The Role of Foundational Ontologies for Preventing Bad Ontology Design

Stefan SCHULZ [a,b,1]

[a] *Institute for Medical Informatics, Statistics and Documentation, Medical University of Graz, Austria*
[b] *AVERBIS GmbH, Freiburg, Germany*

Abstract. Ontology engineering is error-prone, and many published ontologies suffer from quality problems. This paper initiates a discussion about how axiomatically rich foundational ontologies can contribute to prevent and to detect bad ontology design. Examples T-boxes are presented, and it is demonstrated how typical design errors can be detected by upper-level axioms, in particular disjoint class axioms, existential and value restrictions. However, debugging large domain ontologies under an expressive top level raises scalability issues. During domain ontology design this can be mitigated by using small random ontology modules for debugging. For reasoning in applications, however, less expressive variants of such foundational ontologies are necessary.

Keywords. Foundational ontologies, description logics, quality assurance

1. Introduction

Constructing domain ontologies is a demanding endeavour. The formalization of basic regularities of a domain requires not only familiarity with the domain but also understanding of the basic principles of logic and formal ontology. This is especially the case if ontology engineering is understood not as building (closed-world) models limited to the support of well-delineated reasoning use cases in restricted domains, but as providing interoperable and re-usable (open world) representations of the domain itself.

This is a basic principle stressed not only by the defenders of so-called realist ontologies [1,2] but also by some (moderate) critics [3], which documents an increasing consensus on how to represent those areas of knowledge where people tend to agree on an observer-independent reality and benefit from standardised terms, such as in natural science and technology domains. An important tenet of these ontologies is collaborative ontology development and interoperability. Principles for this kind of ontology development have been formulated by the OBO Foundry consortium [4] and within the Good Ontology Design (GoodOD) guidelines [5]. Both propagate a concise foundational upper-level as a mainstay for interoperability, and there is also some evidence that foundational ontologies – domain-independent top-level or domain-related upper-level ontologies – speed up ontology development and improve quality [6].

[1] Corresponding Author: Stefan Schulz, Institute for Medical Informatics, Statistics and Documentation, Medical University of Graz, Auenbruggerplatz 2/V, 8036 Graz, Austria, E-mail: steschu@gmail.com

CEUR-WS.org/Vol-2205/paper22_bog1.pdf

This paper is intended to initiate a discussion on how foundational ontologies can help prevent typical errors in domain ontologies. This is also the reason why prototypical, partly made-up but easily understandable examples are used.

2. Materials and Methods

The work is based on BioTopLite (BTL2), an upper-level ontology [7], linked to BFO [8], using description logics [9] with OWL-DL expressiveness [10]. BTL2 has been designed with the intent to provide a rich set of constraining axioms to enforce the consistency of ontologies modelled thereunder. Although BTL2 is, in principle, domain-independent, its content is geared to the domains of health care and biomedical research. This explains, e.g., the provision of more fine-grained classes for chemical and biological entities (e.g. '*mono molecular entity*', *organism*, *cell*, *population*) as well as the disjunctive class *condition*, created in order to deal with the ontological heterogeneity of key medical concepts like diseases, signs, and symptoms. Fig. 1 provides Protégé screenshots of the class and relation hierarchies, together with sample axioms.

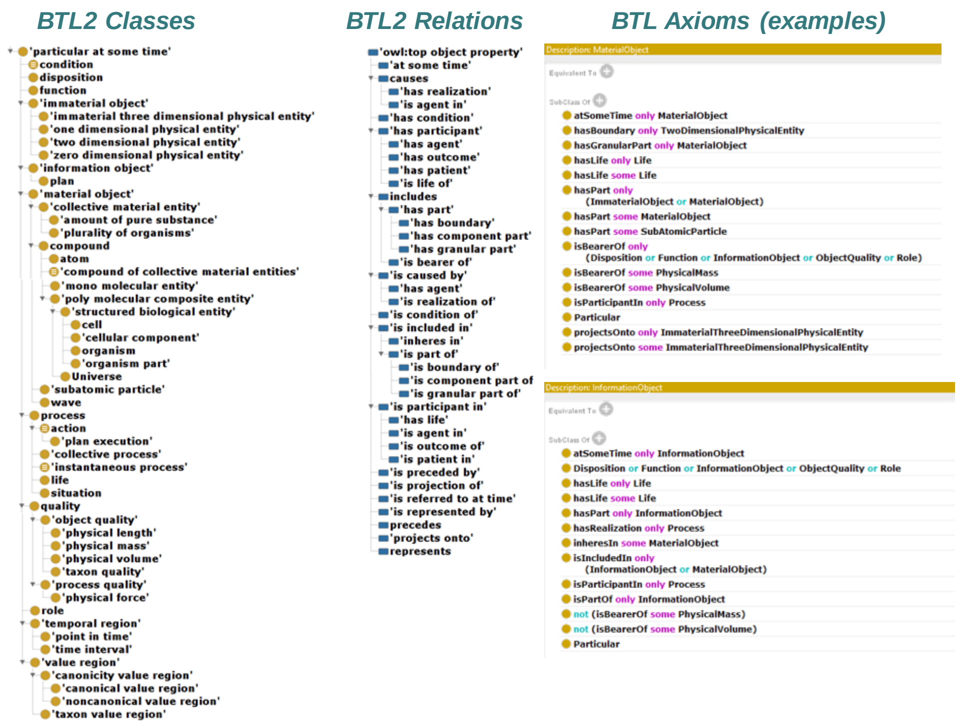

Figure 1. Classes, relations (object properties) and selected axioms in BioTopLite2 (BTL2)[7]

In order to test and demonstrate how BTL2 axioms are useful for preventing ontology design errors, T-boxes with typical examples of bad modelling will be presented in order to challenge the underlying foundational ontology. Some of these examples are formulated very abstractly due to their high level of generality; others use terms from a specific domain but are still understandable for a broader public.

Each T-box is modelled as an extension of BTL2. The HermIT [11] reasoner was used to detect inconsistencies. The explanation of inconsistencies follows Protégé's OWL entailment explanation feature [12]. The examples are provided together with their results in the following section. The presentation of the original OWL expressions, the entailments and their justifications is done in the following order:

- **SRC** – Axioms from ontology source (known satisfiable).
- **CHA** – New axioms added that challenge the satisfiability of **SRC**. Cases in which an axiom is only transiently added, are marked by **CHA-T**.
- **INF** – Inference, in particular the detection of classes that are unsatisfiable w.r.t. the T-box constituted by **SRC** and **CHA**.
- **EXP** – Explanations of **INF**, with axioms collected from the explanation plug-in [12].

3. Results

Reasoning examples under BTL2 are presented, in which deliberately erroneous axioms lead to an incoherent ontology, i.e., where one or more named classes turn out to be unsatisfiable, i.e. necessarily empty w.r.t. the T-box. This is expected to be detected by a DL classifier. In the following, a distinction is made between five error types, *viz*. (1) simple category errors, (2) value restrictions and transitive role errors, (3) complex domain/range constraint errors, (4) physical granularity errors, and (5) errors regarding unrealized realisables and non-referring information entities.

3.1. Simple category errors

A category error occurs whenever a class is a taxonomic descendent of upper-level classes that are modelled as mutually exclusive (Disjoint Classes in OWL). Such errors typically occur when ontology mapping and alignment is guided by lexical criteria. For instance, when mapping content of the clinical ontology SNOMED CT [13] (namespace sct:) to the foundational ontology BTL2 (namespace btl2:) one might be tempted to equate sct:*Process* with btl2:*process* and place 'sct: *Qualifier Value* (*qualifier value*)' under 'btl2:*quality*':

SRC	'sct:*Process* (*qualifier value*)' SubClassOf 'sct:*Qualifier Value* (*qualifier value*)'	(1a)
NEW	'sct:*Qualifier Value* (*qualifier value*)' SubClassOf btl2:*quality*	(1b)
NEW	'sct:*Process* (*qualifier value*)' EquivalentTo btl2:*process*	(1c)
INF	'sct:*Process* (*qualifier value*)' EquivalentTo owl:*Nothing*	(1d)
EXP	btl2:*quality* DisjointWith btl2:*process*	(1e)

The origin of such category errors may be manifold. A common cause is misleading class labelling. Ontology labels should be context-independent and self-explanatory, and they should avoid ambiguous terms [14]. A more severe problem – like here – arises where the ontologies to be aligned fundamentally differ in upper level assumptions. In SNOMED CT, e.g., the subhierarchy 'sct:*Qualifier Value* (*qualifier value*)' is currently a badly organized reservoir for the most diverse terms, and the reason why hierarchies of pathological and physiological processes are placed therein remains unclear[2].

3.2. Value restrictions and transitive roles

Value restrictions (universal constraints expressed by "only" in OWL Manchester syntax) restrict the range of allowed role fillers. It is tempting to use value restrictions together with mereological statements, e.g. stating that all members of a class have only parts of a certain kind, like in the following example:

SRC	btl2:*cell* SubClassOf btl2:*compound*	(2a)
SRC	'*cell culture*' SubClassOf 'btl2:*material object*'	(2b)
CHA	'*cell culture*' SubClassOf and ('btl2:**has part**' some btl2:*cell*) and ('btl2:**has part**' only btl2:*cell*)	(2c)
INF	'*cell culture*' EquivalentTo owl:*Nothing*	(2d)
EXP	'btl2:*material object*' SubClassOf 'btl2:**has part**' some 'btl2:*subatomic particle*'	(2e)
EXP	btl2:*compound* DisjointWith 'btl2:*subatomic particle*'	(2f)

Axiom like (2c) may fulfil their purpose in domain ontologies in which cells are the smallest objects, but as soon as smaller objects are allowed, the expression is inadequate – assuming **'has part'** being transitive. BTL2 obviates such a granularity restriction by stating that all material objects have subatomic particles as (transitive) parts.

3.3. Complex domain / range restrictions

Domain / range restrictions are a suitable means to avoid ontology errors. However, in ontologies that use a small number of object properties like BTL2, this resource may be not expressive enough. For instance, if the relations 'btl2:**is part of**' and 'btl2:**has part**' are valid for material objects, immaterial objects, as well as for processes and information objects, formulating just domain / range restrictions at the level of these relations would still be compatible with an (unintended) model in which an object is part of a process or vice versa. This is why BTL2 encodes these restrictions using axioms like the one in (3f).

SRC	*Object_A* SubClassOf 'btl2:*material object*'	(3a)
SRC	*Process_A* SubClassOf btl2:*process*	(3b)
CHA	*Object_A* SubClassOf 'btl2:**is part of**' some *Process_A*	(3c)
INF	*Object_A* EquivalentTo owl:*Nothing*	(3d)
EXP	'btl2:**has part**' InverseOf 'btl2:**is part of**'	(3e)
EXP	btl2:*process* SubClassOf 'btl2:**has part**' only btl2:*process*	(3f)
EXP	btl2:*process* DisjointWith 'btl2:*material object*'	(3g)

[2] Processes, like all kinds of entities, may play the role of values in information models, e.g., "Infectious process". Their confusion with their referents, i.e. domain entities proper (an infectious process in a patient) is a classic case of use-mention confusion, which still haunts many domain ontologies [15], especially those deriving from frames, thesauri and other knowledge organization systems.

A similar example is typical for medical ontologies, where domain experts are tempted to use “diagnosis” and “disease” interchangeably, and where signs and symptoms are referred to, in colloquial discourse, as being “parts” of diagnoses:

SRC	*Diagnosis_A* SubClassOf *Diagnosis*	(3h)
SRC	*Diagnosis* SubClassOf ‘btl2:*information object*’	(3i)
SRC	*Symptom_A* SubClassOf *Symptom*	(3j)
SRC	*Symptom* SubClassOf btl2:*condition*	(3k)
CHA	*Diagnosis_A* SubClassOf ‘btl2:**has part**’ some *Symptom_A*	(3l)
INF	*Diagnosis_A* EquivalentTo owl:*Nothing*	(3m)
EXP	‘btl2:*information object*’ *SubClassOf* ‘btl2:**has part**’ only ‘btl2:*information object*’	(3n)
EXP	btl2:*condition* EquivalentTo btl2:*function* or btl2:*disposition* or ‘btl2:*material object*’ or btl2:*process*	(3o)
EXP	DisjointClasses: ‘btl2:*material object*’, btl2:*process,* btl2:*function* or btl2:*disposition,* ‘btl2:*information object*’, ‘btl2:*immaterial object*’, btl2:*role,* btl2:*quality,* ‘btl2:*temporal region*’*,* ‘btl2:*value region*’	(3p)

3.4. Physical granularity

Ontologies for natural sciences and technology deal largely with physical objects of several degrees of granularity. Levels of material granularity obey certain mereological constraints, e.g. that biological cells can be parts of organisms but not vice versa, or that polymolecular entities can never be part of single molecules. BTL2 incorporates such constraints. They help detect modelling errors like the following one.

SRC	*Chromosome* SubClassOf ‘btl2:*poly molecular composite entity*’	(4a)
SRC	*ProteinMolecule* SubClassOf ‘btl2:*mono molecular entity*’	(4b)
CHA	*Chromosome* SubClassOf ‘btl2:**is part of**’ some *ProteinMolecule*	(4c)
INF	*Chromosome* EquivalentTo owl:*Nothing*	(4d)
EXP	‘btl2:*poly molecular composite entity*’ *and* (‘btl2:**is part of**’ some (btl2:*atom* or ‘btl2:*mono molecular entity*’ or ‘btl2:*subatomic particle*’)) SubClassOf owl:*Nothing*	(4e)

3.5. Unrealised realisables and non-referring information entities

Realisable entities like functions and dispositions [16] depend on material entities and are realised in processes. However, the existence of realisables does not imply their realisation: The function of a screwdriver is to drive screws, and the disposition of a glass is to break under certain circumstances, but as there are screwdrivers that are never used and glasses that are never thrown to the floor. Because for all types of functions and dispositions there are instances that have never been realised, ontologies have to deal with unrealised realisables, which could be, e.g., expressed by (5a).

SRC	*Unrealized_Function* EquivalentTo btl2:*function* and not (‘btl2:**has realization**’ some owl:*thing*)	(5a)

In order not to preclude the possibility that dispositions and functions happen to be never realised, ontologies under BTL2 should define them using the value restriction constructor:

SRC btl2:*function* SubClassOf 'btl2:**has realization**' only btl2:*process* (5b)
SRC *Function*_A EquivalentTo btl2:*function* and ('btl2:**has realization**' only *Process_A*) (5c)

Nevertheless, BTL2 does not reject a definition using an existential quantifier like in the following definition:

CHA *Function*_A EquivalentTo btl2:*function* and ('btl2:**has realization**' some *Process_A*) (5d)

Function_A would then exclude all unrealised function instances. Such a class (which might be considered anti-rigid [17] if assuming that realisable entities are always unrealised when they come into existence) is most likely not intended by the modeller. The detection of these errors requires checking consistency after transiently adding axiom (5e).

CHA-T btl2:*function* EquivalentTo *Unrealized_Function* (5e)
INF btl2:*function* EquivalentTo owl:*Nothing* (5f)

The explanation is given by the conjunction of axioms (5a), (5d), and (5e).

The axiomatization of non-referring information entities follows the same pattern. Information entities can be referring and non-referring [18]. A typical example is medical diagnosis [19]. BTL2 here uses the relation *represents*. This relation connects information entities with domain entity they correctly characterise. This allows to distinguish wrong diagnoses from correct diagnoses.

SRC *Diagnosis* SubClassOf 'btl2:*information object* (5g)
SRC *False_diagnosis SubClassOf Diagnosis* and (not btl2:**represents** some btl2:*condition*) (5h)
SRC *Cancer_Diagnosis* EquivalentTo *Diagnosis and* btl2:**represents** only (*Cancer* or not btl2:*condition*) (5i)
CHA *Cancer_diagnosis* EquivalentTo *Diagnosis* and btl2:**represents** some *Cancer* (5j)

Challenged by the axiom (5k) the T-box becomes incoherent.

CHA-T *Diagnosis* EquivalentTo *False_Diagnosis* (5k)
INF *Cancer_Diagnosis* EquivalentTo owl:*Nothing*

The explanation is given by the conjunction of axioms (5h), (5i), and (5k).

4. Discussion and Further Work

It was shown how a highly axiomatised foundational ontology like BioTopLite (BTL2) can incorporate constraints that reject bad modelling decisions that lead to unsatisfiable classes. A distinction was made between those cases in which the upper level axioms suffice for detecting such inconsistencies and those in which additional "challenges", i.e. transiently added axioms are necessary.

Most of the former cases capitalise on disjoint class axioms present in the upper level ontology. This comes near to the so-called logical anti-patterns introduced by [20], all of which require disjoint class axioms in order to detect inconsistencies. In contrast to the work presented, anti-patterns are very abstract logical expressions and independent of foundational ontologies. OntoClean [17] was presented as a methodology for detecting improper subclass axioms based on philosophically inspired, domain-independent properties of classes, the metaproperties unity, identity and rigidity. Although DL reasoners do not support meta-level reasoning, it has to be investigated whether certain elements from OntoClean could also included in DL-based foundational ontologies, e.g. by reifying them in terms of additional top-level classes[3].

Several limitations of this work have to be highlighted:

- The typology presented is certainly non-exhaustive. It is primarily motivated by the author's experience and not yet by the relevance of those types of problems in ontologies employed in real-world applications. It could further be related to existing work in ontology evaluation, e.g., the OQuaRE framework [21].
- The proposed approach will probably fail if application ontologies bypass the partition of upper categories of the foundational ontology or introduce new object properties that are not subproperties of the existing ones. The BTL2 authors claim that their inventory of object properties is close to sufficient and recommend to introduce predicates required by the domain (e.g., in the biomedical domain: *treats*, *prevents*, *diagnoses*, *interacts*, *binds*) not as object properties but as subclasses of btl2:*process*.
- Important causes of bad ontology design cannot be prevented or remedied by a foundational ontology. This includes erroneous representation of individuals as classes or vice versa (a typical error would be an A-Box OWL expression like "*SodiumAtom* Type btl2:*atom*), bad naming and insufficient documentation, as well as constraints on a meta-class level like in OntoClean.
- Constraining axioms similar to the proposed ones can be added to domain ontologies, e.g. to assure mereotopological non-overlapping [22].
- Although BTL2 was used as a testbed, the proposed approach would lend itself to other ontologies as well. Especially BFO would benefit from a stronger axiomatization, as the most popular version, which is the umbrella of most OBO ontologies lacks axioms beyond subclass and disjoint class axioms. This deficiency has been addressed by the 2.0 version, which is, however, not fully available in OWL due to its use of ternary relations.
- The fact that BTL2 uses the whole range of constructors allowed by OWL-DL has a negative impact on reasoning performance. This makes debugging of large ontologies intractable. This was the case when aligning SNOMED CT with BTL2 [23]. The solution was to use small modules created from random

[3] Which would be orthogonal to the existing ones, e.g. 'rigid entity', 'anti-rigid entity', 'whole'

signatures as described in [24]. As a solution a two-step approach was proposed: (i) at design time using the rich foundational ontology for debugging random modules of a (large) domain ontology under construction, and (ii) at runtime placing the final domain ontology under a light version of the same foundational ontology for enabling efficient reasoning.

References

[1] I. Johansson, Bioinformatics and biological reality. *J. Biomed Inform* **39** (2006), 274–287

[2] B. Smith, W. Ceusters, Ontological realism: A methodology for coordinated evolution of scientific ontologies. *Applied Ontology* **5** (2010) 139–188.

[3] P. Lord and R. Stevens. Adding a little reality to building ontologies for biology. *PLoS one* **5.9** (2010), e12258.

[4] M. Ashburner, C. Rosse, J. Bard, W. Bug, W. Ceusters, L.J. Goldberg, K. Eilbeck, A. Ireland, C.J. Mungall, the OBI Consortium, N. Leontis, P. Rocca-Serra, A. Ruttenberg, S. A. Sansone, R.H. Scheuermann, N. Shah, P.L. Whetzel, S. Lewis, The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration, *Nature Biotechnology* **25** (2007), 1251.

[5] S. Schulz, D. Seddig-Raufie, N. Grewe, J. Röhl, D. Schober, M. Boeker, L. Jansen, *Guideline on Developing Good Ontologies in the Biomedical Domain with Description Logics*, http://www.purl.org/goodod/guideline , 2012.

[6] C. M. Keet. The use of foundational ontologies in ontology development: an empirical assessment. *Extended Semantic Web Conference*. Springer, Berlin, Heidelberg (2011) 321-335.

[7] S. Schulz. M. Boeker, C. Martinez-Costa, The BioTop family of upper level ontological resources for biomedicine, *Stud Health Technol Inform*. **235** (2017), 441–445.

[8] R. Arp, B. Smith, A. Spear, *Building Ontologies with Basic Formal Ontology*, MIT Press, 2015.

[9] F. Baader (ed.), *The Description Logic Handbook, Second Edition*. Cambridge University Press, 2010.

[10] W3C, *OWL2 Web Ontology Language* http://www.w3.org/TR/owl2-overview/ , 2012.

[11] B. Glimm et al. HermiT: an OWL 2 reasoner. *Journal of Automated Reasoning*, **3** (2014), 245–269.

[12] M. Horridge, B. Parsia and U. Sattler. Explanation of OWL entailments in Protégé 4. *Proceedings of the Poster and Demonstration Session at the 7th International Semantic Web Conference*. CEUR-WS. org, http://ceur-ws.org/Vol-401/iswc2008pd_submission_47.pdf , 2008.

[13] SNOMED international. *SNOMED CT* , https://www.snomed.org/snomed-ct , 2018

[14] D. Schober, B. Smith, S.E. Lewis, W. Kusnierczyk, J. Lomax, C. Mungall, C.F. Taylor, P. Rocca-Serra, S.A. Sansone, Survey-based naming conventions for use in OBO Foundry ontology development, *BMC Bioinformatics* **10.1** (2009), 125.

[15] B. Smith, Against idiosyncrasy in ontology development. *Frontiers in Artificial Intelligence and Applications* **150** (2006), 15.

[16] J. Röhl, L. Jansen, Why functions are not special dispositions: an improved classification of realizables for top-level ontologies. *Journal of biomedical semantics* **5.1** (2014), 27.

[17] N. Guarino, C.A. Welty. An overview of OntoClean. *In Handbook on ontologies*, 151–171, Springer, Berlin, Heidelberg, 2004.

[18] S. Schulz, M. Brochhausen, R. Hoehndorf, Higgs bosons, Mars missions, and unicorn delusions: How to deal with terms of dubious reference in scientific ontologies. *Proc. of ICBO 2011*. CEUR-WS 833 (2011), http://ceur-ws.org/Vol-833/paper24.pdf .

[19] S. Schulz, C. Martínez-Costa, D. Karlsson, R. Cornet, M. Brochhausen, A.L. Rector, An ontological analysis of reference in health record statements. *Formal Ontology in Inf. Systems* (2014), 289–302.

[20] C. Roussey, O. Corcho, L. M. Vilches-Blázquez. A catalogue of OWL ontology antipatterns, *Proceedings of the fifth international conference on Knowledge capture*. ACM, 2009.

[21] A. Duque-Ramos J.T. Fernández-Breis, R. Stevens, N. Aussenac-Gilles, OQuaRE: A SQuaRE-based approach for evaluating the quality of ontologies. *Journal of Research and Practice in Information Technology* **43.2** (2011), 159.

[22] M. Boeker, J. Hastings, D. Schober, S. Schulz (2011). A T-Box generator for testing scalability of OWL mereotopological patterns. *OWLED 2011 – OWL Experiences and Directions*. Eight International Workshop, San Francisco, California, USA, June 5-6, 2011.

[23] S. Schulz, C. Martínez-Costa, Harmonizing SNOMED CT with BioTopLite: An Exercise in Principled Ontology Alignment. *Stud Health Technol Inform*. **216** (2015), 832–836.

[24] P. López-García, S. Schulz, Can SNOMED CT be squeezed without losing its shape? *J Biomed Semantics* **7** (2016), 56.