Resolving and avoiding design conflicts in ontology development and deployment

Maria Keet

Department of Computer Science, University of Cape Town, South Africa
mkeet@cs.uct.ac.za

ISCBaCadeMy Webinar series (bio-ontologies), 21 September 2021
Outline

1. Context and motivation
2. Resolving conflicts
3. Implementation trade-offs
4. Conclusions
Outline

1. Context and motivation
2. Resolving conflicts
3. Implementation trade-offs
4. Conclusions
An ontology

Simplified graphical rendering of a fragment of one:

http://geneontology.org/docs/ontology-documentation/
In an ODE...
Context and motivation

Resolving conflicts

Implementation trade-offs

Conclusions

... happenings behind the GUI ...

SubClassOf(awo:lion awo:animal)
SubClassOf(awo:lion ObjectSomeValuesFrom(awo:eats awo:Impala))
SubClassOf(awo:lion ObjectAllValuesFrom(awo:eats awo:herbivore))
... and underlying that serialisation
Introduction

- Ontologies
  - For their own sake
  - For communication among humans
  - Used for many different ontology-driven information systems (database integration and linking, recommender systems, NLP, textbook annotation and search, question generation, Q&A systems, etc.)
Introduction

- Ontologies
  - For their own sake
  - For communication among humans
  - Used for many different ontology-driven information systems (database integration and linking, recommender systems, NLP, textbook annotation and search, question generation, Q&A systems, etc.)

⇒ Someone has to build them, *somehow*
Typical stages of macro-level methodologies

**Ontology management** (scheduling, controlling, quality assurance)

- **Feasibility study** (problems, opportunities, potential solutions, economic feasibility)

**Ontology development and support**

- **Domain Analysis** (motivating scenarios, competency questions, existing solutions)
- **Conceptualisation** (of the model, integration and extension of existing solutions)
- **Implementation** (ontology authoring in a logic-based representation language)

**Ontology use**

- **Maintenance** (adapting the ontology to new requirements)
- **Use** (ontology-based search, integration, negotiation)
Scenarios for building Ontology Networks (NEON methodology)
Ontology Summit 2013's lifecycle model

http://ontolog.cim3.net/cgi-bin/wiki.pl?OntologySummit2013_Communique
Ontology Summit 2013's lifecycle model

http://ontolog.cim3.net/cgi-bin/wiki.pl?OntologySummit2013_Communique
More cycles within a cycle (for “ontology design”)

CQ added, template filled, or axiom written

Prior feasibility study, architecture, language decisions, ontology reuse decisions, etc etc, CQ specification

Ontology lifecycle

Deployment, documentation, etc.

1. select scenario
2. domain axiom for TDD test
3. TDD test expected to fail
4. update ontology
5. classify ontology; no contradictions
6. TDD test expected to pass
7. refactor
8. regression testing

Keet CM, Ławrynowicz A. Test-Driven Development of Ontologies. ESWC’16.

Ontology development at the ‘micro-level’ level (cf. macro)

- We need to get those axioms into the ontology
Ontology development at the ‘micro-level’ level (cf. macro)

- We need to get those axioms into the ontology
- The actual modelling, or *ontology authoring*, using **micro-level** guidelines, methods, and tools
  - Methods, such as reverse engineering and text mining to start, OntoClean and OntoParts to improve an ontology’s quality
  - Tools to model, to reason, to debug, to integrate, to link to data
Ontology development at the ‘micro-level’ level (cf. macro)

- We need to get those axioms into the ontology
- The actual modelling, or *ontology authoring*, using *micro-level* guidelines, methods, and tools
  - Methods, such as reverse engineering and text mining to start, OntoClean and *OntoPARTS* to improve an ontology’s quality
  - Tools to model, to reason, to debug, to integrate, to link to data

⇒ But what if you’re not sure of the axioms yet? Or it leads to a conflict and possibly also an ‘incoherent’ ontology?
Examples

- BFO does not have Stuff (e.g., mucus, cytosol, water). Deny its existence? Add it as a not quite fitting subclass somewhere? Create/reuse a core ontology?
- Virus ⊑ Organism vs. Virus ⊑ acellular structure
BFO does not have *Stuff* (e.g., mucus, cytosol, water). Deny its existence? Add it as a not quite fitting subclass somewhere? Create/reuse a core ontology?

- Virus $\sqsubseteq$ Organism vs. Virus $\sqsubseteq$ acellular structure
- A class Transformation or a relationship transforms?
- proper parthood is transitive, irreflexive, and asymmetric. Choose one? Give up on decidable reasoning?
The real use case (thanks to Rolf Grütter)

- Epizootic disease outbreak in the Lemanic Arc (France, Switzerland) in 2006
- Human-pathogenic avian influenza H5N1, modelling & data
- Swiss authorities set up protection zones within a radius of 3km, surveillance zones within a radius of 10km.
- Rules to apply; e.g., *poultry must be kept in the henhouse*
- Need to decide which municipalities to include in the protection zones and which in the surveillance zones

Geflügelgrippe: Ursprung – Entwicklung – Ausblick. EVD, Bundesamt für Veterinärwesen BVET. (presentation)
The real use case (thanks to Rolf Grütter)

- How to make those decisions better and faster for a next time? (we’re in mid 2019 then...)
The real use case (thanks to Rolf Grütter)

- How to make those decisions better and faster for a next time? (we’re in mid 2019 then...)
- Two ontologies—epidemiology (finds, etc) and administrative (generic, with Municipality etc.)—and a geodatabase
- Municipality in exactly one region etc.
- The (small) region of the find is contained in the region(s) occupied by the protection zones that are contained in the regions occupied by the surveillance zones
The real use case (thanks to Rolf Grütter)

- How to make those decisions better and faster for a next time? (we’re in mid 2019 then...)
- Two ontologies—epidemiology (finds, etc) and administrative (generic, with Municipality etc.)—and a geodatabase
- Municipality in exactly one region etc.
- The (small) region of the find is contained in the region(s) occupied by the protection zones that are contained in the regions occupied by the surveillance zones

Reasoner: .... non-simple .... beyond OWL 2 DL!
Outline

1. Context and motivation
2. Resolving conflicts
3. Implementation trade-offs
4. Conclusions
How to manage such differences?

- Identify type of conflicts that can arise
- Determine how to preempt or to detect them
- Assess options what to do with it when a conflict arises
- Specify a mechanism to keep track of these three aspects
- Devise a way to make this easy to do and document choice
Note: meaning negotiation vs conflict resolution

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. It necessarily involves a compromise and making it work requires reengineering something in at least one of the ontologies or as a whole.
Sample scenario to detect and resolve conflicts

**Reuse scenario**
Plan: import ontology O2 into ontology O1

**Tool feedback** *(example)*
1. O1+2 violates OWL 2 DL language
2. O1 reifies relations but O2 does not;

**Examine sources of conflict** *(example)*
1. o1:part-of = o2:part-of, but o1:part-of is transitive and o2:part-of is used in a qualified cardinality constraint
2. This concerns o1:Vaccination and o2:vaccinates

**Resolve conflicts** *(choices made for example)*
1. Agree to keep both constraints and thus select a more expressive ontology language.
2. Choose O1’s reification approach in line with its modelling style

**Implement resolution**
1. No further action needed
2. Remodel o2:vaccinates axioms accordingly
3. Import O2’ into O1
What are the key sources of conflicts?

- Ontological differences between established theories
  - DOLCE vs BFO
- Ontological differences at the axiom-level
  - parthood antisymmetric or not? [Cotnoir(2010)]
What are the key sources of conflicts?

- Ontological differences between established theories
  - DOLCE vs BFO
- Ontological differences at the axiom-level
  - parthood antisymmetric or not? [Cotnoir(2010)]
- Different modelling styles
  - foundational ontology-inspired or conceptual model-influenced
    [Fillottrani and Keet(2017), Fillottrani and Keet(2019)]
What are the key sources of conflicts?

- Ontological differences between established theories
  - DOLCE vs BFO
- Ontological differences at the axiom-level
  - parthood antisymmetric or not? [Cotnoir(2010)]
- Different modelling styles
  - foundational ontology-inspired or conceptual model-influenced [Fillottrani and Keet(2017), Fillottrani and Keet(2019)]
- Logic limitations causing conflicts for an ontology, affecting the software ecosystem
  - OWL only or DOL [DOL(2018)] that can do FOL and HOL
- Logic limitations by design, for the purpose of scalability
  - OWL 2 EL vs. OWL 2 DL [Motik et al.(2009)]
- Certain deductions (excluding modelling mistakes) that manifest after adding the axioms, during TDD, or upon ontology matching attempts.
  - disjointness declared among some ancestor
Illustration of language profile conflicts

**Requirement** “The COVID-relevant medical ontology for information systems should not exceed the OWL 2 EL profile (compatibility with OBO, SNOMED CT, scalability)”
Illustration of language profile conflicts

Requirement “The COVID-relevant medical ontology for information systems should not exceed the OWL 2 EL profile (compatibility with OBO, SNOMED CT, scalability)”

- CIDO ontology for COVID-19 [He et al.(2020)] is not in OWL 2 EL
- Class expression with a universal quantifier on rhs; a.o.:
  ‘Yale New Haven Hospital SARS-CoV-2 assay’ ⊑
  ∀‘EUA-authorized use at’.‘FDA EUA-authorized organization’

- Need a tool to find violating axioms: the OWL Classifier
Section of the OWL classifier, having detected that CID0_0000020 (i.e., Yale New Haven Hospital SARS-CoV-2 assay) violated OWL 2 EL.

OWL Classifier https://github.com/muhummadPatel/OWL_Classifier
### ‘Library’ of common conflicts (selection – 1/2)

<table>
<thead>
<tr>
<th>No.</th>
<th>Conflict</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>foundational</td>
<td><em>Conflicting theories at the top-level</em> ontologies adhere to conflicting theories</td>
<td>BFO, DOLCE, GFO, SUMO, UFO, YAMATO</td>
</tr>
<tr>
<td>2</td>
<td>mereological</td>
<td>conflicting mereological theories</td>
<td>with Atom or not, weak vs. strong supplementation</td>
</tr>
<tr>
<td>3</td>
<td>topological</td>
<td>conflicting topological theories</td>
<td>region connection calculus on non-simply connected regions</td>
</tr>
<tr>
<td>4</td>
<td>building blocks</td>
<td>different ontological commitments embedded in the language</td>
<td>whether roles are part of the fundamental furniture of the universe, 3D + time vs. 4D ‘worms’</td>
</tr>
</tbody>
</table>

...
Some of this is ‘easy’ to figure out

- Delegate the choice: use an existing foundational ontology

- Delegate the choice: use an existing ontology language
Some of this is ‘easy’ to figure out

- Delegate the choice: use an existing foundational ontology
  - How to choose an existing foundational ontology?

- Delegate the choice: use an existing ontology language
  - How to choose an existing language?
Some of this is ‘easy’ to figure out

- Delegate the choice: use an existing foundational ontology
  - How to choose an existing foundational ontology?
  - What if it conflicts with the rest of the system?

- Delegate the choice: use an existing ontology language
  - How to choose an existing language?
  - What if it conflicts with the rest of the system?
Choose an existing foundational ontology

Choose an existing foundational ontology

Universals vs. Particulars

Universals are objects that can have instances. Particulars are objects that cannot have instances. e.g. Dog is a universal while 'Bruno' the dog is a particular which cannot be instantiated.

Choose an existing foundational ontology

Choose an existing foundational ontology

Choose an existing foundational ontology

1. BFO is an ontology of Universals.
2. BFO is Realist in nature.
3. The OBO foundry has recommended that ontologies registered on the OBO Foundry should use BFO.
4. BFO is freely available.
5. BFO has been used in Life Sciences ontologies.

Consider language: simple or complicated

- Simple purpose-oriented guidance:

  - Is reasoning required?
    - Yes
    - Expressivity is important?
      - No
        - Only data annotation?
          - Use OBO or OWL 2 EL
      - Yes
        - Text annotation?
          - Use SKOS, OBO, or OWL 2 EL
        - Expressivity is important?
          - No
            - Only data annotation?
              - Use OWL 2 QL
          - Yes
            - large ABox?
              - Use OWL 2 QL
            - Use any FOL, extension thereof, or higher order logic, e.g. Common Logic, DLRus
      - Decidability is important?
        - No
          - Use OWL (2) DL
        - Yes
          - Use OWL 2 QL

- Use ‘translators’ (e.g., SKOS → OWL, OBO → OWL, OWL → FOL) or DOL as ‘glue’

Consider language: simple or complicated

- Simple purpose-oriented guidance:

  - Is reasoning required? No: Only data annotation? Use $\text{SKOS, OBO, or OWL 2 DL}$ or $\text{OWL 2 QL}$
  - Yes: Expressivity is important? Use $\text{OWL 2 DL}$
  - Decidability is important? Use $\text{Any FOL, extension thereof, or higher order logic, e.g. Common Logic, DLRus}$

- Use ‘translators’ (e.g., SKOS $\rightarrow$ OWL, OBO $\rightarrow$ OWL, OWL $\rightarrow$ FOL) or DOL as ‘glue’

- Complicated: design your own!

Well-known fundamental language conflicts

- Attributes/data properties (OWL, UML) or not (OBO)
- Parthood as primitive (originally so in OBO) or not (OWL)
- Some separation of language from ‘semantic layer’ (OBO naming scheme of entities vs OWL, CL etc.)
- 3D+time vs. 4D (in theory at least; time is costly)

Fillottrani PR, Keet CM. An analysis of commitments in ontology language design. FOIS 2020.
### ‘Library’ of common conflicts (selection – 2/2)

<table>
<thead>
<tr>
<th>9</th>
<th>modeling style</th>
<th>applied vs. foundational</th>
<th>whether there are data property axioms, alike height between Person and xsd:decimal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>class vs. object property</td>
<td>Infection vs. infected-by</td>
</tr>
<tr>
<td></td>
<td></td>
<td>subsuming roles vs. roles inhering in objects</td>
<td>doctor is-a person vs. doctor inheres-in person</td>
</tr>
<tr>
<td>10</td>
<td>language</td>
<td>cultural-linguistic and labeling differences, such as preferred/alt labels, orthography, language variants</td>
<td>population immunity vs herd immunity, color vs colour, and non-1:1 mappings (e.g., ‘river’ vs fleuve and rivière)</td>
</tr>
</tbody>
</table>
Modelling style example (1/2)

Requirement: Integrate the CIDO and CODO COVID-19 ontologies
Modelling style example (1/2)

Requirement: Integrate the CIDO and CODO COVID-19 ontologies

- CODO: laboratory testfinding $\equiv \{\text{positive, pending, negative}\}$
- CIDO: positive COVID-19 diagnosis $\sqsubseteq$ COVID-19 diagnosis, presumptive positive COVID-19 diagnosis $\sqsubseteq$ COVID-19 diagnosis, and negative COVID-19 diagnosis $\sqsubseteq$ COVID-19 diagnosis
Modelling style example 2/2

- (Naming issue, or also ontological: finding (some fact) vs. diagnosis (conclusion drawn from the fact) — when taken in context, intention is the same)

⇒ Class vs. instance representations of the same idea
Modelling style example 2/2

- (Naming issue, or also ontological: finding (some fact) vs. diagnosis (conclusion drawn from the fact) — when taken in context, intention is the same)

⇒ Class vs. instance representations of the same idea

- Solution options:
  - Change CODO to use CIDO’s style
(Naming issue, or also ontological: finding (some fact) vs. diagnosis (conclusion drawn from the fact) — when taken in context, intention is the same)

⇒ Class vs. instance representations of the same idea

Solution options:
- Change CODO to use CIDO’s style
- Change CIDO to use CODO’s style
Modelling style example 2/2

- (Naming issue, or also ontological: finding (some fact) vs. diagnosis (conclusion drawn from the fact) — when taken in context, intention is the same)

  ⇒ Class vs. instance representations of the same idea

- Solution options:
  - Change CODO to use CIDO’s style
  - Change CIDO to use CODO’s style
  - A joint outside option; e.g.: use attribute + values instead
Record such information: the conflict set

Conflict set grammar for recording individual conflict sets in or between ontologies

<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> | "none"
<diff> ::= difference between the inferred axioms sets of the two ontologies

(production rules of most terminals are omitted)
Ontology: \( O_1 
\\text{IRI: appl:admin} \\
No.: 1.17 \\
Axiom: has_2D \sqcap \\
\text{has}_2D\_\text{inv} \sqcap \text{located}_\text{in} \sqcap \\
\text{partOf} \sqsubseteq \bot \\
\text{Description: disjointness} \\
\text{Theory: n/a} \\
\text{DL: } (\to), \mathcal{R} \\
\text{No.: 1.22} \\
\text{Axiom: } \top \sqsubseteq (\leq 1 \text{ partOf}) \\
\text{Description: functionality} \\
\text{Theory: n/a} \\
\text{DL: } \mathcal{F}, \mathcal{Q} \\

\text{Inference } O_1: (O_1 \sqcup O_2 \sqcup (\text{appl:admin}\#\text{partOf} \equiv \text{appl:epidemiology}\#\text{partOf})) \sqcap \neg 2.32 \models O'_1 \\

\text{Inference } O_2: (O_1 \sqcup O_2 \sqcup (\text{appl:admin}\#\text{partOf} \equiv \text{appl:epidemiology}\#\text{partOf})) \sqcap \neg (1.17 \sqcup 1.22) \models O'_2 \\

\text{Diff: } O'_1 \sqcap \neg O'_2 \sqsubseteq \bot
<table>
<thead>
<tr>
<th>Context and motivation</th>
<th>Resolving conflicts</th>
<th>Implementation trade-offs</th>
<th>Conclusions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Ontology</strong>: $O_1$</td>
<td><strong>Ontology</strong>: $O_2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>IRI</strong>: appl:admin</td>
<td><strong>IRI</strong>: appl:epidemiology</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>No.</strong>: 1.17</td>
<td><strong>No.</strong>: 2.32</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Axiom</strong>: has_2D $\sqsubseteq$ has_2D_inv $\sqcap$ located_in $\sqsubseteq$ partOf $\sqsubseteq \bot$</td>
<td><strong>Axiom</strong>: Tr(partOf)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Description</strong>: disjointness</td>
<td><strong>Description</strong>: transitivity</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Theory</strong>: n/a</td>
<td><strong>Theory</strong>: M</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>DL</strong>: $(\rightarrow), \mathcal{R}$</td>
<td><strong>DL</strong>: $S, \mathcal{R}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>No.</strong>: 1.22</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Axiom</strong>: $\top \sqsubseteq (\leq 1 \text{ partOf})$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Description</strong>: functionality</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Theory</strong>: n/a</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>DL</strong>: $F, Q$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Inference</strong> $O_1$: $(O_1 \sqcup O_2 \sqcup (\text{appl:admin#partOf} \equiv \text{appl:epidemiology#partOf})) \sqcap \neg 2.32 \models O'_1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Inference</strong> $O_2$: $(O_1 \sqcup O_2 \sqcup (\text{appl:admin#partOf} \equiv \text{appl:epidemiology#partOf})) \sqcap \neg(1.17 \sqcup 1.22) \models O'_2$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Diff</strong>: $O'_1 \sqcap \neg O'_2 \sqsubseteq \bot$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Ontology: $\mathcal{O}_1$

IRI: `appl:admin`

No.: 1.17

Axiom: `has_2D ∩`
`has_2D_inv ∩ located_in ∩ partOf ⊑ ⊥`

Description: disjointness
Theory: n/a
DL: $(\neg), \mathcal{R}$

Ontology: $\mathcal{O}_2$

IRI: `appl:epidemiology`

No.: 2.32

Axiom: `Tr(partOf)`

Description: transitivity
Theory: M
DL: $S, \mathcal{R}$

No.: 1.22

Axiom: `⊤ ⊑ (≤ 1 partOf)`

Description: functionality
Theory: n/a
DL: $F, Q$

Inference $\mathcal{O}_1$: $(\mathcal{O}_1 ⊔ \mathcal{O}_2 ⊔ (\text{appl:admin#partOf} \equiv \text{appl:epidemiology#partOf})) \dashv \neg.2.32 \models \mathcal{O}'_1$

Inference $\mathcal{O}_2$: $(\mathcal{O}_1 ⊔ \mathcal{O}_2 ⊔ (\text{appl:admin#partOf} \equiv \text{appl:epidemiology#partOf})) \dashv \neg(1.17 ⊔ 1.22) \models \mathcal{O}'_2$

Diff: $\mathcal{O}'_1 \dashv \neg \mathcal{O}'_2 \models ⊥$
<table>
<thead>
<tr>
<th>Context and motivation</th>
<th>Resolving conflicts</th>
<th>Implementation trade-offs</th>
<th>Conclusions</th>
</tr>
</thead>
</table>
| **Ontology:** $\mathcal{O}_1$  
IRI: appl:admin  
No.: 1.17  
Axiom: has\_2D $\sqcap$  
has\_2D\_inv $\sqcap$ located\_in $\sqcap$  
partOf $\sqsubseteq \bot$  
Description: disjointness  
Theory: n/a  
DL: $(\neg), \mathcal{R}$ |
| **Ontology:** $\mathcal{O}_2$  
IRI: appl:epidemiology  
No.: 2.32  
Axiom: Tr(partOf)  
partOf $\sqsubseteq \bot$  
Description: transitivity  
Theory: M  
DL: $S, \mathcal{R}$ |

| No.: 1.22  
Axiom: $\top \sqsubseteq (\leq 1 \text{ partOf})$  
Description: functionality  
Theory: n/a  
DL: $\mathcal{F}, Q$ |

Inference $\mathcal{O}_1$: $(\mathcal{O}_1 \sqcup \mathcal{O}_2 \sqcup (\text{appl:admin}\#\text{partOf} \equiv \text{appl:epidemiology}\#\text{partOf})) \sqcap \neg 2.32 \models \mathcal{O}_1'$

Inference $\mathcal{O}_2$: $(\mathcal{O}_1 \sqcup \mathcal{O}_2 \sqcup (\text{appl:admin}\#\text{partOf} \equiv \text{appl:epidemiology}\#\text{partOf})) \sqcap \neg (1.17 \sqcup 1.22) \models \mathcal{O}_2'$

Diff: $\mathcal{O}_1' \sqcap \neg \mathcal{O}_2' \sqsubseteq \bot$
Cognitive walk-throughs for the Avian influenza Case Study

 Administrative ontology O1
 Functional(partOf)
 Epidemiology ontology O2
 Transitive(partOf)

 alignment: O1:partOf = O2:partOf

1. System: O' is beyond OWL 2 DL. This is because partOf is declared transitive and functional. Is the alignment O1:partOf = O2:partOf your intention?
   - Modeller: Yes

2. System: Noted. Do you want to remain within the maximum expressiveness of OWL 2 DL?
   - Modeller: Yes

3. System: Noted. In that case, your main, least disruptive, options to resolve the conflict are as follows.
   a) Remove alignment axiom, remove epidemiology.
   b) Keep epidemiology imported, but remove the transitive axiom.
   c) Keep epidemiology imported, but remove the functional axiom.
   Choose one option.
   - Modeller: a

4. System: Noted. Doing so will remove all other alignments as well. Are you sure? Type yes to implement or ask for details.
   - Modeller: What else will be removed?

5. System: The following 2 axioms with epidemiology will be lost:
   i) O1:has_2D = O2:has_2D
   ii) O1:located_in = O2:located_in
   - Modeller: Ok, delete

6. System: Noted. Your decision has been implemented. No other conflicts were detected. This concludes the session.

A

B

System: O' is beyond OWL 2 DL. Do you want to know why, or just delete the violating axiom(s)?
   - Modeller: Explain why

System: because partOf is declared transitive (from O2) but also functional (from O1), yet you declared O1:partOf = O2:partOf. This violates the language specification for OWL 2 DL. Would you like to keep them or change something?
   - Modeller: Keep them

System: Noted. Keeping them leaves three options:
   a) Stay within the OWL tooling infrastructure.
   b) Change to the DOL framework, and use the Hets reasoning system.
   c) Change to Common Logic.
   Choose one option.
   - Modeller: a

System: Noted. Doing so means that you will lose automated reasoning features. Are you sure? Type yes to continue or ask for details.
   - Modeller: What are the consequences of switching to CL?

System: The ontology will be more precise. Computing consistency likely will take longer, and it may not terminate. Ok?
   - Modeller: No; stay with OWL

System: Noted. Your decision has been implemented. No other conflicts were detected. This concludes the session.
Minimal system requirements

- A conflict resolution workflow management system, be it a Question Answering system or another strategy that avails of a knowledge-to-text controlled language, canned text, a decision tree, and two data structures (the conflict set and the resolution options);
- Algorithms to populate the conflict set, which may avail of new wrappers for existing OWL tools to recast their computation and outputs as detection and conflict resolution functionalities;
- End-user usable DOL and CL tools;
- Software support for the language annotation models and extant assessments on modelling style and language conflicts.
Connecting the knowledge to the data

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Queries for decision-making formulate queries using the knowledge graph to retrieve data.

Implementation the actual information system that stores and manipulates the data.
Knowledge-to-Data Pipeline options

Knowledge base $K$ with instances $D$

“Knowledge with data”

“Knowledge mapping data”

“Data transformation knowledge”

“Data with Knowledge”

DB-oriented

Al-oriented

Fillottrani, P.R., Keet, C.M. KnowID: An architecture for efficient Knowledge-driven Information and Data access. Data Intelligence, 2020, 2(4): 487-512.
### Key distinguishing features of varying computational cost

<table>
<thead>
<tr>
<th>Feature</th>
<th>$K \circ D$</th>
<th>$K \leftrightarrow D$</th>
<th>$D \ni K$</th>
<th>$D \circ aK$</th>
</tr>
</thead>
<tbody>
<tr>
<td>World</td>
<td>OWA</td>
<td>OWA + CWA</td>
<td>CWA</td>
<td>CWA</td>
</tr>
<tr>
<td>Language for $K$</td>
<td>OWL</td>
<td>OWL</td>
<td>relational, DL</td>
<td>relational</td>
</tr>
<tr>
<td>Language for $D$</td>
<td>OWL</td>
<td>relational</td>
<td>relational</td>
<td>relational</td>
</tr>
<tr>
<td>Query language</td>
<td>SPARQL</td>
<td>SPARQL + SQL (fragment)</td>
<td>SQLP</td>
<td>SQL</td>
</tr>
<tr>
<td>Automated reasoning</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>depends on system</td>
</tr>
<tr>
<td>Reasoning w.r.t. data</td>
<td>no separate approach</td>
<td>query rewriting</td>
<td>data completion</td>
<td>data completion</td>
</tr>
<tr>
<td>Mapping layer</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Transformations</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Entity recasting</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Syntactic sugar</td>
<td>available</td>
<td>available</td>
<td>possible</td>
<td>possible</td>
</tr>
</tbody>
</table>
“Knowledge mapping data”: OBDA

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Queries for decision-making formulate queries using the knowledge graph to retrieve data.

Implementation the actual information system that stores and manipulates the data.

Database

C++ application
“Knowledge mapping data”: OBDA

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Mapping layer links each entity to a query over the data source(s).

Implementation the actual information system that stores and manipulates the data.
“Knowledge mapping data”: OBDA

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Mapping layer links each entity to a query over the data source(s).

Implementation the actual information system that stores and manipulates the data.

```
Flower
->
SELECT flowers.id
   FROM flowers
UNION
SELECT blom.name
   FROM blom
...
```

```
Flower ⊑ height.Int
Flower ⊑ ≤ 1 height.Int
Flower ⊑ ≥ colour.AnyType
Flower ⊑ = 1 id.Int
id.int ⊑ Flower
...
```
“Knowledge mapping data”: OBDA

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications

Mapping layer links each entity to a query over the data source(s)

Implementation the actual information system that stores and manipulates the data
"Knowledge mapping data": OBDA

Ontology (or controlled vocab, kg)
provides the common vocabulary
and constraints that hold across
the applications

Mapping layer
links each entity
to a query over the
data source(s)

Implementation
the actual information
system that stores and
manipulates the data
“Knowledge mapping data”: OBDA

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Mapping layer links each entity to a query over the data source(s).

Implementation the actual information system that stores and manipulates the data.

End-user query “give me all red flowers” just click relevant elements in the diagram.
The WONDER System with the early version

- Horizontal Gene Transfer (HGT) database [Garcia-Vallvé et al.(2003)]
- Reverse engineer the conceptual data model
- Formalise it in OWL 2 QL
- Create mappings
- Create (web-based) interface for browsing, querying, and answering as front-end to OBDA back-end

“Knowledge mapping data”: OBDA example in genomics
“Knowledge mapping data”: OBDA example in genomics

```xml
<mapping id="Ending codons">
    <CQ string="EndingCodon(getEndingCodon($CODON)), TripletCode(getEndingCodon($CODON),$CODON)"/>
    <SQLQuery string="SELECT CODON FROM ENDINGCODONS"/>
</mapping>

<mapping id="OrganismHasCodon">
    <CQ string="OHCHasOrganism(getOrganismHasCodon($BUNDLEID),getOrganism($ID)),
        OHCHasCodon(getOrganismHasCodon($BUNDLEID),getCodon($CODON)),
        OrganismHasCodon(getOrganismHasCodon($BUNDLEID)),
        CodonValueOrg(getOrganismHasCodon($BUNDLEID),$CODONVALUE),
        CodonSD(getOrganismHasCodon($BUNDLEID),$CODONSD),
        RSCUorg(getOrganismHasCodon($BUNDLEID),$RSCU),
        SDRSUCU(getOrganismHasCodon($BUNDLEID),$SDRSUCU)"/>
    <SQLQuery string="SELECT ID, CODON, CODONVALUE, CODONSD, BUNDLEID, RSCU, SDRSUCU FROM
        ORGANISMHASCODON"/>
</mapping>

<mapping id="GeneHasCodon">
    <CQ string="GHCHasGene(getGeneHasCodon($BUNDLEID),getGene($ID)),
        GHCHasCodon(getGeneHasCodon($BUNDLEID),getCodon($CODON)),
        GeneHasCodon(getGeneHasCodon($BUNDLEID)),
        CodonValueGene(getGeneHasCodon($BUNDLEID),$CODONVALUE),
        RSCUgene(getGeneHasCodon($BUNDLEID),$RSCU)"/>
    <SQLQuery string="SELECT ID, CODON, CODONVALUE, BUNDLEID, RSCU FROM GENEHASCODON"/>
</mapping>
```

...
"Knowledge mapping data": OBDA example in genomics

<table>
<thead>
<tr>
<th>Construct</th>
<th>Graphical Element</th>
<th>Semantic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class</td>
<td>![Class Diagram]</td>
<td>$C \sqsubseteq \top$</td>
</tr>
<tr>
<td>Object Property</td>
<td>![Object Property Diagram]</td>
<td>$\exists P \sqsubseteq C$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\exists P^{-} \sqsubseteq D$</td>
</tr>
<tr>
<td>Data Property</td>
<td>![Data Property Diagram]</td>
<td>$\delta(A) \sqsubseteq C$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\rho(A) \sqsubseteq \top_d$</td>
</tr>
<tr>
<td>SubClass Relationship</td>
<td>![SubClass Relationship Diagram]</td>
<td>$C \sqsubseteq D$</td>
</tr>
</tbody>
</table>
“Knowledge mapping data”: OBDA example in genomics
“Knowledge mapping data”: OBDA example in genomics

Retrieve all genes of the organisms *Neisseria* for which horizontal gene transfer is predicted or have a GC3 value > 80
“Knowledge mapping data”: OBDA example in genomics

<table>
<thead>
<tr>
<th>Construct</th>
<th>Graphical Element</th>
<th>Semantic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class node</td>
<td>C, D</td>
<td>$C(x), D(x)$</td>
</tr>
<tr>
<td>Object Property link</td>
<td>C → P EFFECT D</td>
<td>$C(x), P(x, y), D(y)$</td>
</tr>
<tr>
<td>Data Property node and link</td>
<td>C → A EFFECT</td>
<td>$C(x), A(x, y)$</td>
</tr>
</tbody>
</table>
“Knowledge mapping data”: OBDA

- OBDA with Ontop [Calvanese et al. (2017)] now more elaborate and more robust
- More case studies: Statoil and EPnet [Calvanese et al. (2016)]
“Knowledge mapping data”: OBDA

- OBDA with Ontop [Calvanese et al.(2017)] now more elaborate and more robust
- More case studies: Statoil and EPnet [Calvanese et al.(2016)]
- Downsides
  - The mapping layer: cumbersome construction and maintenance
  - Low expressiveness for ontology language
  - Limitations on types of queries
Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Queries for decision-making formulate queries using the knowledge graph to retrieve data.

Implementation the actual information system that stores and manipulates the data.
Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Transformation via abstract relational model with additional virtual identifiers.

Implementation the actual information system that stores and manipulates the data.
"Data-transformation-knowledge" example: KnowID

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Transformation via abstract relational model with additional virtual identifiers.

Implementation the actual information system that stores and manipulates the data.
Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Transformation via abstract relational model with additional virtual identifiers

Implementation the actual information system that stores and manipulates the data
Fillottrani, P.R., Keet, C.M. KnowID: An architecture for efficient Knowledge-driven Information and Data access. Data Intelligence, 2020, 2(4): 487-512.
Fillottrani, P.R., Jamieson, S., Keet, C.M. Connecting knowledge to data through transformations in KnowID: system description. Künstliche Intelligenz, 2020, 2020, 34, 373-379.
Knowledge-driven Information and Data access (KnowID)

Fillottrani, P.R., Keet, C.M. KnowID: An architecture for efficient Knowledge-driven Information and Data access. Data Intelligence, 2020, 2(4): 487-512.

Fillottrani, P.R., Jamieson, S., Keet, C.M. Connecting knowledge to data through transformations in KnowID: system description. Künstliche Intelligenz, 2020, 2020, 34, 373-379.
Knowledge-driven Information and Data access (KnowID)

- There’s more on the ‘knowledge and information management’ module:
  - Swap between EER, UML, ORM
    [Keet and Fillottrani(2015), Fillottrani and Keet(2014)]
  - DL (OWL) with reasoner at the back-end
  - Tool: crowd 2.0 (beta)
    http://crowd.fi.uncoma.edu.ar:3335/
    [Braun et al.(2020)]
  - More in the pipeline, such as integrating NOMSA for summarisation and modularisation of ontologies

- Querying with SQLP: SQLP requires less time for understanding and authoring queries, with no loss in accuracy
  [Ma et al.(2018)]

- Data Completion TBD
Outline

1. Context and motivation
2. Resolving conflicts
3. Implementation trade-offs
4. Conclusions
Recap and future work

- Foundational steps towards a framework that can deal in a systematic way with modelling conflicts through conflict resolution
- Notion of conflict set, with a data structure
- A first step towards a library of conflicts
- Some supporting tools for conflict resolution; more needed
- System design trade-offs in connecting the ontologies to the data; more needed
Main collaborators (on the works included in this talk)

- Collaborators: Diego Calvanese and Werner Nutt (FUB, Italy), Pablo Fillottrani (UNS, Argentina), Santi Garcia-Vellido (URV, Spain), Rolf Grütter (WSL, Switzerland), Stephan Jamieson (UCT) Agnieszka Ławrynowicz (PUT, Poland), David Toman (UW, Canada)

- Current and former students: Zubeida Khan, Mandisa Baleni, Kieren Davies, Bradley Malgas, Brian McGeorge, Aashiq Parker, and Muhhammad Patel, Giorgio Stefanoni
References

Distributed ontology, model, and specification language, February 2018.

Germán Braun, Christian Gimenez, Laura Cecchi, and Pablo Fillottrani.
crowd: A Visual Tool for Involving Stakeholders into Ontology Engineering Tasks.

D. Calvanese, P. Liuzzo, A. Mosca, J. Remesal, M. Rezk, and G. Rull.
Ontology-based data integration in epnet: Production and distribution of food during the roman empire.

Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro, and Guohui Xiao.
Ontop: Answering SPARQL queries over relational databases.

Aaron J. Cotnoir.
Anti-symmetry and non-extensional mereology.

P. R. Fillottrani and C. M. Keet.
Dimensions affecting representation styles in ontologies.
24-28 June 2019, Villa Clara, Cuba.
References II

Pablo R. Fillottrani and C. Maria Keet.
Patterns for heterogeneous tbox mappings to bridge different modelling decisions.
30 May - 1 June 2017, Portoroz, Slovenia.

Pablo Rubén Fillottrani and C. Maria Keet.
Conceptual model interoperability: a metamodel-driven approach.
August 18-20, 2014, Prague, Czech Republic.

S. Garcia-Vallvé, E. Guzman, M.A. Montero, and A. Romeu.
HGT-DB: a database of putative horizontally transferred genes in prokaryotic complete genomes.

Yongqun He, Hong Yu, Edison Ong, Yang Wang, Yingtong Liu, Anthony Huffman, Hsin hui Huang, John Beverley, Asiyah Yu Lin, William D. Duncan, Sivaram Arabandi, Jiangan Xie, Junguk Hur, Xiaolin Yang, Luohan Chen, Gilbert S. Omenn, Brian Athey, and Barry Smith.
Cido: The community-based coronavirus infectious disease ontology.
C. Maria Keet and Pablo Rubén Fillottrani.
An ontology-driven unifying metamodel of UML Class Diagrams, EER, and ORM2.
doi: 0.1016/j.datak.2015.07.004.

Weicong Ma, C. Maria Keet, Wayne Oldford, David Toman, and Grant Weddell.
The utility of the abstract relational model and attribute paths in sql.
12-16 Nov. 2018, Nancy, France.

Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, and Carsten Lutz.
**OWL 2 Web Ontology Language Profiles.**
http://www.w3.org/TR/owl2-profiles/.
Thank you!

Questions?

Some self-promotion:

- My textbook on ontology engineering (aimed at computer scientists)
- Also available in paperback: