July 2020

Towards a Framework for Meaning Negotiation and Conflict Resolution in Ontology Authoring

Rolf GRÜTTER [a,1] and C. Maria KEET [b]
[a] Swiss Federal Research Institute WSL, Birmensdorf, Switzerland
[b] Department of Computer Science, University of Cape Town, South Africa

Abstract. Ontology authoring involves making choices about what subject domain knowledge to include. This may concern sorting out ontological differences as well as making choices of conflicting axioms due to limitations in the logic. Examples are different foundational ontologies in ontology matching and OWL 2 DL's transitive object property versus qualified cardinality constraints. Such conflicts have to be resolved. However, there is currently only isolated and fragmented guidance for doing so, which therefore results in ad hoc decision-making. This work aims to address this by working towards a framework dealing with the various types of modeling conflicts through meaning negotiation and conflict resolution in a systematic way. The approach was evaluated with an actual case of domain knowledge usage in the context of epizootic disease outbreak.

Keywords. Meaning Negotiation, Conflict Resolution, Ontology Authoring, Infectious Disease, Disease Control

1. Introduction

An increase in the use of ontologies brings with it the task of reusing existing ones, which is already an aspect of the OBO Foundry approach [1] and incorporated in ontology development methodologies such as NeON [2]. This may be as a single ontology, or imported, merged, or integrated with another. It can become difficult to assess potential for (re)use, as discussed in detail for, e.g., parthood theories [3,5] and deciding on a top-level ontology [6]. Since one feature of ontologies is to tease out subtle differences in meaning, a candidate ontology for use or import either may not have all the desired axioms, have too many axioms, or upon import it may result in an inconsistent or incoherent ontology, be beyond the desired OWL species or otherwise incompatible with one's preferred ontology language. Examples of overlap and reuse experiences vary widely also in the biology domain; recent examples include practical reuse of the Infectious Diseases Ontology for schistosomiasis knowledge [7], the modular design and many reuses of the Gene Ontology [8], and subtle differences across disease ontologies [9], among many. Issues may include, among others, merging two domain ontologies that are aligned to

[1] Corresponding Author: Rolf Grütter, Swiss Federal Research Institute WSL, Zürcherstrasse 111, CH-8903 Birmensdorf, Switzerland; E-mail: rolf.gruetter@wsl.ch.

two different foundational ontologies, a class Infection versus a property infected-by, or one ontology has Virus as a living thing and the other does not.

How to proceed? While one could discard a relevant ontology and start afresh, we assume that a user may wish to attempt to resolve any issues that may arise. A few tools are available to assist with detecting and inspecting issues; e.g., the explanation features in Protégé [10], assessing the differences in inferences [11], and the OWL Species Classifier[2] that lists which axiom(s) violate which OWL species. Also, there are *a priori* choices one can make in comprehensiveness vs. expressiveness [3] and testing for conflicts [16], as compared to a try-and-see approach. These methods and tools will not detect all sources of conflicts, however, such as between fundamental assumptions about a domain or preferred theories. For instance, choosing parthood or connection or both as primitive for a mereotopological theory [13], and whether it is "better" for one's domain to declare parthood as transitive or use it in qualified number restrictions (since one cannot have both in OWL 2 DL [14]). Consider, e.g., some ontology O_1 about anatomy that has declared that a biped is an animal that has part exactly two legs: when it is aligned to DOLCE, it will clash with dolce:has-part that is declared transitive. What can the ontology engineer do? One could *i)* decide to not merge with DOLCE, *ii)* give up on the qualified cardinality constraint and modify the definition of biped, *iii)* import DOLCE separately and remove transitivity making it *de facto* incompatible with DOLCE, *iv)* accept to go beyond OWL 2 DL and use a different logic, or *v)* forsake automated reasoning over one's ontology. The consequences of each choice would then still have to be assessed somehow. This may leave the ontologist with *ad hoc* attempts of trial and error, and therewith hampering redeployment of ontologies, also because the possible consequences of possible solutions may not be clear.

This work aims to contribute to addressing these obstacles by devising a novel approach for *meaning negotiation* in ontology development and (re)use, and for *conflict resolution*. We examine the possible principal sources of conflicts for both individual ontologies and multiple ontologies. For each case, there is a fixed set of feasible solution strategies, so that then explanatory implications may be automatically generated. Some of the components can be computed automatically, whereas others require the human-in-the-loop to make the final, but now well-informed, decision. The approach is illustrated and evaluated with a case of domain knowledge usage to manage an epizootic disease outbreak in Switzerland (avian influenza), which involved negotiation and resolving conflicts regarding the 'appropriate' mereotopological theory, its trade-off with the OWL species, and exceeding OWL 2 DL when combining ontologies.

The remainder of the paper is structured as follows: An approach to meaning negotiation and conflict resolution is introduced in Sections 2 and 3, respectively. The use case is presented in Section 4. A conclusion is drawn in Section 5.

2. Characterizing Meaning Negotiation and Conflict Resolution

Negotiating the meaning of the knowledge to be represented in an ontology involves reaching an agreement on: 1) the exact elements required, 2) the domain theory that will provide these elements, and 3) the required ontology language to represent the former. This may involve *meaning negotiation* and *conflict resolution*, which will be disam-

[2] https://github.com/muhummadPatel/OWL_Classifier/

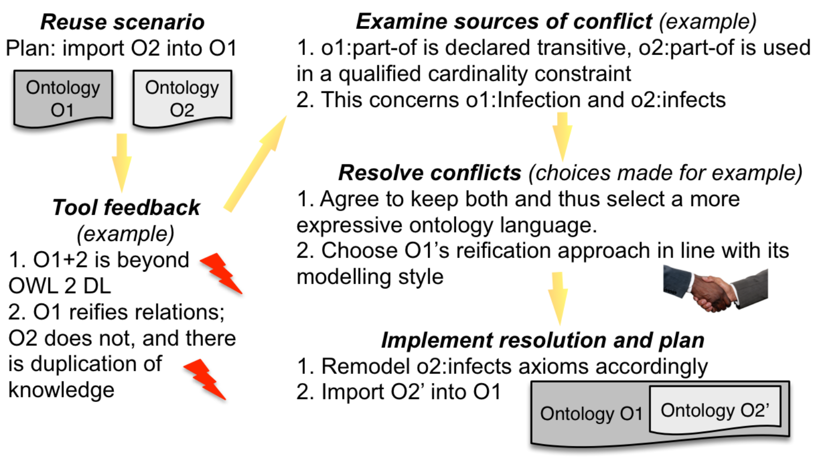

Figure 1. Sample scenario (summarized) of detecting and resolving conflicts in an ontology reuse scenario where ontology Onto2 is being imported into ontology Onto1.

biguated and illustrated first. Potential sources of conflict are identified and the resolution processes elaborated on in more detail.

2.1. Types and Sources of Conflicts

We define informally the notions of meaning negotiation and conflict resolution, illustrate them, and then outline how they arise.

Meaning negotiation concerns deliberations to figure out the precise semantics one wants to represent in the ontology. They are all *positive choices* in the sense of 'which of the options is applicable? then we take that one'.

Conflict resolution concerns choosing one option among a set of two or more options, where that choice is deemed the *'lesser among evils' for that scenario*, necessarily involves a compromise, and making it work requires reengineering something in at least one of the ontologies or as a whole. Subtypes:

- **Language conflict resolution** A conflict arises within the same family of languages or with a more distant one. This is a zero-sum game (i.e., with a winner and a loser) or there may be a joint outside option.
- **Ontological conflict resolution** The ontologies adhere to different theories. They may be foundational philosophical decisions that affect the overall structure of the ontology or subject domain arguments with competing theories. This is likely a zero-sum game (no joint outside option).

They are illustrated in the following example.

Example 1 *Meaning negotiation may include assistance with explanations, such as when modelers are not sure whether they need a mereological theory with or without atom, offering them a dialogue* "if you *add Atom to ground mereology,* then you obtain *the following novel deductions [listing];* do you want that?*" or frame negotiation of alternative commitments as an imperative, e.g.,* "take either *parthood* or *proper parthood as primitive* for your *mereological theory,* but not both.*"*.

Conflict resolution applies to many situations. For instance, there are several possible types of language conflicts, of which a few common ones are:

- *A typical example of a conflict within a language family, such as the Description Logics-based OWL species, is the conflict of either transitivity or qualified cardinality constraints, but not both, with the same object property, as illustrated in the Introduction with* biped *and* has-part.
- *A syntax-level conflict, such as having to merge an ontology represented in CLIF and another one represented in OWL, or OBO and OWL.*
- *A language's semantics issue: e.g., ontologies represented in different languages where one has a model-theoretic semantics and the other a graph-based one, and open vs closed world assumption.*

What to do then? Besides choosing either, there may be a so-called 'joint outside option' (a term from game theory) where neither wins, but there is an alternative option. For instance, instead of debating over CLIF or OWL, one can keep both and move outside either setting and into the DOL framework [4], as illustrated in [3], and instead of transitivity vs qualified cardinality, leave OWL to choose CLIF.
An example of an ontological conflict is a clash in the top-level organisation of the ontology, such as between BFO and GFO, and related philosophical differences, such as whether concepts are allowed in the ontology. At the subject domain level, this may be, e.g., whether a virus is a living thing or not, and, more generally, competing scientific theories. In principle, they do not have a joint outside option, other than reverting it back to the domain experts to resolve this by, e.g., conducting experiments in the lab. ♢

Where do such conflicts come from? The sources of issues arising can be manifold. Six principal cases were discerned, which are non-exclusive and possibly not exhaustive:

1. *Ontological differences between established theories*; e.g., extensional mereology (EM) vs. minimal mereology (MM) and DOLCE vs. BFO as top-level ontology.
2. *Ontological differences at the axiom-level*; e.g., whether part-of is antisymmetric.
3. *Different modeling styles*; e.g., foundational ontology-inspired or conceptual model-influenced; e.g., reification or not [15], like the Infection/infected-by mentioned in the Introduction.
4. *Logic limitations causing conflicts for an ontology, affecting the software ecosystem*; e.g., the biped's has-part being either transitive or have it participate in axioms with qualified cardinality constraints in OWL 2 DL. Resolution options include considering tools outside the Semantic Web infrastructure.
5. *Logic limitations by design for scalability*; e.g., there are axioms in one's ontology that are beyond the desired OWL species, so that one has to choose to abandon the preferred species or remove the axioms.
6. *Certain deductions made by the reasoner (excluding modeling mistakes)*; e.g., an unsatisfiable class due to signed but disjoint ancestors. While this may also have as source an ontological difference at the axiom-level, it manifests either after adding the axioms, during test-driven development (TDD) [16], or upon ontology matching attempts.

The first three are, in principle, *a priori* negotiations by an ontologist, but may manifest only upon ontology matching. Cases 4 and 5 emerge during ontology authoring. The last one may or may not be *a priori*.

2.2. The Conflict Set

Conflict detection offers opportunities for automation and, even though there is no single way of how conflicts can be detected, some tasks can be carried out with the aid of state-of-the-art ontology development environments (ODEs) (see Section 3).

The data structure in which the detected conflicts are stored, and upon which the resolution of conflicts operates, is called *conflict set*. A conflict set is generated in all cases where a conflict is detected. We will illustrate this in Section 4. Without loss of generality, it is assumed that, when matching more than two ontologies, a conflict set is generated for every pair. Conflict sets can be described in a context-free grammar in Backus-Naur Form as follows (the productions for most terminals are omitted):

```
<conflict-set> ::= <ontology> <ontology> [<diff>]
<ontology>     ::= <IRI> [<species>] <axiom> {<axiom>} [<inference>]
<species>      ::= "OWL DL" | "OWL Lite" | "OWL Full" | "OWL 2 EL" |
                   "OWL 2 QL" | "OWL 2 RL" | "OWL 2 DL" | "OWL 2 Full" |
                   "FOL" | "HOL"
<axiom>        ::= [<number>] <formula> [<description>] {<theory>}
                   {<dl-expressivity>}
<theory>       ::= <IRI> | <name> | <IRI> <name>
<diff>         ::= difference between the inferred axioms sets
                   of the two ontologies
```

Accordingly, there are two ontologies (or two fragments of the same ontology), each identified by an IRI (or another identifier) and composed of a (possibly singleton) set of axioms. An axiom may adhere to an ontologically well-founded theory, such as BFO or ground mereology.

3. Resolving Conflicts

In practice, conflict resolution often starts with some issue raised by the ODE, specifically, when an axiom is added, or an ontology is merged or integrated into the active ontology. Examples of such issues are *undecidability*, *language profile violation*, and *incoherence*. They can be seen as cues indicating that something is wrong with the active ontology. The author then has to find out what raised the issue. Thereby, they may be supported by the ODE. Proceeding that way is not as straightforward as one might expect, because there is no one-to-one correspondence between conflict and issue. Examples of such 'causal investigations' will be given in Section 3.2–3.5. For the rest of this section, the following principal choices are presupposed:

(i) The author sticks to Occam's razor when authoring an ontology for the case at hand: the least expressive language in which the required axioms can be represented fully is preferred over all more expressive ones.
(ii) The author wants to capture as much of the semantics of the domain theory as possible.
(iii) The author prefers a decidable language over FOL or HOL for representing a domain theory and a coherent ontology over an incoherent one.

The first point is a general principle in almost every situation in life. The second point assumes that the author prefers representing a full axiomatization over a partial axiomatization and, by extension, a partial one is better than mere primitives without any axioms.

While the third point may not hold in all situations, we deemed it realistic to include, since most software infrastructure caters for decidable ontology languages and coherent ontologies, and Semantic Web and Knowledge Graph languages in particular.

Table 1. A sample of conflicts possibly emerging during ontology authoring

No.	Conflict	Description	Examples
		Conflicting theories at the top-level	
1	foundational	ontologies adhere to conflicting theories	BFO, DOLCE, GFO, SUMO, UFO, YAMATO
2	mereological	conflicting mereological theories	with vs. without Atom, whether parthood is antisymmetric or not, weak vs. strong supplementation
		Conflicting theories at the subject domain level	
3	domain theory	competing theories	monotheism vs. polytheism, marxism vs. leninism
4	status of an element	competing (scientific) theories	whether virus is a living thing or not
		Axiom-level conflicts	
5	ontological	conflicting theories acting out on the axiom-level	see rows 1–3
6	within-language family	undecidable violation of a language profile	some of the non-admissible axiom combinations in Example 2
		decidable violation of a language profile	functional and transitive properties in OWL 2 QL
		Other conflicts	
7	modeling style	applied vs. foundational	there is / there is no data property axiom
		class vs. object property	Infection vs. infected-by

3.1. Conflicting Theories

If an ontologically well-founded theory underlying some axioms to add or an ontology to integrate is in conflict with the ontology representing the desired theory (nos. 1–3 in Table 1), the respective IRIs must be added to the conflict set. This presupposes that the conflict is known and the pair of IRIs is listed somewhere, for instance, in a *library of common conflicts*. To give an example, if one considers adding the part-whole relations taxonomy that happens to be aligned with DOLCE to a BFO-aligned infectious disease ontology (IDO), then the theory conflict (BFO vs. DOLCE) will be detected by looking up the library of common conflicts. Conflict resolution, in this case, aims at preserving a consistent theory. Since for conflicting theories there is no joint outside option, the ontology author has to decide in favor of one theory and discard the other. Their decision may be informed by the deliberations of what should be represented in the ontology made during meaning negotiating.

State-of-the-art ODEs support the import of ontologies. After import, their IRIs can be read from the metadata of the active ontology and looked up manually for common conflicts in a library. Accordingly, this conflict detection approach is straightforward to implement. Uncommon conflicts are harder to detect, hence to resolve. The use of ontology design patterns for a theory would be helpful in automating detection, as would annotations. Further, the library of common conflicts may grow upon finding more conflicts, so that it can prevent the same or a similar conflict from emerging later on in the project.

3.2. Conflicts Manifesting Themselves in an Undecidable Language

See nos. 5–6 in Table 1. Here, conflict resolution aims at preserving a decidable ontology language or raising awareness of undecidability when opting for a joint outside option. In the first case, this most often is a zero-sum game: the ontology author has to chose which ones of some conflicting axioms in the conflict set to keep. For mereotopological theories, these types of conflicts are well-investigated [3]. In most instances, incorporating a full axiomatization renders the active ontology at least undecidable, and possibly also incoherent (i.e., with at least one unsatisfiable class [17]). To support the author's decision, some criteria can be established such as the following:

- Least number of axioms affected;
- preferred axiom type identified by assigning weights;
- least number of inferences lost.

The *least number of axioms affected* can be read from the conflict set. Assigning weights to *axiom types* implies that certain types are *a priori* considered more valuable than others. For instance, one may weigh existentials more than universals and unqualified cardinality more than qualified cardinality. The *least number of inferences lost* requires an additional step at which the inferences of the ontologies are computed and recorded in the conflict set. If undecidability is caused by an *ontological conflict* at the axiom-level that was not resolved along with conflicting theories (e.g., weak vs. strong supplementation in Example 2), then also the decisions taken when negotiating meaning upfront may serve as a criterion. The authors' decision and the criteria upon which it is based should be recorded, in case the same or a similar conflict emerges later on in the project.

The second case (opting for a joint outside option) requires that principal choice (iii) (preferring a decidable language) is relaxed. Theories that are represented in different logics can be dealt with by the DOL framework [4]. This includes cases where the resulting logic is undecidable.

State-of-the-art ODEs provide some support for detecting and resolving these kinds of conflicts. To give an example, the OWL API [18] of Protégé 5.2 [19] issues an error message reporting the conflict arising from a violation of the expressive OWL 2 DL specification, caused by a non-admissible axiom combination such as those listed in Example 2. In addition, Protégé 5.2 is equipped with an OWL reasoner, and a diff tool for computing the differences between OWL ontologies is available as a plug-in [20]. In order to compute the number of inferences lost, the axioms inferred from the merged ontology are first computed using the OWL reasoner. This requires that the merged ontology is saved as two decidable versions by removing one conflicting axiom set in exchange for the other. The difference between the sets of inferred axioms is then computed.

Example 2 *Some mutually exclusive axiom combinations are as follows:*

- *Within language family: OWL 2 DL with transitivity or role chain excludes any of minimum cardinality, maximum cardinality, exact cardinality, functionality, inverse functionality, reflexivity, irreflexivity, asymmetry, role disjointness.*
- *Mereology: weak supplementation ($pp(x,y) \rightarrow \exists z(p(z,y) \wedge \neg o(z,x))$) in MM vs. strong supplementation ($\neg p(y,x) \rightarrow \exists z(p(z,y) \wedge \neg o(z,x))$) in EM.*

- *Temporal logics: dense time* ($\forall t, t' \in T, t < t', \exists t''. t < t'' < t'$) *vs. discrete time (there is a first and last time point t, and no time point between t and* $t+1$*).*

3.3. Conflicts Manifesting Themselves in a Language Profile Violation

See nos. 5–6 in Table 1. Here, the case where conflict resolution aims at preserving the language to the extent that it is decidable again has to be distinguished from the case where the original language profile should be preserved. Presupposing the principal choices above, conflict resolution in the first case aims at capturing as much of the semantics of the domain theory as possible. Since the violated language profile is not the most expressive one, there may be room for a (decidable) joint outside option. To give an example, the axiom $O(x,y) =_{def} \exists z(P(z,x) \wedge P(z,y))$ (i.e., 'overlaps') cannot be expressed in any decidable OWL species. While preserving decidability, the author may still want to state that $P^{-}(x,z) \wedge P(z,y)$ is a sufficient condition for $O(x,y)$. Or they may want to state that $O(x,y)$ is a reflexive and symmetric property. Doing so may violate the original language profile. Whether the ontology language still is undecidable with the modified axioms and conditions can be figured out in the same way as described in Section 3.2. Weakening the theory step by step this way will end up in a decidable language, since the representation of properties as mere primitives is always possible in OWL and other ontology languages.

In the second case, conflict resolution aims at preserving the original language profile at the expense of relaxing principal choice (ii), i.e., accepting that 'as much semantics as possible' is less than anticipated. This applies to conflicts emerging from what is called *design for scalability* in Section 2.1. Here, conflict resolution is likely to be a zero-sum game similar to that described in Section 3.2 (undecidability). The OWL 2 QL profile, for instance, is aimed at applications that use very large volumes of instance data, such as conventional relational database systems, and where query answering is the most important reasoning task. Violating this profile means accepting that query answering may no longer be implementable by rewriting queries into a standard relational query language [21].

In the first case, tool support is the same as that described in Section 3.2 (undecidability). In the second case, the OWL Species Classifier supports authors of OWL ontologies by listing which axioms violate which OWL species (see footnote 2). The OWL Species Classifier was also used to search through the 417 axioms of the CIDO ontology for COVID-19 [12] to check for profile violations, as illustrated next.

Example 3 *Let us assume that medical ontologies for information systems should not exceed the OWL 2 EL profile, considering scalability and compatibility with typical OBO Foundry ontologies and SNOMED CT. CIDO [12] is not in OWL 2 EL, however, since it has a class expression with a universal quantifier on the right-hand side; more specifically,* 'Yale New Haven Hospital SARS-CoV-2 assay' $\sqsubseteq \forall$'FDA EUA-authorized organization' *is one of the multiple axioms that violate the OWL 2 EL expressiveness restrictions in the* `cido.owl` *of 14 June 2020, and is also present in the* `cido-base.owl` *of 18 June 2020.*

3.4. Conflicts Manifesting Themselves in an Incoherent Ontology

Conflict resolution, in this case, aims at preserving a coherent ontology. Examples include ontological misspecifications at the axiom-level, such as disjoint ancestors, resulting in unsatisfiable classes (no. 5 in Table 1). Such conflicts manifest only when making deductions by a reasoner. In the simplest case, they are resolved by keeping some of the conflicting axioms and removing others in a way similar to that described in Section 3.2 (undecidability). In the example, either the disjointness axiom on the ancestors or the subclass axioms on the class may be kept, but not both.

State-of-the-art ODEs allow for making deductions. After running the OWL reasonser in Protégé 5.2, for instance, unsatisfiable classes and properties are displayed in red color. In order to find out what made them unsatisfiable, justifications can be computed using the respective plug-in. A *justification* is a set of axioms from an ontology that is sufficient for an entailment to hold [10]. In the case of unsatisfiable classes and properties, justifications can be computed specifically for entailments with `owl:Nothing` and `owl:bottomObjectProperty` on the right-hand side. In this way, the sources of incoherence can be identified.

3.5. Conflicting Modeling Styles

These conflicts arise from source 3 in Section 2.1. Resolving them aims at restructuring (parts of) an ontology such that correspondences with entities of a different ontology can be established. For instance, if the same notion is modeled in one ontology as a class and in another as an object property (see Figure 1), or even in both ways in the same ontology, and the ontology language does not permit heterogenous alignments, then either the object property has to be reified or the class has to be recast as an object property (see no. 7 in Table 1). Typical examples are object properties such as o1:married-to and o1:has-member and corresponding reifications as o2:Marriage and o2:Member, respectively. A concrete difference is illustrated in Example 4.

Example 4 *Consider again the CIDO ontology and now also the CODO ontology*[3] *for COVID-19:* codo:'laboratory test finding' $\equiv$ {positive, pending, negative}, *i.e., the outcomes are instances, whereas in CIDO, there is a* cido:'COVID-19 diagnosis' *class with three subclasses [negative/positive/presumptive positive] COVID-19 diagnosis. This is an example of class vs. instance modeling of the same idea.*

What is recorded in the conflict set depends on the case at hand; for the class vs. property example, these would be the respective axioms to match and the axioms they are used in, which may be found by using an NLP-based algorithm with POS tagging and stemming. Generally speaking, there are two different options of dealing with conflicting modeling styles. The first is to convert one modeling pattern into the other, and the second option is to match patterns by a set of axioms, rather than by a single bridging axiom, which is referred to as a heterogeneous TBox mapping [15].

The way how state-of-the-art ODEs deal with conflicting styles depends on the kind of conflict and the ODE. For instance, Protégé 5.2 restrict alignments to `owl:equivalentClass` statements. Simply put, it does not provide the necessary means to bridge a class and an object property. A joint outside option may be the DOL framework [4].

[3] `https://bioportal.bioontology.org/ontologies/CODO`

4. Case Study

The approach proposed here is tested against a case of epizootic disease outbreak in the Lemanic Arc (France, Switzerland) in 2006 [22]. To this end, case records of three occurrences of human-pathogenic avian influenza (H5N1) in wild birds were examined. The measures taken by the Swiss authorities to prevent the virus from infecting, in a first instance, domestic poultry consisted of establishing protection zones within a radius of at least 3 kilometers and surveillance zones within a radius of at least 10 kilometers. In these zones, regulations, such as 'poultry must be kept in the henhouse', were introduced. The Swiss authorities had to decide which municipalities to include in the protection zones and which in the surveillance zones (see Figure 2).

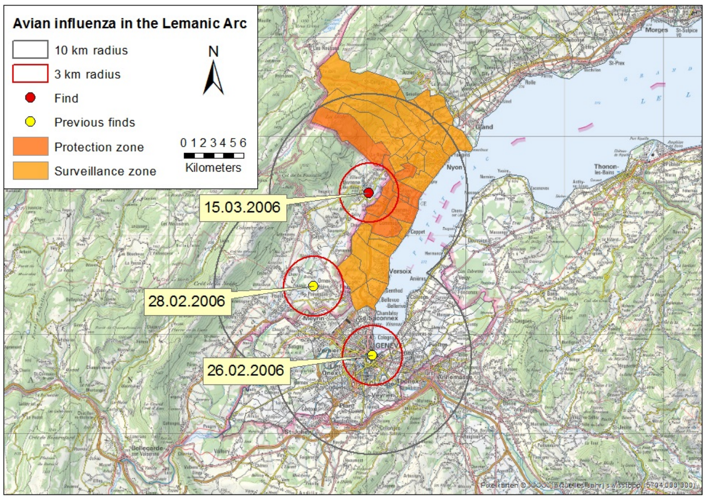

Figure 2. Avian influenza in the Lemanic Arc (adapted from [23]). National Map 1:200,000 © 2008 swisstopo

Assume the administrative division of Switzerland is represented in administrative ontology $\mathcal{O}_1$ and the finds of infected birds as well as protection and surveillance zones are represented in epidemiology ontology $\mathcal{O}_2$ (the ontologies can be downloaded from `https://www.envidat.ch/dataset/icbo2020`). In order to construct a query against a geodatabase to figure out which municipalities to include in which zones, the two ontologies need to be merged. Both are OWL 2 DL ontologies with an expressivity of $\mathcal{ALCRIF}$ and $\mathcal{SRIF}$, respectively. They have been implemented using Protégé 5.2 [19].

In order to represent the administrative division properly, every region occupied by a municipality is assigned to exactly one region occupied by a district. Accordingly, the object property partOf is functional in ontology $\mathcal{O}_1$. For the finds of infected birds in ontology $\mathcal{O}_2$, on the other hand, the same object property needs to be transitive: The (small) regions occupied by the finds are contained in the regions occupied by the protection zones. These are contained in the regions occupied by the surveillance zones. Merging the two ontologies, thus, results in a conflict which is reported by the following conflict set:

Ontology: $\mathcal{O}_1$	Ontology: $\mathcal{O}_2$
IRI: `appl:administrative` No.: 1.17 Axiom: $\mathsf{has\_2D} \sqcap \mathsf{has\_2D\_inv} \sqcap \mathsf{located\_in} \sqcap \mathsf{partOf} \sqsubseteq \bot$ Description: disjointness of roles Theory: n/a DL: $(\neg), \mathcal{R}$ No.: 1.22 Axiom: $\top \sqsubseteq (\leq 1\ \mathsf{partOf})$ Description: functionality Theory: n/a DL: $\mathcal{F}, \mathcal{Q}$ Inference: $(\mathcal{O}_1 + \mathcal{O}_2 - 2.32) \models \mathcal{O}_1'$	IRI: `appl:epidemiology` No.: 2.32 Axiom: $\mathsf{Tr}(\mathsf{partOf})$ Description: transitivity of roles Theory: M DL: $\mathcal{S}, \mathcal{R}$ Inference: $(\mathcal{O}_1 + \mathcal{O}_2 - 1.17 - 1.22) \models \mathcal{O}_2'$ Diff: $(\mathcal{O}_1'$ - $\mathcal{O}_2') = \emptyset$

The conflict is resolved by trading transitivity of `appl:epidemiology#partOf` for functionality of `appl:administrative#partOf` and disjointness of roles in the administrative ontology. Doing so affects less, but equally preferred, axioms than the other way round, namely, one axiom vs. seven axioms (please note: 'axiom' 1.17 is a shorthand notation for six individual axioms omitted due to space limitations). It loses exactly the same inferences (diff is empty) as trading in the opposite direction.

5. Conclusion

First steps towards a framework dealing with the various types of modeling conflicts through meaning negotiation and conflict resolution in a systematic way have been proposed. We introduced and specified the notions of meaning negotiation and conflict resolution, made clear what their components are, and took a first step towards conceiving a library of conflicts. The notion of conflict set was introduced as a minimal data structure in which the detected conflicts can be stored and upon which a software-mediated conflict resolution can operate. This approach was evaluated with an actual case of domain knowledge usage in the context of epizootic disease outbreak.

As has been shown, there is no single way of detecting and resolving conflicts. A number of common cases were described by distinguishing (1) ontology from language, (2) theory-level from axiom-level, and (3) upfront negotiation from resolution upon manifestation. While there are some tools and plugins that can assist with meaning negotiation and conflict resolution, no integrated support is currently provided. Future work includes refining the framework such as to specify software design requirements for automating conflict detection, based on which algorithms for conflict resolution can be devloped or fine-tuned, as well as establishing the conflict library in more concrete terms.

References

[1] Smith B, Ashburner M, Rosse C, Bard J, Bug, W et al. The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration . Nature Biotechnology 2007 Nov;25(11):12515.

[2] Suárez-Figueroa MC, Gómez-Pérez A, Motta E, Gangemi A, editors. Ontology Engineering in a Networked World. Berlin Heidelberg: Springer; 2012.

[3] Keet CM, Kutz O. Orchestrating a Network of Mereo(topo)logical Theories. Proceedings of the 9th International Conference on Knowledge Capture; 2017 Dec 4-6; Austin, Texas, USA. New York (NY, US): ACM; 2017. p 11:1-11:8.

[4] Object Management Group. Distributed Ontology, Model, and Specification Language, v1; 2018. Available from: `http://www.omg.org/spec/DOL/`.

[5] Fernández-López M, Gómez-Pérez A, Suárez-Figueroa MC. Selecting and Customizing a Mereology Ontology for its Reuse in a Pharmaceutical Product Ontology. In: Eschenbach C, Grüninger M, editors. Formal Ontology in Information Systems. Amsterdam (The Netherlands): IOS Press; 2008. p. 181–94.

[6] Khan Z, Keet CM. ONSET: Automated Foundational Ontology Selection and Explanation. In: ten Teije A et al., editors. Proceedings of the 18th International Conference on Knowledge Engineering and Knowledge Management; 2012 Oct 8-12; Galway, Ireland. LNAI, vol. 7603. Berlin Heidelberg: Springer; 2012. p. 237-51.

[7] Cisse PA, Camara G, Dembele JM, Lo M. An Ontological Model for the Annotation of Infectious Disease Simulation Models. In: Bassioni G, Kebe CMF, Gueye A, Ndiaye A, editors. Innovations and Interdisciplinary Solutions for Underserved Areas. LNICST, vol. 296. Cham (Switzerland): Springer; 2019. p. 82–91.

[8] The Gene Ontology Consortium. The Gene Ontology Resource: 20 years and still GOing strong. Nucleic Acids Research 2019 Jan;47(D1):D330–8.

[9] Haendel MA, McMurry JA, Relevo R, Mungall CJ, Robinson PN, Chute CG. A Census of Disease Ontologies. Annual Review of Biomedical Data Science 2018;1:305–31.

[10] Horridge M, Parsia B, Sattler U. Laconic and Precise Justifications in OWL. In: Sheth A. et al., editors. Proceedings of the 7th International Semantic Web Conference; 2008 October 26-30; Karlsruhe, Germany. LNCS, vol. 5318. Berlin Heidelberg: Springer; 2008. p. 323-38.

[11] Matentzoglu N, Vigo M, Jay C, Stevens R. Inference Inspector: Improving the verification of ontology authoring actions. Web Semantics: Science, Services and Agents on the World Wide Web 2018 Mar;49:1-15.

[12] He Y, Yu H, Ong E, Wang Y, Liu Y, Huffman A, Huang H, Beverley J, Hur J, Yang X, Chen L, Omenn GS, Athey B, Smith B. CIDO, a community-based ontology for coronavirus disease knowledge and data integration, sharing, and analysis. Scientific Data 2020;7:181.

[13] Varzi AC. Spatial Reasoning and Ontology: Parts, Wholes, and Locations. In: Aiello M, Pratt-Hartmann I, van Benthem J, editors. Handbook of Spatial Logics. Berlin Heidelberg: Springer; 2007. p. 945-1038.

[14] Motik B, Parsia B. OWL 2 Web Ontology Language: Structural Specification and Functional-Style Syntax (Second Edition). W3C Recommendation 11 December 2012. https://www.w3.org/TR/2012/REC-owl2-syntax-20121211/

[15] Fillottrani PR, Keet CM. Patterns for Heterogeneous TBox Mappings to Bridge Different Modelling Decisions. In: Blomqvist E et al., editors. The Semantic Web. Part I. LNCS, vol. 10249. Heidelberg: Springer; 2017. p. 371–86.

[16] Davies K, Keet CM, Lawrynowicz A. More Effective Ontology Authoring with Test-Driven Development and the TDDonto2 Tool. International Journal on Artificial Intelligence Tools 2019;28(7):1950023.

[17] Flouris G, Huang Z, Pan JZ, Plexousakis D, Wache H. Inconsistencies, Negations and Changes in Ontologies. In: Cohn A, editor. Proceedings of the 21st National Conference on Artificial Intelligence, vol. 2; 2006 July 1620; Boston, Massachusetts. Boston (MA): AAAI Press; 2006. p. 1295-300.

[18] Horridge M, Bechhofer S. The OWL API: A java API for OWL ontologies. Semantic Web 2011;2(1):11-21.

[19] Musen MA. The Protégé project: A look back and a look forward. AI Matters. Association of Computing Machinery Specific Interest Group in Artificial Intelligence 2015 Jun;1(4):4–12.

[20] Gonçalves RS, Parsia B, Sattler U. Ecco: A Hybrid Diff Tool for OWL 2 ontologies. In: Klinov P, Horridge M, editors. Proceedings of OWL: Experiences and Directions Workshop; 2012 May 27-28; Heraklion, Crete, Greece. Aachen: CEUR Workshop Proceedings; 2012. p. 8.

[21] Motik B, Grau BC, Horrocks I, Wu Z, Fokoue A, Lutz C. OWL 2 Web Ontology Language Profiles. W3C Recommendation 27 October 2009. http://www.w3.org/TR/2009/REC-owl2-profiles-20091027/

[22] Grütter R. A framework for assisted proximity analysis in feature data. J Geogr Syst. 2019;21:367-94.

[23] Perler L. Geflügelgrippe: Ursprung – Entwicklung – Ausblick. Eidgenössisches Volkswirtschaftsdepartement EVD, Bundesamt für Veterinärwesen BVET; 2007. http://docplayer.org/34156871-Gefluegelgrippe-ursprung-entwicklung-ausblick.html