Patterns for Heterogeneous TBox Mappings to Bridge Different Modelling Decisions

Pablo Rubén Fillottrani[1,2] and C. Maria Keet[3]

[1] Departamento de Ciencias e Ingeniería de la Computación, Universidad Nacional del Sur, Bahía Blanca, Argentina, prf@cs.uns.edu.ar
[2] Comisión de Investigaciones Científicas, Provincia de Buenos Aires, Argentina
[3] Department of Computer Science, University of Cape Town, South Africa
mkeet@cs.uct.ac.za

Abstract. Correspondence patterns have been proposed as templates of commonly used alignments between heterogeneous elements in ontologies, although design tools are currently not equipped with handling these definition alignments nor pattern alignments. We aim to address this by, first, formalising the notion of design pattern; secondly, defining typical modelling choice patterns and their alignments; and finally, proposing algorithms for integrating automatic pattern detection into existing ontology design tools. This gave rise to six formalised pattern alignments and two efficient local search and pattern matching algorithms to propose possible pattern alignments to the modeller.

1 Introduction

Ontology developers face choices on how to represent the subject domain knowledge during the ontology authoring stage. In some instances, there really is a right and a wrong way of representing it—e.g., not confusing subsumption with parthood—but this is not always the case. For instance, whether Marriage should be represented as a class or as an object property. Some OWL files may also lean more toward being a logic-based conceptual data model for a single application, rather than have its knowledge represented such that many applications may avail of it, such as with ontologies for ontology-based data access applications [2] versus foundational ontologies such as DOLCE [16], respectively. In the context of ontology alignment, such different modelling decisions result in modeller-intended alignments that cannot be found automatically with current alignment tools [19, 23] and may have a mismatch logically despite a deemed 'sufficient semantic equivalence' conceptually. For instance, ontology O_1 may have $\mathsf{Employee} \sqsubseteq \mathsf{Person}$, which is a typical subsumption in conceptual modelling, versus $\mathsf{Person} \sqsubseteq \mathsf{PhysicalObject}$, $\mathsf{Employee} \sqsubseteq \mathsf{Role}$, and $\mathsf{Employee} \sqsubseteq \exists \mathsf{inheresIn}.\mathsf{Person}$ in ontology O_2 that is inspired by foundational ontology modelling practices, and where $\mathsf{PhysicalObject} \sqsubseteq \neg\mathsf{Role}$ is also asserted. Aligning Employee and Person across these two ontologies with a simple 1:1 equivalence axiom will then result in an unsatisfiable Employee in O_1. This and similar situations raise the question of how to link or integrate O_1 with O_2 when there is no single vocabulary element

to match with in *both* ontologies, or when intuitively the knowledge represented is deemed sufficiently the same with respect to the subject domain knowledge, but an alignment will result in an inconsistency nevertheless.

This question can be recast as one of somehow having to manage and resolve heterogeneous modelling *patterns*, rather than the customary pairwise alignments of single elements [3, 19]. This requires the capability to align patterns *as a whole*, and thus also involve different types of elements that will have to be bridged, such as a class and an object property, which has been noted widely as an open issue in ontology matching [19]. Ontology Design Patterns (ODPs) have a type of pattern called correspondence pattern, with a few correspondences proposed in [20–22] and several submissions are included in the ODP catalogue[1]. They are mostly definition alignments, however, i.e., with a class on the left-hand side of the equivalence and a pattern (the definition of the class) on the right-hand side (automated only recently [7]), rather than patterns on both sides. Ghidini et al [8] focus on logic and reasoning complexity and cover only a small subset of possible correspondences with single heterogeneous elements. In short, heterogeneous alignments between modelling patterns is a manual, individual, process at present.

We aim to address this mapping impasse by identifying common modelling patterns, their rationale, and providing a formalisation of the corresponding ODPs, inclusive of a formalisation of an ontology pattern itself. This resulted in six complex alignments between modelling patterns. Secondly, we show that *automated finding* of the patterns and *checking correctness* of a possible pattern-based alignment is indeed possible. Algorithmically, finding such patterns can be reduced to localised pattern matching in the ontology. We present two such algorithms for one of the pattern alignments to demonstrate that feasibility; the algorithms for the other patterns follow the same design. They are being implemented in the ICOM tool [4] that already caters for inter-ontology assertions between single elements in different OWL ontologies and reason over them.

The remainder of this paper is structured as follows. Section 2 describes related work. Sections 3 and 4 present the main theoretical contribution of the paper with the formal representation of a design pattern and the pattern alignments. Section 5 describes the optimised algorithms, which is followed by a discussion (Section 6) and conclusions (Section 7).

2 Related works

The work most closely related to ours is that of [20–22], who present mostly definition alignments as correspondence patterns within the context of ontology design patterns. That is, there is one named concept in ontology O_1 that matches with an axiom in O_2, such as $\mathsf{PinotageWine}$ in a simple wine ontology O_1 being aligned to $\mathsf{Wine} \sqcap \exists \mathsf{grapeVariety}.\mathsf{Pinotage}$ from a more detailed *wyn ontologie* O_2. Based on the alignment naming, Scharffe et al.'s (no longer available)

[1] `http://www.ontologydesignpatterns.org`

library [22] also includes a few homogeneous mappings that involve more than one element, such as a range restriction. Somewhat related to this are entity transformations within the same ontology [25], which covers single-entity swaps and one pattern similar to one in [21]. Such correspondences were, at the time, a substantial extension to the 1:1 mappings common in ontology matching [3, 19] and independent of later automated alignment efforts [19]. Definition alignments and automatic finding thereof now has a logic-based tool to assist with this [7].

Other recent efforts in automated ontology matching has commenced considering aligning domain ontologies to a foundational ontology [23], which brings afore a separate set of issues. One is that such matching involves mostly subsumption rather than equivalence [23]. The domain ontology↔foundational ontology alignment issues regarding modelling choices, such as a process as a class (e.g., Running) or an object property (Op) (e.g., runs) and those mentioned in the introduction of this paper, were not addressed. Logic-based approaches for heterogenous alignments were investigated in [8] that focussed on the class↔Op mappings (DL concept, role), such as the marriage example mentioned in the introduction, and Op↔Dp (DL role, attribute) mappings, using bridge rules. These two patterns are a subset of the range needed for heterogeneous alignments and the logic is a subset of OWL ($\mathcal{ALCQI}_b$). The Distributed Ontology Language [17] takes another approach by defining a framework by which to link and extend logical theories. This is useful for the case where the ontologies are represented in different logics, but it does not consider patterns yet.

Conceptual model alignment patterns have been proposed by [5, 14], which resemble some of the cases in ontologies, such as their class↔relationship and the relationship↔attribute, and ORM Value Type↔UML Attribute. They used a rule-based approach to inter-model links in general and more applied with ATL in particular, which is popular in model-driven engineering but not immediately usable for ontologies, and they avail of a particular unifying metamodel to mediate between the conceptual models, which is different for OWL.

In sum, while some reusable ideas have been proposed, logic-based linking of heterogeneous patterns between ontologies it yet to be solved.

3 Formal representation of patterns and alignments

We need a way to formally describe a pattern before being able to declare links between patterns. Such precision is useful for determining a logical or a 'subject domain semantics' equivalence or subsumption or another motivation for aligning the patterns. To the best of our knowledge, there is no formal definition of an ODP yet, therefore we introduce this first before the actual alignment patterns. Note that the focus is on the ontology and logic aspects; for a specification of ODP documentation and metadata, the reader is referred to [12].

Definition 1 (Language of pattern instantiation). *OWL Ontology O with language specification adhering to the W3C standard [18], which has classes $C \in V_C$, object properties $OP \in V_{OP}$, data properties $D \in V_D$, data types $DT \in V_{DT}$*

of the permitted XML schema types, axiom components ('language features') $X \in V_X$*, and such that* $Ax \in V_{Ax}$ *are the axioms.*

Of course, this could be another ontology language, but it is easier for the current presentation to cast it in the context of OWL. The 'axiom components' in Definition 1 include features such as functionality on an object property "≤ 1", transitivity, 'at least one' "$\exists$", and cardinality that can be used according to the syntax of the language.

The pattern itself is a meta-level specification, alike UML's stereotyping. In addition, because some patterns will refer to categories from a foundational ontology, yet which are also included in the ontology, we acknowledge their status here with respect to the context of a pattern.

Definition 2 (Language for patterns: Vocabulary $\mathcal{V}$). *The meta-level (second order) elements (or* stereotypes*) for patterns are:*

- *class* $C \in V_C$ *as* $\mathcal{C}$ *in the pattern;*
- *object property* $OP \in V_{OP}$ *as* $\mathcal{R}$ *in the pattern;*
- *data property* $D \in V_D$ *as* $\mathcal{D}$ *in the pattern;*
- *data type* $DT \in V_{DT}$ *as* $\mathcal{DT}$ *in the pattern;*
- *reserved set of entities from a foundational ontology, as* $\mathcal{F}$ *in the pattern;*

where added subscripts i *with* $1 \leq i \leq n$ *may be different elements. Two elements in the vocabulary are called* homogeneous *iff they belong to the same type, i.e., they are both classes, or both object properties, and so on. Elements can be used in axioms* $Ax \in V_{Ax}$ *that consists of axiom components* $x \in V_X$ *in the pattern such that the type of axioms are those supported in the ontology language in which the instance of the pattern is represented.*

Subsumption and equivalence axiom components relate *homogeneous* elements; functionality and at least one axiom components relate *heterogeneous* elements from the vocabulary. The set of "reserved entities" in $\mathcal{F}$ depends on which foundational ontology is used; e.g., for DOLCE, this set includes, among others, *Perdurant*, *Endurant*, and *qt* and for BFO, *Independent Continuant* and *Quality*.

An ontology pattern P can now be defined as follows.

Definition 3 (Ontology Pattern P). *An* ontology pattern P *consists of more than one element from vocabulary* $\mathcal{V}$ *which relate through at least one axiom component from* V_X*. Its specification contains the:*

- *pattern name;*
- *pattern elements from* $\mathcal{V}$*;*
- *pattern axiom component(s) from* V_X*;*
- *pattern's full formalisation.*

For instance, a simple *named class subsumption* pattern with axiom component $\sqsubseteq$ (in V_X) and two named classes is formalised as $\mathcal{C}_1 \sqsubseteq \mathcal{C}_2$, which can be instantiated with classes, e.g., $\mathsf{Human}, \mathsf{Animal} \in V_C$ as, in DL notation, $\mathsf{Human} \sqsubseteq \mathsf{Animal}$. A slightly more comprehensive one, the *basic all-some* pattern, then has as specification: *pattern name*: basic all-some, with *pattern elements*: $\mathcal{C}_1, \mathcal{C}_2, \mathcal{R}$, the *pattern axiom component(s)*: $\sqsubseteq$, $\exists$, and the *pattern's full formalisation*: $\mathcal{C}_1 \sqsubseteq \exists \mathcal{R}.\mathcal{C}_2$. An instantiation of this pattern is, e.g., $\mathsf{Professor} \sqsubseteq \exists \mathsf{teaches}.\mathsf{Course}$.

Now that we have a precise notion of patterns and instantiations, we can proceed to define mappings between elements in different ontologies in two steps: the component of common homogeneous *mappings* and the homogeneous with heterogeneous *alignments* that are typically needed to align patterns as a whole. The former is defined in Definition 4, which is a shorthand version of the lengthy specification in [3], and the latter in Definition 5.

Definition 4 (Homogeneous mapping). *Let O, O' be two ontologies with vocabularies $\mathcal{V}, \mathcal{V}'$. A* mapping *is a subsumption or equivalence axiom relating two homogeneous elements, one in $\mathcal{V}$ and the other in $\mathcal{V}'$.*

For example, to relate the class Teacher in O to Instructor in O' with the mapping Instructor $\sqsubseteq$ Teacher. Mappings provide bindings between homogeneous elements in different ontologies, constituting the basis for ontology pattern alignment.

Definition 5 (Ontology Pattern Alignment, OPA). *An* ontology pattern alignment *OPA consists of two ontology patterns, P and P', such that its signature Σ is a subset of the signature of the respective ontologies O and O', i.e., $\Sigma(P) \subseteq \Sigma(O)$ and $\Sigma(P') \subseteq \Sigma(O')$, and alignment axioms*

- *alignment pattern name;*
- *pattern elements;*
- *alignment patterns' context, consisting of:*
  - *O's pattern P*
  - *O''s pattern P'*
- *alignment pattern axiom component(s) from V_X;*
- *pattern alignment's formalisation, composed of:*
  - *a (possibly empty) set of mappings between homogeneous elements in P and P'*
  - *a set of axioms made from components in V_X connecting heterogeneous elements in P and P'*

An OPA thus relates different patterns in separate ontologies, based on previous knowledge of class or *Op* alignments.

Note that here, and in the remainder of the paper, a pattern must neither be inconsistent nor have unsatisfiable classes or *Op*s. That is, a pattern is well-formed, and verified to be so before any alignment of patterns will occur.

4 Aligning alternate modelling patterns

In presenting the patterns, we first introduce those motivated by 'conceptual' or 'subject domain semantics-motivated' mappings that have sufficient semantic approximation and are typical for differences in modelling decisions between domain ontologies and foundational ontologies (Section 4.1). Subsequently, we analyse, update, and formalise those presented elsewhere, which are mostly of the definitorial type rather than true pattern mappings (Section 4.2). In interest of space, we omit from the presentation the "pattern axiom component(s)", for they they are principally relevant for the language fragments aspects, which is not the current scope and they can be seen from the full formalisation anyway.

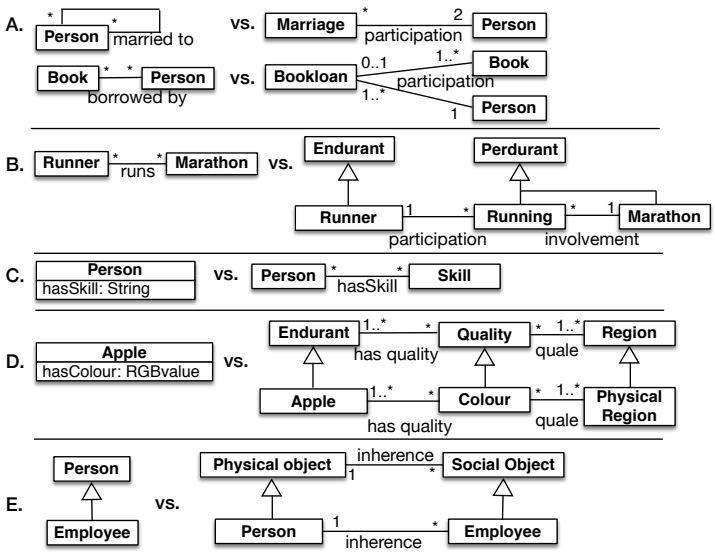

Fig. 1. Examples of the modelling patterns. A: generic class vs *Op*; B: Perdurant class vs *Op*; C: class vs. *Dp*; D: qualities vs *Dp*s; E: example on representing roles.

4.1 Matching modelling patterns

Five pattern alignments are introduced, of which common examples are shown in Fig. 1. While they may look different, some are slight variants of the same pattern, so, in the interest of space, we formalise the principal cases only.

A: Class vs. object property (*Op*) is the more generic, unconstrained version of Fig 1-B ('Perdurant class vs. *Op*'; see below), which is called *granularity mismatch* in [6]. The typical example is Marriage. It may well be that most practical cases turn out to be of that pattern because typically verbs are used as names of *Op*s, which, in turn, are typically reified as dolce:Perdurant or bfo:Occurrent.

- *alignment pattern name*: *class-OP*
- *pattern elements*: $\mathcal{C}_1, \mathcal{C}_2, \mathcal{R}_1$ from O, $\mathcal{C}'_3, \mathcal{C}'_4, \mathcal{C}'_5, \mathcal{R}'_2, \mathcal{R}'_3$ from O'
- *alignment pattern' contexts*:
  - *pattern P in O*: $\exists \mathcal{R}_1.\mathcal{C}_2 \sqsubseteq \mathcal{C}_1$ and $\exists \mathcal{R}_1^-.\mathcal{C}_1 \sqsubseteq \mathcal{C}_2$;
  - *pattern P' in O'*: $\exists \mathcal{R}'_2.\mathcal{C}'_4 \sqsubseteq \mathcal{C}'_3$, $\exists \mathcal{R}'^-_2.\mathcal{C}'_3 \sqsubseteq \mathcal{C}'_4$, $\exists \mathcal{R}'_3.\mathcal{C}'_5 \sqsubseteq \mathcal{C}'_3$, $\exists \mathcal{R}'^-_3.\mathcal{C}'_3 \sqsubseteq \mathcal{C}'_5$, $\mathcal{C}'_3 \sqsubseteq (\exists \mathcal{R}'_2)$, and $\mathcal{C}'_3 \sqsubseteq (\exists \mathcal{R}'_3)$.
- *pattern's full formalisation*:
  - *homogeneous mappings*: between $\mathcal{C}_1$ and $\mathcal{C}'_4$ and between $\mathcal{C}_2$ and $\mathcal{C}'_5$, which may be subsumption or equivalence relations.
  - *heterogeneous alignments*: $\exists \mathcal{R}_1 \sqsubseteq \mathcal{C}'_3$, $\exists \mathcal{R}_1^- \sqsubseteq \mathcal{C}'_3$, $\mathcal{C}'_3 \sqsubseteq \exists \mathcal{R}_1 \sqcap \exists \mathcal{R}_1^- \sqcap (\leq 1\, \mathcal{R}_1) \sqcap (\leq 1\, \mathcal{R}_1^-)$.

Fig. 1-A includes two instantiations of this pattern. In the first one, $\mathcal{C}_1 \equiv \mathcal{C}_2$ for Person, $\mathcal{C}'_4 \equiv \mathcal{C}'_5$ for Person as well, and the pattern associates the role married to

($\mathcal{R}_1$) with the class Marriage (instantiating $\mathcal{C}'_3$). There must be an equivalence or a subsumption mapping between the Person classes in both ontologies. Note that due to cardinality constraints in P' that are not in P, the alignment *de facto* generates a new property that is a sub property of $\mathcal{R}_1$.

B: Perdurant class vs. *Op* This mismatch tends to occur more often when aligning a domain ontology to a foundational ontology (FO). FOs typically have a class Perdurant (Occurrent) for objects unfolding in time (cf. endurants that are wholly present at each time instant they exist), which is the branch in the TBox to represent entities such as Running and Performance. They relate to endurants, like a runner participates in a running (event) like a marathon. Domain ontologies may use a modelling approach more typical of conceptual modelling, where there would not be a 'running', but an *Op* runs between a class runner and the class marathon, and an *Op* performs etc. That is, the choice between the verb or its reification. For the FO, there are thus three classes (runner, running, and marathon) with two *Op*s, whereas in the domain ontology, there are two classes (runner and marathon) and one *Op*. The formalisation of this pattern is similar to case A, but $\mathcal{F}$ is non-empty such that $\mathcal{C}'_3 \sqsubseteq PD$ (DOLCE's Perdurant, or an equivalent [13]) and at least $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}'_4$ and $\mathcal{C}'_5$ in the pattern are subsumed by ED (Endurant) or its equivalent.

C: Class vs data property (*Dp*) A typical example that also persists in conceptual modelling, is whether 'skill' should be modelled as a class Skill with subclasses or as a *Dp* hasSkill with data values. From an ontological viewpoint, the former is better, as the latter diminishes the chances of interoperability and usability of ontologies, but it does occur. Note that in the pattern alignment, we take *Dp*s to be specialisations of *Op*s, which are recast as a subsumption between two unnamed classes.

- *alignment pattern name*: *class-DP*
- *pattern elements*: $\mathcal{C}_1, \mathcal{D}_1, \mathcal{DT}_1$ from O, $\mathcal{C}'_2, \mathcal{C}'_3, \mathcal{R}'_1$ from O'.
- *alignment pattern contexts*:
  - *pattern P in O*: $\exists \mathcal{D}_1 \sqsubseteq \mathcal{C}_1$ and $\exists \mathcal{D}_1.\mathcal{DT}_1 \sqsubseteq \mathcal{C}_1$;
  - *pattern P' in O'*: $\exists \mathcal{R}'_1 \sqsubseteq \mathcal{C}'_2$, and $\exists \mathcal{R}'^{-}_1 \sqsubseteq \mathcal{C}'_3$.
- *pattern's full formalisation*:
  - *homogeneous mappings*: $\mathcal{C}_1 \sqsubseteq \mathcal{C}'_2$;
  - *heterogeneous alignments*: $\exists \mathcal{D}_1.\mathcal{DT}_1 \sqsubseteq \exists \mathcal{R}'_1.\mathcal{C}'_3$

Op* vs *Dp This is a consequence of the 'class vs data property' choice, for the *Dp* turns into a class, which therewith forces the introduction of an *Op*. The details of *Op* vs. *Dp* are a fragment of the previous one, and is therefore omitted.

D: Qualities vs. *Dp*s This distinction is another case of FO modelling vs the more widely-known option from conceptual modelling practices. Instead of a *Dp* such as hasColour and hasHeight that are known in philosophy as *attributions*, one can turn that into a class of type Quality (or similar) and then have two properties, one from the class of type Endurant to the Quality and one from the Quality to a Region for the value space. This is a modification of case C, for it

entails 'class vs. data property' as the attribute is turned into a class, yet it is not just any class but specifically Quality from a FO (i.e., it is in $\mathcal{F}$), with associated *Op*s and a subclass of Abstract $\in \mathcal{F}$ for the category of value regions, and associated constraints.

- *alignment pattern name*: *quality-DP*
- *pattern elements*: $\mathcal{C}_1, \mathcal{D}_1, \mathcal{DT}_1$ from O, $\mathcal{C}'_2, \mathcal{C}'_3, \mathcal{C}'_4, \mathcal{C}'_5, \mathcal{C}'_6, \mathcal{C}'_7, \mathcal{R}'_1, \mathcal{R}'_2$ from O' where $\mathcal{C}'_3, \mathcal{C}'_5, \mathcal{C}'_7, \mathcal{R}'_1, \mathcal{R}'_2 \in \mathcal{F}$.
- *alignment pattern contexts*:
  - *pattern P in O*: $\exists \mathcal{D}_1.\mathcal{DT}_1 \sqsubseteq \mathcal{C}_1$;
  - *pattern P' in O'*: $\mathcal{C}'_2 \sqsubseteq \mathcal{C}'_3$, $\mathcal{C}'_4 \sqsubseteq \mathcal{C}'_5$, $\mathcal{C}'_6 \sqsubseteq \mathcal{C}'_7$ $\exists \mathcal{R}'_1.\mathcal{C}'_5 \sqsubseteq \mathcal{C}'_3$, $\exists \mathcal{R}'^{-}_1.\mathcal{C}'_3 \sqsubseteq \mathcal{C}'_5$, $\exists \mathcal{R}'_2.\mathcal{C}'_7 \sqsubseteq \mathcal{C}'_5$, $\exists \mathcal{R}'^{-}_1.\mathcal{C}'_5 \sqsubseteq \mathcal{C}'_7$, $\mathcal{C}'_5 \sqsubseteq \exists \mathcal{R}'^{-}_1.\mathcal{C}'_3$, $\mathcal{C}'_5 \sqsubseteq \exists \mathcal{R}'_2.\mathcal{C}'_7$, and these are inherited down the hierarchy for $\mathcal{C}'_2$, $\mathcal{C}'_4$, and $\mathcal{C}'_6$.
- *pattern's full formalisation*: (in addition to the "alignment pattern contexts")
  - *homogeneous mappings*: $\mathcal{C}_1 \sqsubseteq \mathcal{C}'_2$;
  - *heterogeneous alignments*: $\exists \mathcal{D}_1.\mathcal{DT}_1 \sqsubseteq \exists \mathcal{R}'_1.(\exists \mathcal{R}'_2.\mathcal{C}'_6)$

If $\mathcal{F}$ draws from DOLCE's vocabulary, then $\mathcal{C}'_3 \equiv ED$ (Endurant), $\mathcal{C}'_5 \equiv Q$ (Quality), $\mathcal{C}'_7 \equiv R$ (Region), $\mathcal{R}'_1 \equiv qt$ ('has quality'), and $\mathcal{R}'_2 \equiv ql$ ('has quale'); see [13] for their mappings to BFO and GFO.

E: Representing roles The differences for this case can be traced back to both conceptual modelling practices and OntoClean [9] that conflict with FO guidelines. For instance, one can represent Employee $\sqsubseteq$ Person in an ontology, or, more generally in OntoClean terminology, that an anti-rigid property (like Employee) is subsumed by a rigid one (like Person). FOs put those anti-rigid properties—they being the roles played by rigid properties (Independent Continuant in BFO or Physical Object in DOLCE)—in another branch in the taxonomy, typically as subclasses of, e.g., Role in BFO or Social Object in DOLCE. The two are then related through an *Op* generally known as *inherence*, where the role inheres in the physical object or the physical object is the bearer of the role.

- *alignment pattern name*: *subs-Role-inherence*
- *pattern elements*: $\mathcal{C}_1, \mathcal{C}_2$ from O, $\mathcal{C}'_3, \mathcal{C}'_4, \mathcal{C}'_5, \mathcal{C}'_6, \mathcal{R}'_1$ from O' and $\mathcal{C}'_4, \mathcal{C}'_6, R'_1 \in \mathcal{F}$.
- *alignment pattern contexts*:
  - *pattern P in O*: $\mathcal{C}_1 \sqsubseteq \mathcal{C}_2$;
  - *pattern P' in O'*: $\mathcal{C}'_3 \sqsubseteq \mathcal{C}'_4$, $\mathcal{C}'_5 \sqsubseteq \mathcal{C}'_6$, $\exists \mathcal{R}'_1.\mathcal{C}'_6 \sqsubseteq \mathcal{C}'_4$, $\exists \mathcal{R}'_1.\mathcal{C}'_5 \sqsubseteq \mathcal{C}'_3$, $\exists \mathcal{R}'^{-}_1.\mathcal{C}'_4 \sqsubseteq \mathcal{C}'_6$, $\exists \mathcal{R}'^{-}_1.\mathcal{C}'_3 \sqsubseteq \mathcal{C}'_5$, $\mathcal{C}'_6 \sqsubseteq = 1\, \mathcal{R}'_1.\mathcal{C}'_4$, $\mathcal{C}'_5 \sqsubseteq = 1\, \mathcal{R}'_1.\mathcal{C}'_3$.
- *pattern's full formalisation*:
  - *homogeneous mappings*: $\mathcal{C}_1 \equiv \mathcal{C}'_5$, $\mathcal{C}_2 \equiv \mathcal{C}'_3$;
  - *heterogeneous alignments*: the subsumption relation in O aligns with $\mathcal{R}_1$, which is not expressible in DL or OWL.

If $\mathcal{F}$ draws from DOLCE, then $\mathcal{C}'_4 \equiv POB$ (Physical object), $\mathcal{C}'_6 \equiv SOB$ (Social object), and $\mathcal{R}'_1 \equiv OD$ (one-sided constant dependence) or $\mathcal{R}'_1 \equiv OGD$ (one-sided generic constant dependence); see [13] for their mappings to BFO and GFO.

4.2 Assessment and formalisation of other correspondence patterns

As noted, correspondence patterns have been proposed before in [20–22] and there are several submissions in the ODP catalogue. The analysis of these CPs

brings afore the difference between patterns that are *patterns* and those that are essentially *definition mappings*. That is, the former has more than one vocabulary element on the left-hand side of the mapping (recall Definition 3 of an ODP) *and* more than one vocabulary element on the right-hand side of the inclusion or equivalence, whereas the latter has *one* vocabulary element on the left-hand side and more than one on the right-hand side (or vv.). We first provide the analysis of the proposed CPs and refine and formalise the the patterns afterward.

Ritze et al. [20] surveyed multiple sources, and proposed four patterns based on that, three of which are of the definition-mapping type:

CAT: Class Attribute Type: $C_1 \equiv \exists R_2.C_2$, where R can be an Op or Dp, with as example: $\mathsf{PositiveReviewedPaper} \equiv \mathsf{Paper} \sqcap \exists \mathsf{hasEvaluation.positive}$ [20]. By that example, R is a Dp, not an Op, and the right-hand side in the pattern misses a parent class. This would need to be separated into a CAT for Ops and and CAT for Dps, so as to recognise the pattern properly in alignment tools. The omission of the named class on the rhs seems to have been unintended, for [20] note it is the same as the CAT in the ODP catalogue (see below), which does include the class on the rhs, as does its 'inverse', below.

CAT$^-$: $C_1 \equiv C_2 \sqcap \exists R_2^-.C_3$; e.g., $\mathsf{Researcher} \equiv \mathsf{Person} \sqcap \exists \mathsf{researchedBy}^-.\top$ [20].

CAV: Class Attribute Value, with nominals: $C_1 \equiv \exists R_2.\{...\}$, with as example $\mathsf{submittedPaper} \equiv \mathsf{submission}.\{\mathsf{true}\}$, where 'true' is discussed as if it were a Boolean but is represented as one of the values, like the one for passing the course as $\mathsf{hasExamScore}.\{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}\}$ [20]. However, with nominals/one-of, which is a class, CAV turns out to be a variant of CAT.

PC: $R_1 \equiv S_2 \circ S_3$, where S_3 is a Dp [20]. However, OWL2 does not permit a chain combining Ops and a Dp, and if S_3 were to be an Op, then the ontologies+mapping assertion goes beyond OWL 2 DL, for true role composition is undecidable. This can be corrected by asserting subsumption rather than equivalence, which brings us back to our pattern D.

Scharffe and Fensel [21, 22] claim to have a library of 35 patters, but the URL in the paper is broken, therefore, we assess only those described in [21, 22], three of which are definitorial. Those patterns all have observed instances, such as when trying to align the wine ontology with the ontologie du vin, DOAP with OSSV, and FOAF with itself, whose examples are omitted here for brevity.

CAT: Class Attribute Type: $C_1 \equiv C_2 \sqcap \exists R_2.C_3$ [21], as intended by [20].

CRD: $C_1 \sqsubseteq \exists R_2.(\exists S_2.datatype) \equiv D_3 \sqsubseteq \exists R_3.datatype$, so S_2 and R_3 are Dps [21], and noting that the 'Property-Relation Correspondence' in [22] is a fragment of CRD. This is a sort of specialisation of our pattern D, as a Dp is a kind of Op.

UI: Union and intersection patterns: they were not formalised in [21, 22], but are obvious, being: $C_1 \equiv C_2 \sqcup C_3$ and $C_1 \sqcap C_2 \equiv C_3$.

The other correspondence patterns listed in [22] include homogeneous mappings of one or more entities, such as 'Equivalent Attribute' and domain/range axioms, whereas a 'Class to instance' mapping constitutes a modelling error in either O or O' and is therefore not considered here.

The ODP catalogue (see fn. 1) has two types of correspondence patterns: reengineering and alignment patterns. The former can be seen as a 'swapping' of elements within the same ontology for syntax refactoring, whereas the latter are matchings between two ontologies. There are 13 submitted alignment patterns. They include the aforementioned Class Union (UI), CAT, CAV, Class by path attribute value (alike the PC of [20]), Class correspondence by relation domain with an unsupported `AttributeOccurenceCondition`, (named) Class Equivalence, (named) Class Subsumption, disjointness between (named) classes, and three Vocabulary Alignment Patterns (VAP) that have the same formalisation in the catalogue, amounting to an *Op* subsumption where the subsumed one has a domain and range axiom whose classes are aligned as well.
This results in the following two heterogeneous alignments inspired by [21]'s CRD and the catalogue's VAP:

Class-role-attribute pattern

- *alignment pattern name*: *class-OP-DP*
- *pattern elements*: $\mathcal{C}_1$, $\mathcal{R}_1$, $\mathcal{D}_1$, $\mathcal{DT}_1$ from O, and $\mathcal{C}'_2$, $\mathcal{D}'_2$, $\mathcal{DT}'_2$ from O'.
- *alignment pattern' contexts*:
  - *pattern P in O*: $\mathcal{C}_1 \sqsubseteq \exists\mathcal{R}_1.(\exists\mathcal{D}_1.\mathcal{DT}_1)$;
  - *pattern P' in O'*: $\mathcal{C}'_2 \sqsubseteq \exists\mathcal{D}'_2.\mathcal{DT}'_2$.
- *pattern's full formalisation*:
  - *homogeneous mappings*: $\mathcal{C}_1 \equiv \mathcal{C}'_2$, $\mathcal{DT}_1 \equiv \mathcal{DT}'_2$;
  - *heterogeneous alignments*: $\exists\mathcal{D}'_2 \sqsubseteq \exists\mathcal{R}_1.(\exists\mathcal{D}_1.\mathcal{DT}_1)$.

Vocabulary alignment pattern

- *alignment pattern name*: *OP-subs*
- *pattern elements*: $\mathcal{C}_1$, $\mathcal{R}_1$, $\mathcal{C}_2$ from O, and $\mathcal{C}'_3$, $\mathcal{R}'_2$, and $\mathcal{C}'_4$ from O'.
- *alignment pattern' contexts*:
  - *pattern P in O*: $\mathcal{C}_1$, $\mathcal{R}_1$, $\mathcal{C}_2$ (i.e., three independent entities);
  - *pattern P' in O'*: $\exists\mathcal{R}'_2.\top \sqsubseteq \mathcal{C}'_3$, $\exists\mathcal{R}'^{-}_2.\top \sqsubseteq \mathcal{C}'_4$.
- *pattern's full formalisation*:
  - *homogeneous mappings*: $\mathcal{C}_1 \sqsubseteq \mathcal{C}'_3$, $\mathcal{C}_2 \sqsubseteq \mathcal{C}'_4$, $\mathcal{R}_1 \sqsubseteq \mathcal{R}'_2$;
  - *heterogeneous alignments*: $\exists\mathcal{R}_1.\top \sqsubseteq \mathcal{C}_1$, $\exists\mathcal{R}^{-}_1.\top \sqsubseteq \mathcal{C}_2$.

The formalisation of the six OPAs, and thus 12 ODPs, presented in this section constitute typical cases of modelling decisions. Although this might still be shown to be not exhaustive, it does comprise a systematic approach for extension with other complex alignments.

5 Alignment pattern search and checking algorithms

In order to show viability of our proposal, we introduce in this section two algorithms for automated finding of, and checking correctness for, for handling pattern A from Section 4.1. These algorithms can be incorporated in an ontology development tool based on a DL/OWL reasoner, evidencing patterns are practically relevant in ontology engineering (lest the pattern becomes an abstract constraint relegated to a handwaiving manual check at best). We assume the tool already has the capability of simultaneously handling two or more ontologies, and also can represent inter-ontology homogeneous mappings between classes and between *Op*s. ICOM [4] is an example of such a tool.

Inter-ontology mappings can be explicitly added by the user, or implicitly detected by the tool assisted by the reasoner. Both algorithms are described

using calls to OWLink [15] services in order to make them independent of the actual DL/OWL reasoner used in the tool.

Algorithm 1 searches for all possible pattern matching instantiations in any pair of ontologies such that the homogeneous mappings have already been analysed. The search is done based on reasoner services and proposes to the user to select which of the instantiations found are meaningful for the ontology integration process. It uses OWLink's IsOPSatisfiable for checking *Op* satisfiability and GetSubClasses with the 'direct' flag activated in order to get the immediate descendants of a class expression, and it needs to be run each time before accepting a suggestion made by the user. Algorithm 1's running time is proportional to m^2r^3c, where m is the number of detected homogeneous mappings, r the number of *Op*s and c the number of classes in the integration scenario. This is acceptable for being on-demand available when the user considers the homogeneous mappings stable.

Algorithm 1 Alignment pattern **A** Search

Precondition: All mappings between ontologies O and O' already found. Mappings are of the form $C\text{rel}C'$ with $C \in \mathcal{V}_C$, $C' \in \mathcal{V}'_C$ and $\text{rel} \in \{\sqsubseteq, \sqsupseteq, \equiv\}$.

1: **function** ALIGNMENT PATTERN **A** CANDIDATES( )
2: $A \leftarrow \emptyset$ ▷ A is the set of candidate alignments
3: **for** each pair of mappings $C_1\text{rel}_1C'_4$ and $C_2\text{rel}_2C'_5$, and relations $R_1 \in \mathcal{V}_{OP}$ and $R'_2, R'_3 \in \mathcal{V}'_{OP}$ **do**
4: **for** each ClassSynset S in GetSubClasses(O',$\exists R'_2 \sqcap \exists R'_3$,direct=true) **do**
5: $C'_3 \leftarrow$ a representative class in S
6: **if** (IsOPSatisfiable(O', $(\exists R'_2.C'_4 \sqsubseteq C'_3) \sqcap (\exists R'^{-}_2.C'_3 \sqsubseteq C'_4)$) and
7: IsOPSatisfiable(O', $(\exists R'_3.C'_5 \sqsubseteq C'_3) \sqcap (\exists R'^{-}_3.C'_3 \sqsubseteq C'_5)$) and
8: IsOPSatisfiable(O, $(\exists R_1.C_2 \sqsubseteq C_1) \sqcap (\exists R^{-}_1.C_1 \sqsubseteq C_2)$)) **then**
9: Add $\langle R_1 \in O, C'_3, R'_2, R'_3 \in O' \rangle$ to A
10: **end if**
11: **end for**
12: **end for**
13: **return** A
14: **end function**

To illustrate the algorithm with an example, suppose we are integrating two pre-existing ontologies, represented as different ICOM schemas within the same grand project as shown in Fig. 2. The **TennisCircuit** ontology (top one) contains knowledge about tennis matches and the **Olympiad** ontology (second one) describes athletes who participated in Olympic competitions (setting aside whether they are 'good'). ICOM's homogeneous mappings are shown with thin cross-ontology lines, being TennisPlayer $\sqsubseteq$ Athlete and Tennis $\sqsubseteq$ Tournament, which may have been explicitly added by the user, detected by deduction from the tool, or (when integrated) suggested by an ontology alignment technique [1]. After assessing the mappings, the user may ask the tool to find possible alignment patterns. It runs Algorithm 1, and in line 5 it finds class Match such that it is connected to partOf and played. Then it asks the reasoner to check if these

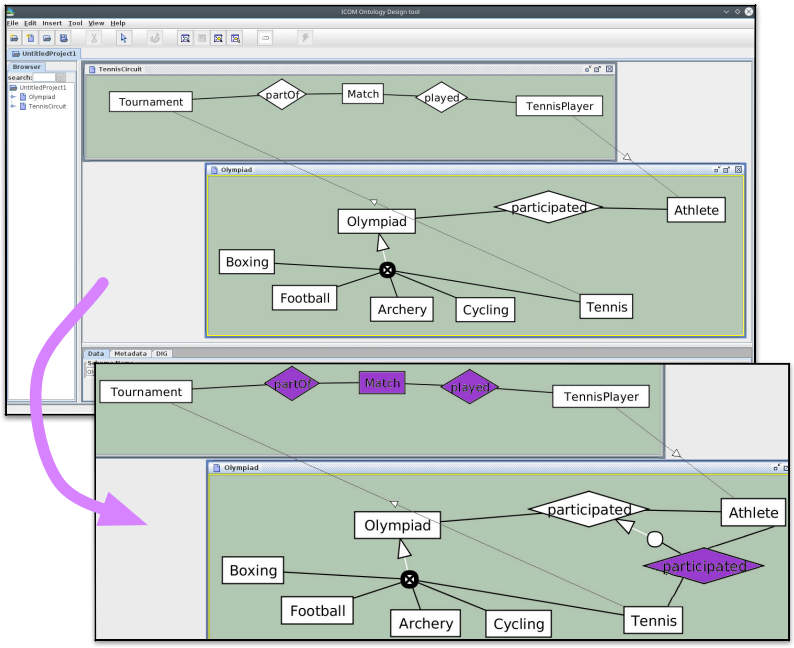

Fig. 2. ICOM integration project with mappings (top) and pattern alignment suggestion (inset).

two *Op*s are related to the already mapped classes Tournament and TennisPlayer from **TennisCircuit** (lines 6-7), and finds participated that connects the mapped classes Tennis and Athlete from **Olympiad** in line 8. Given everything is satisfiable, it sets a new candidate alignment connecting a subproperty of participated in **Olympiad** with the concept Match in **TennisCircuit**, and presents it to the user in a different colour as depicted in Fig. 2 (inset). Observe that the matched *Op* in **Olympiad** is not directly defined over the mapped class, but instead over a class higher in the hierarchy. Therefore, the suggested alignment is based on the restriction of the original *Op*. The role of the reasoner in the algorithm is essential in order to resolve which *Op* is applicable for the alignment.

Algorithm 2 checks whether a proposed pattern A instantiation follows the formalised pattern properties. This would admit the alignment pattern to be included in the integration process. The application scenario for this algorithm is that the designer suspects the pattern is relevant for the integration, and has already set the alignments in the project. Then she asks tool to verify all conditions for the alignment pattern are met. For this algorithm there is no precondition for its execution, and it simply communicates with the reasoner checking satisfiability of the pattern formalisation within both ontologies.

Algorithms for the other patterns can be developed analogously, and are not shown due to space limitations. If the pattern references a FO, as in pattern

Algorithm 2 Alignment pattern **A** Check

```
1: function Alignment Pattern A Check(R_1 ∈ V_OP, C'_3 ∈ V'_C, R'_2.R'_3 ∈ V'_OP )
2:     for each ClassSynSet S_1 in GetSubClasses(O, ∃R_1, direct=true) do
3:         for each ClassSynSet S_2 in GetSubClasses(O, ∃R_1^-, direct=true) do
4:             C_1, C_2 ← a representative class in S_1, S_2
5:             if not IsOPSatisfiable(O, (∃R_1.C_2 ⊑ C_1) ⊓ (∃R_1^-.C_1 ⊑ C_2) then
6:                 return false
7:             end if
8:         end for
9:     end for
10:    for each ClassSynSet S_3 in GetSubClasses(O', ∃R'^-_2, direct=true) do
11:        for each ClassSynSet S_4 in GetSubClasses(O', ∃R'^-_3, direct=true) do
12:            C'_4, C'_5 ← a representative class in S_3, S_4
13:            if not IsOPSatisfiable(O', (∃R'_2.C'_4 ⊑ C'_3)⊓(∃R'_3.C'_5 ⊑ C'_3)⊓(∃R'^-_2.C'_3 ⊑
   C'_4) ⊓ (∃R'^-_3.C.3' ⊑ C'_5) then
14:                return false
15:            end if
16:        end for
17:    end for
18:    return true
19: end function
```

B, D, and E, this fact can be used to shorten the algorithm execution time by restricting the search of candidates to only the descendants of the FO's elements in the pattern. Pattern C can be detected by exhaustively analysing each *Dp* involved in a given mapped class. Pattern D can be handled by similar algorithms as algorithms 1 and 2 by replacing the *Op* in the first ontology (O) with a *Dp*, and then the search shortened with the FO optimisation.

6 Discussion

The patterns and their alignments proposed in the previous sections allow for both specifying and finding more complex alignment than the customary 1:1 mappings and definition mappings. Reconsidering the ODP formalisation of Section 3, this bears similarity to the framework in [21], for it takes into account a certain 'layering' of components, from the actual pattern instantiations in an ontology represented in a particular language and their mapping to another one (called "ground correspondences" in [21]), a "correspondence level" where the correspondences between the patterns are represented in an abstract format cf. the ground correspondence in the application, and the abstraction of that as a correspondence pattern. Having patterns on both sides of the (equivalence or subsumption) mapping bridge essentially extends these layers to the more complex scenarios of pattern alignments considered in this paper. Further, the algorithms have been designed such that one can do a local search compared to brute force exhaustive searches across the whole ontology. This speeds up the alignment process.

The pattern alignments do not come 'for free' computationally, however. Some issues and consequences of aligning patterns were already mentioned in passing: one cannot assert that some relation R in ontology O is the same as $S \circ T$ in ontology O', for $R \equiv S \circ T$ is undecidable, and thus outside OWL 2 DL. Other combinations may also result in undecidability whilst the separate ontologies are within OWL 2 DL, especially due to the regularity constraint on role hierarchies [10]. Our alignments do remain within OWL 2 DL, but it does not guarantee to remain within the O's or O''s fragments; e.g., pattern alignment A adds a conjunction.

While the scope here is complex alignments of ontologies, one also could use the results obtained as a basis to *replace* patterns in one single ontology, i.e., moving toward a *refactoring* of a single ontology as a preparatory step for alignment. For instance, to prepare a domain ontology for linking to a foundational ontology, which thereby can substantially extend the entity transformations of PatOMat [25]. It is expected to be useful also for the process of transforming a foundational ontology inspired domain ontology into an 'application ontology' for ontologically well-founded conceptual models, whose ideas were proposed in [11, 24], but that yet still lack a formalisation and automation.

7 Conclusions

The paper introduced a first formalisation of an ontology design pattern, as a prerequisite for complex alignments between different common modelling patterns. Twelve patterns for six pattern alignments were motivated from a modelling viewpoint and formalised, augmented with two alignments inspired by definition alignments. They are supported by two efficient local search and pattern matching algorithms that can propose possible pattern alignments to the modeller. We are currently implementing the algorithms in ICOM. As future work, the proposed pattern alignments may be recast as *refactoring* patterns within a same ontology that may be of use in test-driven development of ontologies or facilitate the automation of the link to ontology-driven conceptual modelling.

References

1. Ardjani, F., Bouchiha, D., Malki, M.: Ontology-alignment techniques: Survey and analysis. Int. J. of Modern Education and Computer Science 7(11), 67 (2015)
2. Calvanese, D., Liuzzo, P., Mosca, A., Remesal, J., Rezk, M., Rull, G.: Ontology-based data integration in EPNet: Production and distribution of food during the Roman Empire. Eng. Appl. of AI 51, 212–229 (2016)
3. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer (2007)
4. Fillottrani, P.R., Franconi, E., Tessaris, S.: The ICOM 3.0 intelligent conceptual modelling tool and methodology. Semantic Web Journal 3(3), 293–306 (2012)
5. Fillottrani, P.R., Keet, C.M.: Conceptual model interoperability: a metamodel-driven approach. In: Bikakis, A., et al. (eds.) Proc. of RuleML'14. LNCS, vol. 8620, pp. 52–66. Springer (2014), 18-20 Aug 2014, Prague, Czech Republic

6. Gangemi, A.: Ontology design patterns for semantic web content. In: Gil, Y., et al. (eds.) Proc. of ISWC'05. pp. 262–276. Springer (2005), galway, Ireland
7. Geleta, D., Payne, T.R., Tamma, V.: An investigation of definability in ontology alignment. In: Blomqvist, E., et al. (eds.) Proc of EKAW'16. LNAI, vol. 10024, pp. 255–271. Springer (2016), 19-23 Nov 2016, Bologna, Italy
8. Ghidini, C., Serafini, L., Tessaris, S.: Complexity of reasoning with expressive ontology mappings. In: Eschenbach, C., et al. (eds.) Proc. of FOIS'08. Frontiers in Artificial Intelligence and Applications, vol. 183, pp. 151–163. IOS Press (2008)
9. Guarino, N., Welty, C.: An overview of OntoClean. In: Staab, S., Studer, R. (eds.) Handbook on ontologies, pp. 151–159. Springer Verlag (2004)
10. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\mathcal{SROIQ}$. Proceedings of KR-2006 pp. 452–457 (2006)
11. Jarrar, M., Demy, J., Meersman, R.: On using conceptual data modeling for ontology engineering. Journal on Data Semantics 1(1), 185–207 (2003)
12. Karima, N., Hammar, K., Hitzler, P.: How to document ontology design patterns. In: Proc. of 7th WS on Ontology Patterns (WOP'16) (2016), 18 Oct, Kobe, Japan
13. Khan, Z.C., Keet, C.M.: Foundational ontology mediation in ROMULUS. In: Fred, A., et al. (eds.) Knowledge Discovery, Knowledge Engineering and Knowledge Management: IC3K 2013 Selected Papers, CCIS, vol. 454, pp. 132–152. Springer (2015)
14. Khan, Z.C., Keet, C.M., Fillottrani, P.R., Cenci, K.: Experimentally motivated transformations for intermodel links between conceptual models. In: Pokorný, J., et al. (eds.) Proc. of ADBIS'16. LNCS, vol. 9809, pp. 104–118. Springer (2016), 28-31 Aug 2016, Prague, Czech Republic
15. Liebig, T., Luther, M., Noppens, O., Wessel, M.: OWLlink. Semantic Web Journal 2(1), 23–32 (2011)
16. Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A.: Ontology library. WonderWeb Deliverable D18 (ver. 1.0, 31-12-2003). (2003)
17. Mossakowski, T., Kutz, O., Codescu, M., Lange, C.: The distributed ontology, modeling and specification language. In: Proc. of 7th Int. WS Modular Ontologies (WoMo'13). CEUR-WS, vol. 1081 (2013), 15 Sept 2013, Corunna, Spain
18. Motik, B., Patel-Schneider, P.F., Parsia, B.: OWL 2 web ontology language structural specification and functional-style syntax. W3c recommendation, W3C (27 Oct 2009), http://www.w3.org/TR/owl2-syntax/
19. Otero-Cerdeira, L., Rodríguez-Martínez, F.J., Gómez-Rodríguez, A.: Ontology matching: A literature review. Expert Syst. Appl. 42, 949–971 (2015)
20. Ritze, D., Meilicke, C., Svab-Zamazal, O., Stuckenschmidt, H.: A pattern-based ontology matching approach for detecting complex correspondences. In: Shvaiko, P., et al. (eds.) Ontology Matching OM-2009 (2009)
21. Scharffe, F., Fensel, D.: Correspondence patterns for ontology alignment. In: Gangemi, A., Euzenat, J. (eds.) Proc. of EKAW'08. LNAI, vol. 5268, pp. 83–92. Springer (2008)
22. Scharffe, F., Zamazal, O., Fensel, D.: Ontology alignment design patterns. Knowledge and Information Systems 40, 1–28 (2014)
23. Schmidt, D., Trojahn, C., Vieira, R.: Analysing top-level and domain ontology alignments from matching systems. In: Ontology Matching OM-2016 (2016), 18 Oct 2016, Kobe, Japan
24. Sugumaran, V., Storey, V.C.: The role of domain ontologies in database design: An ontology management and conceptual modeling environment. ACM TODS 31(3), 1064–1094 (2006)
25. Zamazal, O., Svatek, V.: PatOMat - versatile framework for pattern-based ontology transformation. Computing and Informatics 34, 305–336 (2015)