A Metaontology for Ontological Engineering. 
A Philosophers’ Perspective

Pawel Garbacz Robert Trypuz
The John Paul II Catholic University of Lublin

Abstract. The paper defines a schema in which to describe applied ontologies of any kind. In contradistinction to the main trend in engineering metaontology the main ideas that support this schema are inspired by philosophy. Namely, we look at the domain of applied ontologies from the point of view of a certain metaphilosophical tradition. Nevertheless, the final result of our considerations is an engineering artefact, i.e., an OWL ontology. For the sake of validation we employ it to describe a few exemplary applied ontologies.

Keywords. metaontology, applied ontology, philosophy

Introduction

One of the symptoms for the maturity in any branch of knowledge is the existence of its methodological framework, which describes, among other things, this specific kind of knowledge. Although the field of applied ontology is relatively young, we have recently witnessed a growing awareness of its needs, goals, specific methods, types of approaches, etc. This methodological self-awareness has brought about a number of research results at the meta level, e.g., engineering methodologies, classifications of applied ontologies, etc., which focus not on the structures represented by applied ontologies, but on the very representations themselves.

The majority of the ideas that shape this metaontological research is of the engineering or applicative nature. This predominance seems to be unbalanced if we compare it to the scope of theoretical research on the ontological level. The domain of applied ontology is famous (or infamous) for being prone to the interdisciplinary influences, including those coming from fairly exotic domains such as philosophy. Our paper aims to extend the scope of the theoretical reflection in engineering metaontology. Namely, we present a metaontological schema for characterisation of applied ontologies, which schema is inspired by a certain metaphilosophical understanding of the notion of ontology. In other words, the novelty of our approach consists in a new perspective, from which we attempt to describe applied ontologies as if they were philosophical systems of thought.

1Corresponding Author: Pawel Garbacz, e-mail: garbacz@kul.pl
In section 1 we very briefly report the main theoretical and practical results in engineering metaontology. This outline is definitely too sketchy as a survey of the state-of-the-art research, but its main function is to set up the stage for our own contribution, which is presented in section 2. The last section shows how our metaontological conception may be implemented as an engineering artefact.

1. Related Work

The metaontological research in applied ontology does not match the scope and richness of existing ontological artefacts. The major part of the former endeavour revolves around adequate definition of the notion of applied ontology and subsequent classification(s) thereof.

1.1. Engineering Classifications of Applied Ontologies

Our survey of engineering metaontology showed that there are at least five orthogonal ways of classifying applied ontologies:

1. with respect to the “ontological type” of an applied ontology;
2. with respect to the domain of an applied ontology;
3. with respect to the formal type of an applied ontology’s language;
4. with respect to the complexity of the structure of an applied ontology;
5. with respect to the intended usage of an applied ontology.²

“Ontological type”-based classifications An example of the first type of classification is presented in [5]. N. Guarino and P. Giaretta describe two meanings for the term “ontology”:

- ontology as a logical theory (of a certain type)
- ontology as a conceptualisation.

An applied ontology construed as a logical theory refers to an “engineering artefact, constituted by a specific vocabulary used to describe a certain reality, plus a set of explicit assumptions regarding the intended meaning of the vocabulary words” [4, p. 4]. Its content should be rich enough so that the ontology could provide the meanings of its non-logical terms. If the ontology construed as a logical theory is created in the context of the model theory, it is usually provided with the model-theoretic structures it represents. Guarino and Giaretta claim that these formal structures themselves may be considered as ontologies although only in the derivative sense. In this sense they are ontologies construed as conceptualisations. Ontologies as conceptualisations may be built by means of set theory, however there are other formal means available. Ontologies as conceptualisations can be seen as semantic counterparts of the ontologies as logical theories.

The requirement that an applied ontology needs to be a logical theory seems to be too restrictive as far as the common use of this term is concerned. Its usage is rather relaxed, so the above distinction may be generalised to the distinction

²For the reasons of space, describing these types we are forced to give only one representative example for each type.
between ontologies construed as representing artefacts and ontologies construed as represented structures. The former are bound to be stored in a medium of some sort, which usually takes the form of a formal language.

Language-based classifications  This observation leads us to the second type of ontology classifications, which focuses on the formal properties of the medium in which the ontology is rendered. For instance according to [20] an applied ontology can be:

- rigorously formal – when it is expressed in a formal language equipped with formal semantics (and possibly accompanied with some metalogical results);
- semi-formal – when it is expressed in an artificial and formally defined language (e.g., XML);
- semi-informal – when it is expressed in a structured and/or restricted ethnic language;
- highly informal – when it is expressed in an ethnic language.

Complexity-based classifications  A language-based classification may be confused with a classification that takes into account the complexity of an applied ontology’s structure. Compiling a number of proposals (cf. [22,8,16,19]) we have collated the following list of different types of ontologies:

- controlled vocabularies (e.g., catalogues), which are usually simple lists of terms accompanied with GUIDs (general unique identifiers) for the represented entities;
- glossaries, i.e., lists of terms accompanied with some natural language descriptions of terms (possibly also with GUIDs),
- thesauri, which extend glossaries with relations between listed terms (e.g., with synonymy);
- informal “is-a” hierarchies, which may not support inheritance;
- taxonomies, i.e., formal “is-a” hierarchies supporting inheritance;
- taxonomies with the “instance-of” relation;
- frame-based ontologies, which specify other relations between objects;
- ontologies with explicit value restrictions imposed on their relations;
- ontologies with certain set-theoretical constraints on their categories and relations (e.g. disjointness, transitivity, etc.);
- proper axiomatic theories.

Domain-based classifications  A classification of applied ontologies with respect to the domain takes into account what kind of concepts an applied ontology specifies. Let us focus on just one example. N. Guarino distinguishes in [4] between:

- top-level ontology;
- domain-level ontology;
- task ontology;
- application ontology.

A top-level ontology contains only the most general concepts like “property” or “event” without any bias towards any particular engineering problem or application. On the other hand, the domain-level ontology provides the vocabulary
related to a generic domain (e.g., the domain of web services). A task ontology specifies the concepts related to a generic task or activity like translating (a text) or buying (a product). Finally, the concepts from an application ontology depend both on a particular domain and a particular task. In fact, this example contains certain elements pertaining to the next type of classifications.

Use-based classifications An example of ontology classification with respect to usage can be found in [11]. The authors consider there a typology of ontologies with respect to a way of an engineering ontology is applied within a particular context. The context itself is defined by a need to share and/or reuse a piece of knowledge represented in a computer system. [11] distinguishes:

- content ontologies;
- communication (tell&ask) ontologies;
- indexing ontologies.

A content ontology is an applied ontology that is used directly in an event of knowledge sharing. Thus, any content ontology must be reusable, i.e., suitable for being a component of other ontologies, but it may be not portable: it might require a human agent whose task is (i) to interpret/understand it and (ii) to code the executable programs in which it is applied. Its behaviour may be compared to the function of computer algorithm, which is reusable but not portable. The category of content ontologies is further divided into:

- task ontologies, which specify problem solving processes, in particular they determine what objects and relations are necessary for performing the engineering task in question;
- workplace or environment ontologies (cf. [21]), which specify the environment in which the problem solving process is embedded;
- domain ontologies whose task is to represent the specific features of a given fragment of reality;
- common sense ontologies whose task is to represent the generic features of any fragment of reality from the common sense point of view.

A communication (tell&ask) ontology is understood by analogy to an input/output black box since its users need not know its specific structure and content. Its task is to assist the user in his or her search for those agents who can solve the user’s specific problem. Consequently, any communication ontology (i) is focused on how to query a specific agent for the required information and (ii) is “responsible” for making this communication seamless.

An indexing ontology boils down to a vocabulary of index cases stored in the so-called case base. Its task is to assist in those knowledge sharing processes that are based on the cased-based reasoning.

1.2. OMV as a Formal Engineering Metaontology

Although the research on the metaontological level is not predominant in engineering ontology, there has been several initiatives that go beyond simple definitions or classifications of ontologies. For example, a number of schemas or standards for representing meta-information are already available, e.g., the Dublin Core stan-
Ontology Metadata Vocabulary (OMV) is one of the most recent proposals in this respect (see [7], [13]). OMV is a formal metaontological framework in which to describe applied ontologies from the point of view of knowledge representation. Currently it consists of two modules OMV Core and OMV Extensions, which deal, respectively, with the general aspects of engineering ontologies and with the task- and application-specific ontology-related information. We will focus here on the former module.

The main conceptual distinction drawn within OMV Core separates ontology bases from ontology documents.

An Ontology Base (OB) represents the abstract or core idea of an ontology, so called conceptualisation. It describes the core properties of an ontology, independent from any implementation details. [...] An Ontology Document (OD) represents a specific realization of an ontology base. Therefore, it describes properties of an ontology that are related to the realization or implementation. [7, p. 908]

Needless to say, the properties of an ontology base may be different from the properties of those ontology documents that implement it. Both are characterised by the property that represents ontology’s authorship and the property that represents license models utilised in ontology’s development. An ontology document is further described by:

1. the engineering methods and software tools that were used by the authors of the ontology documents;
2. the language in which the ontology document is expressed;a
3. the type of the ontology document (e.g., upper-level ontology, domain ontology, etc.);
4. the ontology’s domain and task specifications.

The whole OMV metaontology is rendered in OWL with a number of “pre-defined” instances and datatype properties’ values.

2. From Metaphilosophy to Metaontological Engineering

Our goal is to provide a metaontological schema in which to detail the philosophically outstanding aspects of engineering ontologies. We recognise our approach as complementary to the current engineering metaontology. Therefore, for instance, our schema does not include any information on the engineering aspects of applied ontologies, e.g., the specification of the software used in ontology development. The methodology we followed starts from the search for a conception of metaphilosophical description of philosophical ontology. The most adequate description found is then adapted to the specific features and needs of applied ontologies.

---
aOMV distinguishes between a formal language and its syntax, however, the precise import of this distinction is not further elaborated.
Our starting point is the metaontological schema developed by Antoni B. Stepien for the sake of (metaphilosophical) analysis of philosophical ontologies (cf. [17]). He argues that the adequate description of a given philosophical ontology requires four components:

1. characterisation of the sources of ontological knowledge and the methods used in the development of the ontology (e.g., the so-called intellectual insight, phenomenological ideation, abstraction, linguistic analysis, formal deduction, etc.);
2. explanation of the concept of being that is presupposed or implied by the ontology (e.g., being as something definite, being as a possible entity, being as anything one can think of, etc.);
3. list of the main ontological categories together with the relations therebetween (e.g., endurants, properties, parthood, etc.);
4. description of the role or function of the ontology in the broad philosophical framework, e.g., whether the ontology in question provides a foundation for a theory in ethics.

We consider an applied ontology as an epistemic artefact whose main role is to represent a certain domain in such a way that a properly configured computer system is capable of storing and processing this representation. This definition is relatively broad as it covers all kinds of meaningful documents, e.g., even MS Word documents provided that they represent something in a computer-readable way. This interpretation presupposes the primary sense of “ontology” mentioned in section 1.1, i.e., we construe an applied ontology as a representation (of a certain structure) and not as the represented structure itself.

In order to adapt Stepien’s conception for the purposes of knowledge engineering we transformed it into the following schema. An applied ontology is described there by means of:

1. description of its medium, usually the formal language in which the ontology is rendered;
2. description of the methodology used in its development;
3. description of the sources of ontological knowledge utilised; this usually will include various characterisations of the cognitive, psychological, philosophical ideas that shape the main traits of the ontology;
4. description of the ontology’s domain;
5. list of the ontology’s categories (if any);
6. description of actual and/or possible (intended) applications of the ontology in an information system.

**Language** We do not distinguish between ontological conceptualisations and ontological realisations/implementation (as OMV does) because usually even minor changes in the way an “abstract” idea is formulated lead to different views on the domain in question. In particular, when an ontology is rendered in two formal languages, say, the full first-order logic and some weak description logic language like OWL DL, the two formalisations are in fact two different theories due to the difference in the expressivity of their languages. Consequently, each will represent its domain in a slightly (or, as it may happen, radically) different
way. The resulting differences are of particular importance when a ontology in question is to disambiguate the natural language discourse for the sake of, say, semantic negotiations within a network of agents. Then even minor changes in the formalisation may result in significant semantic discrepancies and consequently in negotiation failure. This “metaontological decision” forces us to include in our schema the description of the language the ontology uses.

We adopted the classic understanding of language by means of which a language is construed as a triad constituted by the syntactic, semantic and pragmatic aspects (cf. [12]). Therefore, we cannot incorporate the OMV distinction between a language and its syntax as our notion of language includes its syntactical features. For instance, OWL/XML and OWL Manchester Syntax are understood here as two different formal languages. We believe that one needs to draw an important distinction in the domain of ontological languages that differentiates between the languages that support formal inference and the languages that do not. By an inference we understand any cognitive process whereby a new piece of knowledge is obtained on the basis of some previously acquainted knowledge. An inference is formal if its validity depends on the structures of its premises and conclusion and not on their content. For instance, if an ontology is a simple list of objects rendered as an CSV (comma-separated values) file, no inference within this ontology is supported by the CSV format.

Methodology

As for the description of the methodology used, we restrict this aspect to a limited number of well-developed engineering methodologies, e.g., OntoClean [6], METHONTOLOGY [3], DILIGENT [14], etc. We treat the collection of engineering methodologies as a homogenous whole. In particular, we do not divide them into the static methodologies that describe the preferred structure of an applied ontology and the dynamic methodologies that prescribe the preferred sequence of ontology development.

Sources of ontological knowledge

On the other hand, we do separate the methodological aspect from the epistemological aspect of an applied ontology, which is represented here by means of the description of source of ontological knowledge. The role of this description is to document the ontological choices made when the ontology was developed. In the case of general categories or schemas the ontological knowledge may come from some general scientific ideas, conceptions, or theories. Among those categories we put special emphasis on the philosophical assumptions and/or implications of a given ontology. For the sake of clarity, we organise the general sources of knowledge according to the top-most categories of the Dewey Decimal Classification, however, this assumption is not crucial to our approach and may be dropped in favour of some other classification. The variety of the domain-specific sources of knowledge cannot be so easily tamed. The specific ontological knowledge may originate from the user’s requirement specifications, existing documentation for the legacy data systems, logical schemas of the databases to be integrated by the applied ontology in question, interviews with the subject-domain experts, etc.

Domain

The description of the domain of an applied ontology corresponds to philosophical explanation of the concept of being. In our opinion the comprehensive ontological characteristic of the latter notion is not necessary for our pur-
poses, i.e., we do not need to answer the ontological question “What is the domain (of an applied ontology)?”. However, we did find it useful to associate with each such domain a set of all objects that exist therein. The (intended) extension of the domain of an applied ontology is understood here as a set of objects that are believed by the authors’ of this ontology to fall under one of the ontology’s categories provided that the ontology has any categories at all. When the applied ontology in question contains only specific data without any general schemas or categories, then the (intended) extension of the domain is the set of all entities represented by the ontology’s components (e.g., GUIDs). The latter case is to cover the borderline cases of applied ontologies in the form of flat lists or catalogues.

**Categories** The description of the domain leads us to the list the ontology’s categories, including those expressible by n-ary \((n \geq 2)\) predicates. As before, we would like to keep the notion of ontological category rather informal focusing on, but not reducing to, its extensional aspect. The intended extension of an applied ontology’s category is understood here a set of objects that are believed by the authors of this ontology to fall under this category. Such extensions may be recognised by means of the ontology’s axioms (if any), the informal explanations of the categories provided by the ontology’s developers in the documentation of the ontology (if it has been produced), and the particular instances of those categories given as examples (if any).

Obviously, if the list of an applied ontology’s categories is not empty, then the intended extension of its domain is equal to the set-theoretical union of the intended extensions of those categories.\(^4\) This purely extensional description of the domain may not be sufficient or handy for some metaontological purposes. Therefore, we extend it with a list of the so-called metaontological “flags” that explicitly characterise philosophically salient aspects of the domain. Currently, the list contains only two items:

1. modality flag:
   
   (a) actual - if the domain contains only actual entities, i.e., those that either existed, exist, or will exist;
   
   (b) possible - if the domain contains also possible entities, e.g., unicorns;
   
   this flag covers also the actual entities;

2. objectivity flag:
   
   (a) mind-independent - if the domain contains only mind independent entities,

---

\(^4\)To be more precise, since the list of categories may include both unary and \(n\)-ary \((n \geq 2)\) predicates, the intended extension of the domain of an applied ontology is equal to the set-theoretical union of the intended extensions of the ontology’s unary categories and the *fields* of the intended interpretations of the ontology’s \(n\)-ary categories. However, if the field of any relation in an ontology is not included in the set-theoretical union of the (intended) extensions of the ontology’s unary categories, then we find this ontology as an example of inadequate ontological engineering.
(b) mind-dependent - if the domain contains only mind dependent entities, i.e., those whose existence constantly depends on the existence of (the beliefs, desires, or intentions, etc. of) some particular agent;
(c) mixed - if the domain contains entities of both kinds.

Any flag value is to identified solely on the basis of the ontology’s developers declarations if provided in the specification of the ontology.

We found the idea of domain flags useful, for instance, as a means to characterise those aspects of applied ontologies that are the subject of the realism/anti-realism debate (see the recent exchange between [10] and [15]).

Since we provide neither a theory of ontological categories nor even their criteria of identity, we are not in a position to say that category $c_1$ from ontology $O_1$ is identical to category $c_2$ from ontology $O_2$. However, in order to be able to draw at least some comparisons between different ontological categories we will use the following relation of equivalence: the intended extension of $c_1$ is identical to the intended extension of $c_2$. The relation of “owl:equivalentClass” is a special case of this relation.

Note that including both the description of the domain of an applied ontology and the list of its categories we are able identify not only which portion of reality is represented by the ontology, but also on what aspects or features of this portion the ontology focuses. Since neither the notion of domain nor the notion of ontological category is reduced to the notion of set, our schema allows for a number of intensional interpretations (as, for example, defined in [4]).

Sometimes the authors of an applied ontology explicitly intend to exclude certain entities from its domain. Therefore, we posit that the list of the ontology’s categories may be accompanied by the list of the rejected (i.e., believed by the authors to be empty) categories.

*Use* Finally, our description of the ontology’s applications is composed of two parts. The first part specifies the intended type of the ontology’s usage. To this end, we currently employ the concepts and terminology defined in [11]. The second component describes the actual use of the ontology in a particular instance of computer system.

Among the six aspects of applied ontologies listed on page 6 we find the descriptions of the ontology’s language and domain (mentioned as items 1 and 4 in the list) mandatory for any adequate characterisation of this ontology. The list of categories (item 5 in the list) is also mandatory if the ontology in question has any categories at all. This means that any metaontological description of an applied ontology that lacks any of these components does not sufficiently characterise this ontology as an epistemic (representational) artefact. Other aspects are optional, i.e., a metaontological description of an applied ontology is not claimed to be defective if it lacks any of them.

3. Implementation

Our metaontological schema is not restricted to any specific (meta-)language although we reckon the natural/ethnic languages to be the most flexible tool fit for
this purpose. Nonetheless, because we engaged ourselves in a discipline of applied research, we built an OWL ontology that incorporates the main components of our schema so that the philosophical perspective we provide will become computationally operable. We consider this ontology as a corner stone of a prospective repository of applied ontologies to be built.

The OWL metaontology incorporating the main ideas of our conceptual schema is an OWL DL ontology with the SIF(D) expressivity. It contains 24 OWL classes with seven top-level categories:

1. owl:Class rdf:ID="Category"
2. owl:Class rdf:ID="Domain"
3. owl:Class rdf:ID="Language"
4. owl:Class rdf:ID="Methodology"
5. owl:Class rdf:ID="Ontology"
6. owl:Class rdf:ID="Source_of_Ontological_Knowledge"
7. owl:Class rdf:ID="Usage"

These classes correspond to the items in the list from page 6. Moreover, the OWL metaontology contains eight OWL object properties:

1. owl:ObjectProperty rdf:ID="acknowledges"
2. owl:ObjectProperty rdf:ID="rejects"
3. owl:ObjectProperty rdf:ID="represents"
4. owl:ObjectProperty rdf:ID="isDevelopedWithin"
5. owl:ObjectProperty rdf:ID="isExpressedIn"
6. owl:ObjectProperty rdf:ID="isInformedBy"
7. owl:ObjectProperty rdf:ID="isIntendedAs"
8. owl:ObjectProperty rdf:ID="isExtensionallyEquivalentTo"

The role of the first seven properties is to relate the “Ontology” class to other top-level classes in the metaontology - see figure 1. The last property is to relate the categories of one ontology to the relevant categories of another ontology. Finally, we use five OWL datatype properties, mainly for storing the URIs for the documents representing the objects in the metaontology. Note that we maintain the difference between the relation denoted by “isDefinedBy” and the relation denoted by “isDocumentedIn”. The former relates certain informational objects such languages and methodologies to those informational resources that provide their descriptions. The latter relates sources of knowledge to those informational resources that store them.

The OWL metaontology (currently) includes also 69 “predefined” instances, e.g., several types of languages are instances of the “Language” OWL class. Of course, while OWL classes and properties are to form a (relatively) stable part of the metaontology, the collection of its instances is bound to grow over time. The whole ontology is available at www.13g.pl. Using the OWL metaontology we described three examples of applied ontologies of rather different types:

1. DOLCE [9]
2. CIDOC CRM ontology [2]
3. FAO geopolitical ontology [18]
Figure 1. UML model of the OWL metaontology

Needless to say, except perhaps for the list of an applied ontology's categories all other components of our metaontological schema must be filled in manually.

We regard the three descriptions above as a proof of concept for our metaontological investigation, however, due to the lack of space we could not specify them here in any detail, so the interested reader should consult www.l3g.pl.

4. Conclusions and Further Work

It seems to us that our paper supports the claim that metaphilosophy may provide a number of non-trivial insights for engineering metaontology. Compared to the existing metaontological standards (e.g., to OMV) our schema reveals the need for a proper specifications of the sources of ontological knowledge. Secondly, the schema underlines the representational aspects of applied ontologies by means of the description of domain and the list of ontological categories.

We believe that further development of this initial proposal may lead to the idea of a repository of applied ontologies with an extended list of functionalities. We think, in particular, about queries related to the conceptual foundations of applied ontologies or queries for applied ontologies containing a queried list of ontological categories.

Acknowledgements

Both authors were supported by the grant N N101 150037 from the Ministry of Science and Higher Education and the EuJoint project funded by the IRSES Schema of the Seventh Framework Programme (FP7-PEOPLE-2009-IRSES, no 247503).
References


