontology integration, data integration, data warehouses.
  - analysis and matching is done manually or semi-automatically
- **run-time matching**
  - used in emerging applications, characterized by dynamics, such as peer-to-peer information sharing, web service composition, search and query answering.
  - matching is done usually automatically with use of more explicit conceptual models compared to traditional applications.

References

[1] Pavel Shvaiko and Jérôme Euzenat. “Ontology matching: state of the art and future challenges”. In: *Knowledge and Data Engineering, IEEE Transactions on* 25.1 (2013), pp. 158–176.