Semantic Web 0 (0) 1
IOS Press

Referring to multiple unspecified objects of a type: multi-instance fact pattern family [1]

Vojtěch Svátek [a,*], Martin Homola [b], Ján Kľuka [c], and Miroslav Vacura [d]
[a] *Dept. of Information and Knowledge Engineering, University of Economics, Prague*
Nám. W. Churchilla 4, 130 67 Praha 3, Czech Republic
E-mail: svatek@vse.cz
[b] *Department of Applied Informatics, Comenius University in Bratislava*
Mlynská dolina, 842 48 Bratislava, Slovakia
E-mail: homola@fmph.uniba.sk
[c] *Department of Applied Informatics, Comenius University in Bratislava*
Mlynská dolina, 842 48 Bratislava, Slovakia
E-mail: kluka@fmph.uniba.sk
[d] *Dept. of Philosophy, University of Economics, Prague*
Nám. W. Churchilla 4, 130 67 Praha 3, Czech Republic
E-mail: vacuram@vse.cz

Abstract. We introduce the problem of capturing multi-instance facts: modeling situations when a given object is related to multiple unspecified objects of a certain type. We describe the situation in an abstract way (using the PURO ontological background modeling method) and provide three alternative patterns for representing it in OWL. The alternatives make use of different ontology pattern structures: logical structures, naming conventions and annotations; pattern-based invention of new ontology terms is considered aside with reuse of existing terms. The multi-instance fact problem is also aligned with the closest one among the popular logical pattern families, namely, CPV.

Keywords: Ontology pattern, background model, CPV

1. Introduction

A common requirement in data and knowledge modeling is to refer to the existence of entities whose *identity*, but possibly also *number*, remains unspecified; the only available information on these entities is usually their *type* and the specific *relationship* by which an explicitly identified entity is linked to them. This situation most typically arises when the relationship is directed towards something possibly happening in the future: for example, a company *offers to sell* an unspecified number of physical products of some (catalog) type, or some activity *can only be carried out* by agents of a certain type (i.e., whose identity and number is unknown when making the statement).

The most obvious ways of expressing a relationship to anonymous entity/entities in semantic web realms is to employ an existential or cardinality restriction over the respective property (in OWL) or directly model such a (single) anonymous entity as a blank node (in RDF/S, however, without clear semantics) that appears as object in the property instance. Both approaches allow to qualify such an anonymous entity with the class it belongs to. However, these straightforward approaches have certain shortcomings. The semantics of existential quantifier is strictly given by OWL semantics, in the sense that it should not be, under normal

[1] This work has been supported from the VSE IGA project no. 34/2014, from the Slovak VEGA project no. 1/1333/12, and from project APVV-0513-10.
[*] Corresponding author. E-mail: svatek@vse.cz

circumstances, interpreted with other cardinality distinction than 'at least one'; it is thus (psychologically) biased against expressing a relationship to multiple anonymous entities at once. Other cardinality restrictions (on minimum or maximum property instances) do express multiplicity; however, they require an *exact* number of anonymous entities to be specified. A further issue with this kind of modeling is that it is beyond the expressiveness of RDFS, which simpler vocabularies may target; cardinality restrictions with higher arity are even beyond the expressiveness of some OWL profiles. The first issue is also shared by blank nodes, which are, moreover, considered as bad practice by a significant part of the linked data community and have no meaning to the description logic community. In this paper we attempt to go beyond these simple solutions by proposing a family of (three) patterns for the same type of state of affairs (labeled as multi-instance fact), considering:

- possibilities to approximate the *cardinality* of the relationship considered
- pattern design based on *ontological background models* [5]
- generic logical patterns in combination with *naming/annotation* pattern and *vocabulary reuse* considerations
- interconnection to previously published patterns, in particular, the *classes-as-property-values* pattern [2].

While one of the patterns is an extension of the existential restriction approach, the remaining two structurally differ from it in their core.

The paper is structured as follows. Section 2 analyzes the overall situation to be modeled and explains why its OWL representation is not obvious. Short section 3 reviews the inventory available when designing an ontology pattern for OWL. Section 4 presents the three alternative modeling patterns, including axiomatization and examples. Section 5 summarizes the patterns, points out their differences and outlines criteria for choosing among them. Section 6 mentions some real (though implicit) usage of the patterns. Section 7 compares the studied problem with a well known logical design pattern, CPV. Finally, section 8 concludes the paper and drafts the prospects for further research.

2. Multi-instance fact: background model

The modeling situation we target can be characterized as follows: *There is a distinguished real-world entity that is related, by the same kind of relationship, to (possibly even one, but usually) multiple undistinguished objects of a certain type.* We can depict this situation as in Fig. 1.

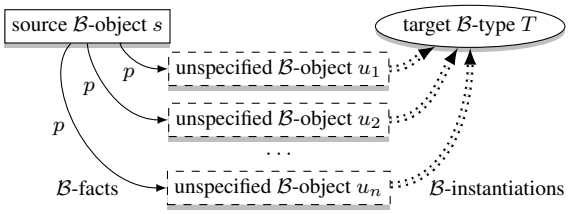

Fig. 1. Abstract (PURO) schema of a multi-instance fact

The diagram conforms to the principles of the *PURO* method of ontological background modeling, introduced in [5,6]. Without going into details of this method, we can summarize that it aims to make explicit the ontological distinctions of 'particular vs. universal' and 'object vs. relationship (between objects) vs. valuation (as quantitative characteristic of an object)' that exist behind existing, or newly designed, OWL ontologies and vocabularies. We presume that these distinctions are mostly correctly sorted out in the heads of ontology designers (as 'background model') but get obscured when the ontology is crafted in OWL (as 'foreground model') as language with limited expressiveness, or because of computational efficiency concerns in logical inference and linked data management.

Understanding PURO in full is however not required for reading this diagram. It simply depicts an unspecified number of 2-chains of relationships, the first always being a 'fact' and the second an 'instantiation' (in PURO terms, we use the notions of $\mathcal{B}$-fact and $\mathcal{B}$-instantiation, so as to stress their 'background' nature and avoid mismatch with the analogous notions in OWL as 'foreground' language); the source object and the target type (again, $\mathcal{B}$-object and $\mathcal{B}$-type) are the same in all chains. Therefore, we propose the whole structure to be called *multi-instance fact* (MIF).[1]

A natural and most 'semantically faithful' choice for representing the source object and target type in OWL would be to use an OWL individual and an OWL class, respectively. Since the explicit representation of the undistinguished entities is not obvious, the mod-

[1] We also happen to colloquially refer to this structure as to '*slingshot* pattern': with some imagination, each of the unspecified objects can be seen as connected by the 'two portions of the rubber strip' (fact and instantiation) to the tips of the Y-handle (source object and target type, respectively), and corresponding to the changing position of the 'ball' during the firing event.

eling problem can be easily perceived as that of connecting an individual (*s*) to a class (*T*); this is however tricky in OWL. The pattern – more precisely, pattern family – providing guidance in the 'classes as property values' (CPV) problem has been first coined by the W3C Ontology Engineering and Patterns a decade ago [2]. In [5] we analyzed the CPV pattern family from [2] in depth (using the PURO apparatus) and identified that its members have different 'affinity' to expressing three different 'states of affairs': relating the individual (1) to an abstract topic derived from the class, (2) to its intension, and (3) to its extension. We will briefly revisit this analysis, specifically for the 'multi-instance fact' problem, in Section 7 of this paper. However, in the main part of the paper, we start from the background state of affairs (as in 1) rather than from a problem manifested on the surface and possibly having different causes.

3. Pattern modeling inventory

Before explaining in detail the members of the proposed pattern family, we will first briefly recapitulate the inventory available for an (OWL-based) ontology pattern designer. Different elements of this inventory will be then employed in the individual alternative MIF patterns in Section 4.

Obviously, first of all, since OWL is a language with rigorous logical semantics, the pattern has to be expressed in *logical terms*. The advent of the semantic web and OWL, during which the ontology pattern repositories such as the W3C SWEO WG collection,[2] the Manchester-based catalog[3] and the Rome-based portal[4] emerged, was dominated by purely logical interpretation of ontology structures. Furthermore, the tiny part of the popular RDF-centric catalog developed for linked data designers[5] that is devoted to ontological (in the sense of OWL T-box) modeling issues, such as the 'N-ary relation pattern', is also confined to the 'logical space'.

However, *naming conventions* gradually came into light in the last couple of years [7,1,3,4]. They can serve not only as guidance for a human inspecting the ontology (and associated data) but also as subject of automated analysis aiming to reconstruct the ontological background (as we refer to it in this paper). Therefore they are prone to make a meaningful part of any ontological pattern design effort.

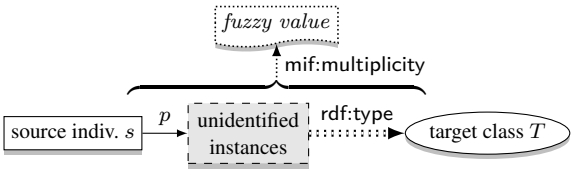

Fig. 2. Schema of the existential restriction MIF pattern

Yet another kind of space for entering important information into OWL ontologies and knowledge bases are *structured annotations*. The topic of extending the expressiveness of OWL ontologies using structured canonical annotations would deserve a separate study. We believe that the provision for rich annotations embedded in OWL 2 has not yet been deployed to its full potential, and that annotations can become an integral part of best practice patterns for ontological modeling.[6]

Finally, rather than advising the pattern users how to design new ontological entities based on generic patterns, it is also possible to let them *reuse specific entities*, either from the pattern itself – which then becomes a 'mini-ontology' in the sense of *content ontology design patterns*[7] – or from existing ontologies and vocabularies, especially those already popular on the linked data web. This reuse may regard not only logical entities but also also annotation entities.

4. MIF pattern family

We outline and exemplify three alternative patterns. While the first one is a simple extension of the baseline existential restriction one, the other two are novel, though already refer to their implicit uses in common ontologies/vocabularies.

4.1. Existential restriction with annotation

The *logical* part of the pattern simply states that the source individual s is an instance of an anonymous class defined by an existential restriction on property p, such that the filler of the restriction is class T. Existential restriction is a construction well anchored in OWL

[2] http://www.w3.org/2001/sw/BestPractices/OEP/

[3] odps.sourceforge.net/

[4] http://ontologydesignpatterns.org/

[5] http://patterns.dataincubator.org/book/

[6] Preliminary ideas for such deployment have been outlined in [8]

[7] http://ontologydesignpatterns.org/wiki/Category:ContentOP

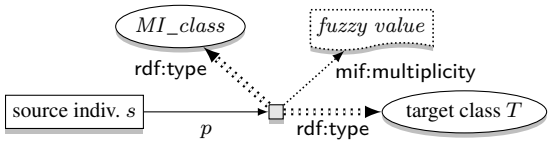

Fig. 3. Schema of the placeholder individual MIF pattern

standard (it is part of even simple dialects of the language[8]) as well as in ontological modeling practice.

As noted in the Introduction, the existential restriction pattern does not allow to express the multiplicity of the relationship to anonymous instances at the *logical* level. *Naming* does not help either. However, in OWL 2 the existential axiom as whole could be *annotated* with a specific annotation property that would indicate that the 'some' keyword (as in the Manchester syntax) actually means, e.g., 'several' or 'quite many' in the specific axiom.

Fig. 2 depicts the inferential product of the axiom rather than its syntactical form:[9] there is a constructed 'skolem' node representing the unindentified instances of T linked from s via property p. The annotation property mif:multiplicity (newly proposed as part of the MIF pattern) indicates that the whole axiom annotated by it is a multi-instance fact, and its value declares the fuzzy (linguistic) cardinality of the set of unindentified instances. This value could be a simple literal (see below) or, possibly a value from a dedicated classification in the future.

The syntactical form of the (extended part of the) pattern in Manchester syntax is as follows:

```
Individual: s
  Types:
    Annotations: mif:multiplicity <value>
    p some T
```

where <value> is the fuzzy value, for instance, "many"^^xsd:String .

4.2. Linking to placeholder individual

When downgrading the previous model into an OWL dialect disallowing existential restriction, the only modeling option at the logical level seems to be a blank node (which, in fact corresponds to the result of evaluation of an existential restriction in tableau reasoning: creation of a 'skolem' individual); via multiple blank nodes, an approximation of the diagram from Fig. 1 could be reconstructed, however, again for a fixed, exact cardinality (as with the higher-arity OWL cardinality restrictions, to whose product this would again correspond). Moreover, using blank nodes outright as truly undistinguished (rather than just unknown to date) entities is odd. A pragmatic approach to explicitly modeling the 'intermediate objects' between the source object and target class of the MIF structure then is to create a (named) individual representing the multiple undistinguished objects as their common *placeholder* (or, possibly, 'proxy'). The placeholder[10] then needs to be classified in two ways:

- First, to it should declare the type of the undistinguished individuals it represents, i.e. link to the target class.
- Second, to avoid being mistaken for a real instance of the target class, it has to be either typed by a dedicated 'utility' class (we call it thereafter as MI_class) unifying all such placeholders, or equipped with an adequate property (presumably, an annotation one).

Furthermore, we would like to be able to express the 'fuzzy multiplicity' as in the previous approach. Note that the multiplicity can now be assigned either to the individual itself or the the MI_class (all of its instances then being placeholders for the 'same' fuzzy number of true instances); we can use the same mif:multiplicity property as we used for axioms in the previous approach. In the following we will however stick to the option of assigning this property to the placeholder individual itself.

The depiction of the pattern is in Fig. 3. Regarding the choice of MI_class, the designer can opt for:

- reusing the generic class from the pattern namespace; we provisionally suggest to call it mif:SomeInstances
- reusing a class from another namespace (below we discuss the gr:SomeItems class from the popular GoodRelations vocabulary)
- coining a proprietary class for the given domain; then the *name* of the class should indicate the multiplicity, for example, in the medical domain one could propose a class called *SomePatients*

[8] This, as well as the fact that exact cardinality statements would be misleading when actually meaning 'vague' multiplicity, is the reason why we do not consider other cardinality restrictions here.

[9] We use this convenient form of depiction so as to remain coherent with [2], at the cost of deviating from the syntactical structure of the anonymous 'existential' class.

[10] For brevity we will omit discussion of relationship of the notion of placeholder here to notions such as 'prototype' in ontological engineering literature.

– omitting the class entirely and only using the annotation property.

The syntactical form of the pattern in Manchester syntax is as follows:

```
Individual: s
  Facts: p _:A
Individual: _:A
  Types: T
  Types: mif:SomeInstances
  Annotations: mif:multiplicity <value>
```

where <value> is again a suitable fuzzy value.

This approach has been chosen as central for one of the most widely used (lightweight) semantic web ontologies, *GoodRelations* (GR). By simplifying the example snippet from the GR specification,[11] we get:

```
foo:offer a gr:Offering;
    gr:includes foo:product .
foo:product a gr:SomeItems;
    gr:name "Canon Rebel T2i (EOS 550D)"@en .
```

In terms of the MIF pattern, foo:product corresponds to p and gr:includes to p. Type T is not present in the snippet, but it could easily be, e.g., class obk:DigitalCamera from the GR-compliant Digital Camera Vocabulary,[12] i.e.:

```
foo:product a obk:DigitalCamera.
```

Although the *SomeItems* class is defined in a domain-specific *vocabulary*, we assume that it is sufficiently generic to be reused beyond the e-commerce (or, even, commercial offering) domain. According to its verbal specification, its instance should be understood as 'a placeholder instance for unknown instances of a mass-produced commodity.' If we relax the adjective 'mass-produced', the class could be suitable for any use case where we consider an undistinguished number of such unknown instances. Reuse of the GR term thus looks as an optimal solution in many cases, aside reusing the more generic mif:SomeInstances class from the MIF pattern or coining a new domain-specific property (but coherently with the MIF pattern). In long term, a natural unifying step might be to declare gr:SomeItems as subclass of mif:SomeInstances,[13] in the GR ontology (though probably not before the MIF pattern gains some reputation, since adding external links to unproven academic artifacts might be harmful to industry-level models such as GR).

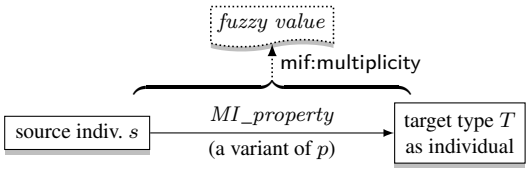

Fig. 4. Schema of the shortcut property MIF pattern

4.3. Shortcut property with name and annotation

The third option is to 'fold' the whole central part of the MIF structure into an instance of a new property, say, $MI\_property$, which directly connects the source individual with the target type. This allows to avoid both anonymous classes and placeholder individuals. On the other hand, the consequences of this variant are:

– The target type can no longer be a syntactic class in OWL-DL: its meta-modeling with an individual is required.[14]
– The new, 'shortcut' property has similar but not identical meaning to the original property p: it differs by its 'multiplicity' and it further includes the instantiation links.

The multiplicity can be indicated and 'quantified' in the same way as in the previous alternatives: using the property mif:multiplicity, this time, however, with a property assertion in its subject. The pattern schema is in Fig. 4. It can be further extended by a similar logical construction as in the placeholder approach: the MI property as such can be declared to be subproperty of a general multi-instance property, mif:multiInstanceProperty.

Traces of this approach can be observed, again, in the GR ontology. It allows one to restrict an offering of a product or service to (undistinguished) possible customers from a certain category, namely to public institutions, as follows:

```
ex:offering231  a  gr:Offering ;
    gr:eligibleCustomerTypes
    gr:PublicInstitution .
ex:PublicInstitution a gr:BusinessEntityType .
```

The OWL individuals ex:offering231 and gr:PublicInstitution correspond to the source individual s and target type T, and the gr:eligibleCustomerTypes

[11] http://www.heppnetz.de/ontologies/goodrelations/v1.html#SomeItems

[12] http://purl.org/opdm/digitalcamera

[13] It is worth mentioning that the deprecated predecessor of 'SomeItems' is 'ProductOrServicesSomeInstancesPlaceholder', i.e., it already including 'SomeInstances' in its string.

[14] At least in the form of OWL 2 punning.

property assertion is the shortcut property: what is actually meant here is that the eligible customers are the unspecified number of unknown instances of gr:PublicInstitution. The implicit *naming convention* is the 'Types' suffix in the property name, indicating that the background of the property is rather a chain of a specific relationship ('being eligible for an offer' – or, rather, its inverse form) and the generic instantiation relationship ('having a certain type'). However, in the MIF pattern we would rather suggest to keep the suffix in singular, i.e. '-Type' as suffix, as *naming pattern*.

The Manchester syntax form of the logical part of the pattern is as follows:

```
Individual: s
  Facts:
    Annotations: mif:multiplicity <value>
    MI_Property T
```

given again some suitable fuzzy value <value>.

5. Overview and selection criteria

Table 5 summarizes what the different approaches reuse from the MIF pattern entities and what some other consequences of using the pattern are.

Clearly, alternative 1 (existential restriction) is most attractive when designing ontologies that are assumed to be processed by OWL-aware tools. The logical part of the pattern is sound for them even without understanding the annotation properties. Alternative 2 (placeholder) is suitable when we want to declare that the unspecified individuals share, in addition to type *T*, also various features (as it is the case for mass-produced commodities) expressible by property assertions. Finally, alternative 3 (shortcut property) is most parsimonious in terms of knowledge base size. It only fits when we do not care for additional properties of the unspecified objects.

6. Implicit pattern usage

As mentioned in the patterns explanation, the latter two (i.e. less obvious) alternatives are used, although without the annotation apparatus, in the GoodRelations ontology. According to the GoodRelations statistics service,[15], the placeholder class gr:SomeItems is, at the time of writing this article (June 2014) used nearly 86 thousand times in the observed fragment of the Linked Open Commerce dataset;[16] entities corresponding to the property variant pattern such as gr:eligibleCustomerTypes or gr:acceptedPaymentMethods (considering that the method by which a payment action is carried out can be viewed as a 'type' of this action) are however not tracked by the statistics and probably not extensively used so far.

We are confident that various occurrences of the pattern could be found in numerous other ontologies and vocabularies. An empirical study of these phenomena is ongoing.

7. Relationship to the CPV family

Although the 'shortcut link' in the MIF structure clearly refers to the CPV problem, there are some differences from the way the CPV problem has been analyzed in [2].

First, MIF only amounts to one of the three subcategories of CPV, as identified in [5]: that of relating the individual to the *extension* of the class. Only the Approach 4 of [2] – using an (existential) property restriction with the target class as filler – fully conforms to this sub-category. We can rewrite the gist of the Approach 4 A-box example in Manchester syntax as:

```
LionsLifeInThePrideBook dc:subject some Lion
```

i.e. the book 'Lions: Life in the Pride' has as subject some (at least one) unindentified lion. Second, [2] devotes specific attention to the problem of considering the target class inside of a taxonomy, which is beyond the scope of our MIF pattern analysis. On the other hand, as reflecting in its name, the MIF pattern (family) specifically accounts for the *multiplicity* of the relationship between the source object and target class; therefore, we had to extend even Approach 4 (of [2]) by a 'multiplicity' annotation.

8. Conclusions

We described the modeling problem denoted as *multi-instance fact* (partly with the help of the recently proposed PURO background modeling method) and provided basic descriptions of three kinds of pat-

[15] http://goodrelations-stats.appspot.com/

[16] This dataset contains data extracted from web pages containing e-commerce information.

Approach	Reused entities	Consequences
Existential	mif:multiplicity	OWL expressiveness required in logical part
Placeholder	mif:multiplicity mif:SomeInstances	Placeholder individuals may interfere with true ones
Shortcut property	mif:multiplicity mif:multiInstanceProperty	Type *T* meta-modeled by individual Additional, non-obvious variant of property *p*

Table 1
Overview of the alternative patterns

terns for expressing it in OWL. The pattern is, in a way, hybrid: it contains both a logical/structural description based on examples (as common for the W3C SWEO patterns) and a proposal for reusable entities within a common namespace[17] (mif:multiplicity property, mif:SomeInstances class and mif:multiInstanceProperty). We also positioned the new problem and patterns with respect to the closest related effort in the ontology pattern field: representing classes as property values.

The work presented in the paper is not a pattern proposal ready to be immediately published as guidance for ontology designers. It is rather a focused analysis from which a recommendation such as [2] could be distilled relatively easily. The principles seem clear; however, a discussion of a larger community is required so as to decide about, at least (1) the priority order in which the three options should be presented to the reader; (2) the precise form of naming conventions as well as rich annotations – since the number of options is quite large here. Of course, novel proposals for solutions addressing the problem described here may arrive as well.

While we justified the importance of the modeling problem by pointing at its occurrence in the e-commerce domain, additional investigation is still needed in further domains.

References

[1] N. A. Abdul Manaf, S. Bechhofer, R. Stevens, A Survey of Identifiers and Labels in OWL Ontologies, in: OWLED 2010.

[2] N. Noy (ed.), *Representing Classes As Property Values on the Semantic Web.*, W3C Working Group Note 5 April 2005, online at `http://www.w3.org/TR/swbp-classes-as-values/`.

[3] D. Schober, B. Smith, S. E. Lewis, W. Kusnierczyk, J. Lomax, C. Mungall, C. F. Taylor, P. Rocca-Serra, S.-A. Sansone, Survey-based naming conventions for use in OBO Foundry ontology development, *BMC Bioinformatics* 10 (2009).

[17] The URI for this namespace is yet to be set up.

[4] D. Schober, I. Tudose, V. Svátek, M. Boeker, OntoCheck: verifying ontology naming conventions and metadata completeness in Protégé 4. *J. Biomedical Semantics* 3(S-2): S4 (2012).

[5] V. Svátek, M. Homola, J. Kl'uka and M. Vacura, Mapping Structural Design Patterns in OWL to Ontological Background Models, in: Proc. K-CAP 2013, ACM, 2013, pp. 117–120.

[6] V. Svátek, M. Homola, J. Kl'uka and M. Vacura, Metamodeling-Based Coherence Checking of OWL Vocabulary Background Models, in: Proc. OWLED 2013, online `http://ceur-ws.org/Vol-1080/owled2013_6.pdf`.

[7] V. Svátek, O. Šváb-Zamazal, V. Presutti, Ontology Naming Pattern Sauce for (Human and Computer) Gourmets, in: Workshop on Ontology Patterns at ISWC'09, Washington DC, 2009.

[8] V. Svátek, M. Vacura, M. Ralbovský, O. Šváb-Zamazal, B. Parsia, OWL Support for (Some) Non-Deductive Scenarios of Ontology Usage, in: Proc. OWLED 2008