An Analysis of Commitments in Ontology Language Design

Pablo R. FILLOTTRANI [a,b] and C. Maria KEET [c,1]

[a] *Department of Computer Science and Engineering, Universidad Nacional del Sur, Argentina*

[b] *Comisión de Investigaciones Científicas de la Provincia de Buenos Aires, Argentina*

[c] *Department of Computer Science, University of Cape Town, South Africa*

Abstract. Multiple ontology languages have been developed over the years, which brings afore two key components: how to select the appropriate language for the task at hand and language design itself. This engineering step entails examining the ontological 'commitments' embedded into the language, which, in turn, demands for an insight into what the effects of philosophical viewpoints may be on the design of a representation language. But what are the sort of commitments one should be able to choose from that have an underlying philosophical point of view, and which philosophical stances have a knock-on effect on the specification or selection of an ontology language? In this paper, we provide a first step towards answering these questions. We identify and analyse ontological commitments embedded in logics, or that could be, and show that they have been taken in well-known ontology languages. This contributes to reflecting on the language as enabler or inhibitor to formally characterising an ontology or an ontological investigation, as well as the design of new ontology languages following the proposed design process.

Keywords. Ontology Language Design, Ontology Engineering, Conceptual Modelling, Ontological Foundations

1. Introduction

Ontology engineering aims to study and develop methodologies, tools, and languages that support building ontologies to be used in information systems [1,2,3]. Determining which formal ontology language to use to represent the intended meaning is, or ought to be, a key decision in this process [4]. This because, firstly, it helps in understanding the nuances of the interpretation, facilitates addressing possible ambiguities, and enables capturing the intended meaning as precise as possible. Secondly, the ontology language interacts with the tools so that such semantic descriptions can be automatically processed, including services such as consistency checking, and be integrated with other ontologies or become part of an information system.

Several ontology languages have been used for this purpose, which were either forked from previous knowledge representation languages and their applications or specifically designed for ontologies. This brings afore two key components: selecting the

[1]Corresponding Author E-mail: mkeet@cs.uct.ac.za

appropriate language for the task at hand and language design itself. Making the right decision for language design or its use requires gathering information, specifying requirements, identifying alternative solutions, and weighting pros and cons. This entails one has to examine the 'commitments' each language has embedded in it, so as to make it clear what the choices for existing languages are and to eventually justify a possible need for designing a new ontology language. Examples of philosophical commitments are whether the language should have a separate type of element for attributes (e.g., alike the OWL data properties), whether parthood should be a primitive, and whether the world is crisp or inherently vague. Such choices in the design of an ontology language, or which have been made for the modeller already, are fundamental in the way the language enables or obstructs representing reality. Ignorance of limitations assures that no change towards proper correction is possible. Two common examples that have been investigated and shown to affect modelling ability and precision are the improved understanding by the modeller with look-here vs. look-across syntax notation for n-ary relationships [5] and the increased use and disambiguation of part-whole relations when it is a primitive in the language [6].

This raises the question: *what are the sort of commitments one should be able to choose from in language design or selection, which have an underlying philosophical point of view?* And, taking a step back: *which philosophical stances have a knock-on effect on the specification or selection of an ontology language?* In this paper, we provide an initial step towards answering these questions. We identify and analyse ontological commitments embedded in logics, or that could be, and show that they have been taken in well-known ontology languages. They include, among others, crisp vs. vague, 3- or 4-dimensionalist, the 'fundamental furniture' (basic building blocks, also called "epistemological primitives" [7]) and typical possible refinements, and a logic's interaction with natural language. To position this in an overall engineering process and take a step to make it actionable for ontologists, we framed it in a process of language design and devised a requirements catalogue aimed to help identify or select commitments for a language. We hope that this will provide the ontology engineer with better means to decide the language to represent an ontology, design a new one, or at least be aware of the commitments they have selected.

We start by describing the steps in the process of language design in Section 2, in order to acknowledge the different stages that are involved, since they also interact and affect each other. Then in Section 3, we identify the choices that are made and analyse their consequences. In Section 4 we assess several key ontology languages in view of these commitments, and present a use case of the the application of the design process for a new ontology language. Finally, we present some conclusions in Section 5.

2. Language Design for Ontologies

The design of a logic (representation language) to develop ontologies or to formally characterise the ontological nature of the topic under investigation may be seen as an engineering task, just like the development of an artefact. That is, a process with several steps from start to finish, such as requirements engineering and testing. For instance, requirements may be that the logic has to be decidable in subsumption reasoning, that it needs to have *n*-ary relationships (with $n \geq 2$), and that it must have support for modularity;

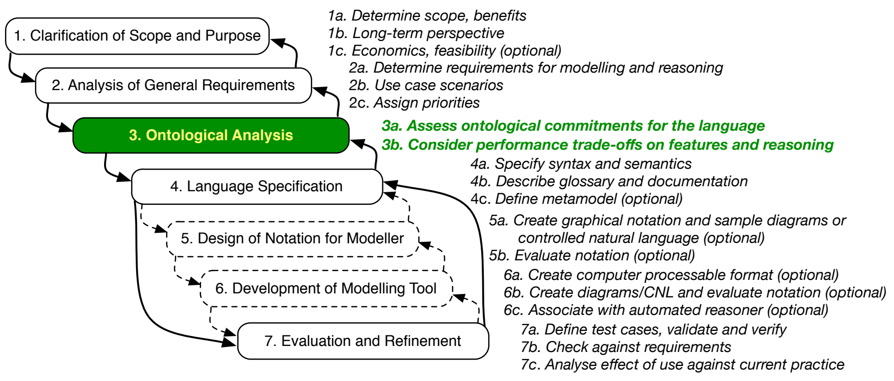
1. Clarification of Scope and Purpose
2. Analysis of General Requirements
3. Ontological Analysis
4. Language Specification
5. Design of Notation for Modeller
6. Development of Modelling Tool
7. Evaluation and Refinement
1a. Determine scope, benefits
1b. Long-term perspective
1c. Economics, feasibility (optional)
2a. Determine requirements for modelling and reasoning
2b. Use case scenarios
2c. Assign priorities
3a. Assess ontological commitments for the language
3b. Consider performance trade-offs on features and reasoning
4a. Specify syntax and semantics
4b. Describe glossary and documentation
4c. Define metamodel (optional)
5a. Create graphical notation and sample diagrams or controlled natural language (optional)
5b. Evaluate notation (optional)
6a. Create computer processable format (optional)
6b. Create diagrams/CNL and evaluate notation (optional)
6c. Associate with automated reasoner (optional)
7a. Define test cases, validate and verify
7b. Check against requirements
7c. Analyse effect of use against current practice

Figure 1. Development steps for the creation of a logic for ontologies, where the focus of this paper is highlighted in bold and shaded (green); the two dashed stages are optional, depending on what has been specified as scope in step 1.

see, e.g., the set of requirements and goals for OWL [8] and the use cases and feature set of DOL [9]. To the best of our knowledge, there is no existing methodology to design a logic. A design pipeline does exist for domain-specific languages (DSLs), notably as outlined by Frank [10], and there are a few works that focus on one aspect of the process, such as the requirements engineering step [11], or that propose guidelines tailored to one class of modelling language specifically (e.g., DSLs [12]). For the purpose of designing a logic for ontologies in particular, we modified Frank's "macro-process" (an iterative waterfall) for DSLs in the way as shown in Fig. 1, with the specific changes being a modified requirements step (mainly also including reasoning), addition of "3. Ontological Analysis", having made steps 5 and 6 optional, and including CNLs as optional syntactic sugar cf. only diagrammatic notations. Acknowledging that in practice the final version of the language typically is not obtained in a single sequential pass through these steps, we added the possibility of backtracking between each step and the previous one, and between Step 7 and Step 4. The highlighted Step 3 is the focus of this paper, which we shall turn to after illustrating briefly how each step applies to ontology language design.

For step 1, the *scope* is ontology languages (rather than, say, DSLs, Business Process Modeling languages etc.), whereas the *goals* and *purpose* may already vary for the case at hand. An example of a scope specification can be found in the DOL standard [9]. A typical goal for ontologies may be *to have the necessary constructs to be able to represent that what needs to be formalised as precise as possible* in order to match the intended models as closely as possible with the models that the theory admits, to assist with developing 'good' ontologies [13]. In most ontology research (cf. their applications in industry), economics does not play a role, whereas a long-term perspective is assumed.

For step 2, Frank refers to a "requirements catalogue" to consult and choose one's requirements from [10], which does not exist for ontology languages. Perhaps a 'modelling features on offer' list, alike in the appendix of the Description Logics handbook [14], could be seen as contributing toward a catalogue, but a features-on-offer list is distinct from a broad requirements list. Broader requirements for a logic may encompass not just the modelling features (e.g., qualified cardinality, transitivity, etc.), but also concern the language as a whole, such as *must be usable in a multilingual setting*, and its context, such as *must be able to work with ontology modules*; examples of sets of requirements are those for OWL [8] and CL [15, section 5]. The requirements for automated reasoning are, typically, inconsistency checking, subsumption reasoning, instance checking, and querying the ontology and data in ontology-based data access. It may be that one reasoning service is deemed more important than the others, hence, priorities can be, or have been, assigned (item 2c) implicitly or explicitly. This is the case with the OWL family: for the OWL profiles [16], querying instances was deemed much more important for the OWL 2 QL developers, and size trumped inconsistency checking for the OWL 2 EL developers. A few trade-off tables are also available, such as for logics for conceptual models [17] and for representing various fragments of the KGEMT mereotopology in multiple logics [18]. There are few explicit use cases designed to inform ontology language development beyond the toy examples for a particular features (e.g., 'need to be able to model narcissist' to motivate reflexive object properties [19]), with the exception of the 12 use cases for DOL [9].

Step 3 may seem closely related to both requirements and language design, so why then a separate step? It turns out that the jump from requirements to language design leaves implicit many decisions of an ontological nature, where meeting a requirement entails committing to one philosophical stance or another. This may have to be deliberated first, or else at least be a conscious decision to document. For instance, if there is a requirement *track the entities through time*, then does that mean 3D objects+time or 4D objects, and if the former, does that have to come with discrete or dense time, and with linear time or trees, or if the requirement states *with attributes*, then does that mean attributions with a universalist stance or one with tropes, and as separate element in the language to support the representation of either, or not? More generally, this raises the question *what is the ontological analysis to conduct, to the extent that it will have a bearing on the logic?* We shall delve into answering this question in the next section.

The language specification step (no. 4) is comparatively well-known, with Description Logics probably the most popular by number of papers, and OWL by number of ontologies represented in that language. Most of these logics do not have a glossary, documentation (beyond the scientific paper), or a metamodel. Regarding the latter, this is likely thanks to the formal semantics (which is different for DSLs), although the UML diagrams in the OWL standard [20] amount to a metamodel. The DOL and CL standards do include a "terms and definitions" section, functioning as a glossary [9,15].

Steps 5 and 6 in the pipeline are optional, since an ontology language may well be a fully paper-based logic without any syntactic sugar or tooling support. Some logics for ontologies do have such interfaces and tools. For instance, there are multiple controlled natural language (CNL) interfaces to several OWL fragments, as well as graphical renderers, and a relatively comprehensive tooling infrastructure for manipulating and reasoning over OWL files. There are a few tools for Common Logic [15] and OBO [21].

The last step in the process is to evaluate the artefact, being the logic for representing the ontology: does it meet the requirements? Does it solve the problems described in the use cases? Does it adhere to the priorities? The answers probably will not be a threefold "yes", but with a systematic procedure in place, it should at least have become clear where concessions have been made, how, and why, which at least contributes to an understanding of why that logic (ontology language) is the way it is.

3. Design choices for ontology languages

We will now zoom in on Step 3 of the design process (recall Fig. 1).

Assess ontological commitments for the language (Step 3a of the process) While the seminal paper on ontology-driven information systems [13] did note *ontological commitment* as "intensional interpretation" in addition to the standard extensional one in a model-theoretic semantics, it did not consider the affordances and features of the logic. The latter entails two components: 1) the ability to represent the conceptualisation more or less precisely with more or less constraints[2], and 2) whether the representation language contributes to support, or even shape, the conceptualisation and one's ontological analysis for the ontology, or embeds certain philosophical assumptions and positions. Regarding the latter, we identified four key decision points that each have multiple sub-questions and decisions each, which may not yet be exhaustive, but it is the first and most comprehensive collection to date for this approach. They are drawn from related work and our own, and are elaborated on afterward.

1. What belongs to the 'fundamental furniture of the universe', or: what are the 'epistemological primitives'? From a logic viewpoint: what are the building blocks in the logic? This includes answering questions such as:
   (a) Does the world have an abundance of 3-dimensional objects (with an optional time dimension), or are there 4-dimensional space-time worms and thus a language catering for that? Related to that: is the world made up of processes, actions, etc. that static entities participate in or are there static entities that may, or may not, participate in this dynamism?
   (b) Do two or more elements relate directly, or through the roles they play in the relation?
   (c) Related to, or perhaps underlying, the former two items: is there a prioritisation among the primitives, some being more or less relevant, and does that affect the notion of 'fundamental'?
2. Should one refine the kinds of general elements (from item 1) and promote them to have their own representation element in the logic, to possibly result in a different (better) ontology? For instance,
   (a) Refining Relationship with an pre-defined element for parthood;
   (b) Refining Class (or concept or universal) with, e.g., stereotypes or a many-sorted logic to indicate ontological distinctions between the kind of entities, such as between a rigid and non-rigid entities, or between sortals, quasi-types, and attributions etc;
   (c) Setting the arity of the relationship (or n-ary predicate with $n \geq 2$): if only binary relationships are allowed, then the modeller may assume there are only binary relations in the world, reifications of n-aries vs the existence of n-aries ($n \geq 2$) proper, and fixed arities vs. relationships with variable arity;
3. Should the logic be intertwined with natural language, or is natural language a layer 'on top of' the logic and thus separable from (and perhaps even independent of) the core knowledge or ontology? Related to this: What must be named?

[2]this is different from subject domain coverage; to be able to represent, say, "has part =2 legs" vs only "has part $\geq$ 1 legs" concerns *precision*, whereas omitting information about the legs concerns *coverage*.

4. Is the world crisp in the sense of something either being true or false, or may an entity be something to a degree and the world is thus inherently vague?

There is scant research on the effects of choosing one option over the other, with one notable observation in the related area of conceptual modelling: binaries vs. *n*-aries (Item 2c) and just plain relationship vs. also with aggregation (roughly parthood, Item 2a) do indeed make a difference at least for UML vs ER and ORM: UML class diagrams have significantly more aggregation associations than ER and ORM diagrams have, yet fewer *n*-aries [6]. The former is attributed to it being a separate element and the latter at least partially because there are obstructions to draw *n*-aries and to understand them due to its graphical notation [5], compared to ER and ORM that use the same notation for both binaries and *n*-aries and have no primitive for parthood. Parthood also features prominently in ontologies represented in the OBO format, where it was a primitive [22], whereas noting that absence of such a primitive in OWL might be an explanation for the well-known is-a/part-of confusion by novice ontologists. There is a 25-year old proposal to include parthood as a primitive in DLs [23], but this has never been pursued further. The lack of *n*-ary relationships—i.e., where $n \geq 2$ rather than just $n = 2$—in OWL has been a long-standing complaint, since it is possible to have them in DLs, notably the $\mathcal{DLR}$[24] and $\mathcal{CFD}$ [25] families, whilst modellers are facing workarounds with ontology design patterns to approximate it by means of a reification with partial constraints. The *n*-aries problems do not exist in full FOL or HOL, but they do not have a primitive for parthood, since there there are just *n*-ary predicates ($n \geq 1$), not predicates + parthood, in the definition of the language. Surely, one can define a 'FOL with parthood', but that is different from the regular definition of full first-order predicate logic.

We will elaborate on the remainder of the items in the following paragraphs.

On the fundamental furniture of the universe (Item 1) Practically: what are the building blocks in the logic? More principled with respect to ontology, this includes assessing:
– Are there just predicates with $n \geq 1$, or are there entity types that are (necessarily) related by means of *n*-ary relationships (with $n \geq 2$), where that distinction between unaries and *n*-aries is ontologically meaningful?
– Are the roles that objects play in relation(ship) fundamental components, and therewith that a relationship does not have a directionality, no inverses, and is so-called *positionalist* (cf *standard view*)? (rewording of Item 1b)
– Within the predominant 3D scope: are stuffs distinct from objects, are they types of objects, or do they not exist?

Let us discuss Item 1b first, since different decisions have been taken. As illustration, assume there is some binary relationship called teach that holds between Professor and Course. In the "standard view" [26], there would be at least one predicate, teaches (or taught by), in which Professor and Course participate *in that specific order* (or in the reverse, respectively). The 'there are roles too'-option ("positionalism" [26]) would argue that Professor plays a role, say, [lecturer], in the relationship teach and Course plays the role [subject] in the relationship teach. The relationship has no 'direction', as the standard view has, and the roles thus do not have an ordering, since the participation is clear from the assignment of the object to a role in the relationship. Objects always play a role in the relations they participate in, and inherently so; hence, *role* will have to be an element in the language. Besides the philosophical arguments, positionalism is also deemed better for natural language interaction and expressing more types of constraints than with standard view relationships [26,27,28,29], therewith contributing to more precise representations of the universe of discourse. Most DLs, OWL, FOL, and HOL adhere to a standard view commitment, whereas the main conceptual data modelling languages (EER, UML, and ORM) and the $\mathcal{DLR}$ family of DLs is positionalist (having so-called DL role components). There is also an anti-positionalist stance, which is argued to be even better than positionalist [26,29] and a language specific for the anti-positionalist commitment is proposed in [30].

Stuffs are generally assumed to be distinct from countable objects (third item, above), yet there is no uniform approach to deal with it because it is not agreed upon how distinct they are. Are they different categories of things, or are they both universals or particulars, or is a stuff (or its particular amount of matter) just another type of endurant, or is each amount of matter an object? The answer may affect the ontology language. For instance, Donnelly and Bittner use a many-sorted logic for the portions to distinguish it from objects with parts [31], stereotyping in UML [32] or formalising it in a HOL [33], or merely a subclass of endurant [34] and therewith thus would not merit a separate element in the logic. Currently, most logics do not make that distinction.

3D vs. 4D (Item 1a) 3-dimensionalism assumes there are objects in space where the objects are wholly present at each point in time (i.e., do not have temporal parts) and statements are true at the 'present', whilst being ignorant of the object in the past and future. Time can be added as an orthogonal dimension, as in, e.g., $\mathcal{DLR}_{\mathcal{US}}$ [35], for which there are further choices, notably linear vs trees and dense vs discrete time. 4-dimensionalism, sometimes also called 'fluents', assumes entities exist in four dimensions, being in spacetime, entities unfold in time, and thus do have temporal parts, and statements can be about not only the present, but also the past and future; examples for ontologies and ontology-driven modelling include [36,37]. What does this mean for representing knowledge? Let us take as example a holding or supra-organisation [37], such as Alphabet and Nestlé: these companies exist for some time and keep their identity all the while they acquire and sell other (subsidiary) companies. In a 3D-only representation where there is only an atemporal 'current' snapshot, one would have a record of which companies they own now, but not whether they are the same ones as last year.

The predominant choice made by developers of popular ontology languages is 3D objects with optional temporal extensions. We could not find any scientific evidence that demonstrates explain why this is the case. From a computational viewpoint, the temporal extension is expensive, hence, is prohibitive for designing scalable systems.

Refining the core elements or not (Item 2) Principled ideas for refining the unaries (concepts, classes, universals) avail of notions such as rigidity that then leads to types of unaries [38], such as "type" for the kind of objects that supply an identity criterion (e.g., Person) and "role" that an object plays (e.g., Professor), as examples for Item 2b. Like aforementioned stuff, they could be defined in a HOL, or used in declaring a many-sorted logic, be it based on such metaproperties or, say, by taking the main entities from a foundational ontology.

Another possible refinement of core elements may be the aforementioned parthood relation (Item 2a), and one could do likewise with other common relations, such as participation, causation, and constitution. A refinement of binary relationship is *attribute* (OWL data property), which holds between a class and a data type. It is debatable whether attributes with their data types belong in an ontology language. For instance, KL-ONE is clear in stating that they are *not* among the "epistemological primitives" for they have "no semantically justifiable place in the epistemology" and therefore do not belong in the logic [7]. Common Logic does not have attributes either, but one can specify its inclusion through an extension [15]. Similarly, it is absent from most DLs, yet it made it into the OWL specification [20]. There are indeed no data types, hence, no attributes, in a world without humans having conceived of the construct. Whether an ontology and its language are permitted to contain 'unnatural' things is debatable. One also may argue that an attribution or quality such as length means the same thing regardless whether it is used for the length of, say, a Sofa or a Table and thus would be one thing, so rather than to split up into two or more types of attributes, alike $\mathsf{lengthS} \mapsto \mathsf{Sofa} \times \mathtt{Integer}$, $\mathsf{lengthT} \mapsto \mathsf{Table} \times \mathtt{Float}$, and $\mathsf{lengthT'} \mapsto \mathsf{Table} \times \mathtt{Int}$, it should be one datatype-independent property; hence, an argument from ontological parsimony vs abundance is also possible.

Another refinement is implication—assuming implication is a core notion. In FOL and HOL, there is only implication, but it has been argued that not all implications are the same. In particular, there is the notion of *inheritance* for classes/concepts/universals (unaries) and *subsetting* and *subtyping* for relationships (n-aries, with $n \geq 2$). For instance, while the DL axiom $\mathsf{Cat} \sqsubseteq \mathsf{Animal}$ gets translated into FOL as $\forall \mathsf{x}(\mathsf{Cat}(\mathsf{x}) \rightarrow \mathsf{Animal}(\mathsf{x}))$, the former has the embedded notion of *property inheritance along the taxonomy* where the properties of Animal are inherited by Cat. Logics with a syntactic sugar communicates this differently to modellers and domain experts yet again. For, instance, the aforementioned is verbalised as 'Each Cat is an Animal', whereas, say, $\forall \mathsf{x}(\mathsf{Cat}(\mathsf{x}) \rightarrow \exists \mathsf{y}(\mathsf{eats}(\mathsf{x},\mathsf{y}) \land \mathsf{Mouse}(\mathsf{y})))$ is not verbalised as 'Each Cat is a something that eats at least one Mouse' but as 'Each Cat eats at least one Mouse', in that there's a relationship eats between two entities, on par with each other, not the notion of subsumption or inheritance between an entity and a relation it participates in.

Naming things (Item 3) Naming something involves identifying it and acknowledging its relevance; conversely, the nameless may be redundant, irrelevant, or non-existent. The process of naming something involves the interaction between natural language and ontology. There are millennia-old philosophical debates on that interaction. Naming elements may come before or after determining the 'fundamental furniture', and differ by ontology language, as illustrated in Table 1; observe that none of the languages names all six types of elements. Moreover, those elements are given names, such as Prof and $\mathsf{teaches}$, embedding natural language into the logic. There are alternatives to this approach. OBO uses identifiers for concept names, which each have one or more labels for the natural language name of that natural language-independent entity. This approach can be ported into the OWL world where, e.g., the Protégé tool then renders the labels in the interface. An attempt to systematically address that natural language $\leftrightarrow$ ontology interaction has been proposed in the context of the Semantic Web, where the natural language dimension has its own extension on top of an OWL file [39], by means of a W3C community-based annotation scheme [40]. For the related conceptual modelling languages, it is worth noting that ORM diagrams commonly have *reading labels* for a fact type (relationship), where modellers hardly add names for the roles or the fact type themselves [6] (modelling tools add those automatically in the serialisation), whereas for UML Class diagrams, association end/memberEnd (i.e., role) names are expected, but not association (relationship) names, as also illustrated in the UML standard.

Table 1. Terminology of some of the logics for ontologies (relevant selection).

General term	**KL-ONE**	**OWL**	**FOL**	**CL**
Relationship	RoleSet	Object Property	Predicate	Name
Role	Link	n/a	n/a	n/a
Entity type	Concept	Class	Predicate	Name
Attribute	n/a	Data Property	n/a	n/a
Datatype	n/a	Datatype	n/a	n/a
Function	n/a	n/a	Function	Functional term (that is a Name)

A crisp world or not (Item 4) While the debate on concepts vs universals have received ample attention, they both assume a crisp world: they have clear boundaries, specified by a set of properties of the universal or concept, and where some instance is either an instance of that universal or concept, or not. There are alternatives. In cognition and learning, the notions of *prototypes* and *exemplars* are well-known [41]. Prototypes have fuzzy boundaries since some object fits a category more or less well, and some exemplars may be better or not that good: they are more or less of a member of that set. For instance, Penguin is not a prototypical bird, or not a good exemplar of it, compared to, say, Raven. Also, entity types may have a rough boundary: it is clear what is in and what is out, but membership cannot be established around the border, be it not ever or not with the knowledge or data at one's disposal; i.e., the notion of a *rough set* [42]. Further, recent years has seen widespread use of statistics, such as in machine learning and deep learning, where non-crisp models are built. It remains to be seen whether adoption of these techniques entail an underlying difference in philosophical stance or a 'disagree and commit' because of the remarkable outcomes, and how, if at all, such outcomes would be connected to ontologies. Depending on one's philosophical inclination, the logic will either have only true/false, or some way to deal with the vagueness that may be inherent [43,44]. Logics with vagueness have been investigated and some tools and applications exist, including for DLs [45].

Engineering factors (Step 3b of the design process) How do the chosen primitives of Items 1 and 2 relate to the logical symbols to obtain the intended meaning? It is also necessary to analyse the role of the logic. Logic maps explicit extensional primitive elements (facts or relations, for example) into implicit intensional elements through the available logical symbols and inference rules. This mapping raises the point of determining which logical symbols and inference rules are adequate, which have to be together with the well known trade-off between *expressivity* and *efficiency* [46]. For instance, instead of mere n-ary predicates with $n \geq 1$, as in FOL, one could design a logic with a syntax where unaries denote universals, indicated with, e.g., camel case notations in serif (e.g., a universal Meerkat) and n-aries with $n \geq 2$ as relational properties denoted in the syntax with italics serif in all lowercase (e.g., *eating*), and roles in square brackets (e.g., [prey]), to enable declaring that there are eating events where meerkats are the prey of jackals, in a positionalist universe and underlying sets for semantics: $\exists \mathit{eating} \Rightarrow [\mathsf{prey}]\mathsf{Meerkat} \times [\mathsf{predator}]\mathsf{Jackal}$. Another example is the temporal operators in a temporal logic, such as 'until' U and 'at all times' $\Box$: while it indeed can be reconstructed in FOL, it is deemed essential so as to merit to be explicitly available in the language. One also could add alternative symbols to distinguish that generic 'mere' implication with subsumption between unaries, as alluded to with the Cat and Mouse, example, above; e.g., using different notation alike, say, a Cat $\Rightarrow$ Animal vs. a Cat $\rightarrow \exists$eats.Mouse. Such considerations forces a designer to indeed have taken a stance on Item 1c of the ontological commitments and deal with its consequences in this step b. Non-logical symbols, such as syntactic sugar, may afterwards be added but are not in the focus of the logical language; e.g., typical natural language renderings of "$\exists$" are 'there are', 'at least one', and 'some'.

Besides ontological commitments, there may be practical considerations in designing the representation language. These are mainly concerned with automated reasoning: 1) which reasoning services are needed, 2) how scalable it has to be, if at all, and 3) whether and to what extent can one avail of existing tooling infrastructure. Examining this step is not the focus of this paper.

4. Evaluation

We apply and evaluate the language design and ontological analysis step in two ways. First, we assess several key ontology languages that were, and are, used throughout the decades. Second, we step through the design process to see how it may work for a hypothetical 'FK Ontology Language', *FKOL*, that should meet our *ad hoc* requirements.

4.1. Assessment of popular ontology languages

The logics for the assessment were selected for the following reasons. OBO [21] has been widely popular with bio-ontologies since it use since 1998 for the Gene Ontology [22] and it helped popularising the use of lightweight domain ontologies especially in science. SKOS is also popular, notably for bringing thesauri and similar vocabularies into the Semantic Web [47] and for its subsequent use in bottom-up ontology development. KL-ONE is essentially the predecessor to DL and, unlike the DL documentation, did try to express its rationale for the design of the language [7]. $\mathcal{DLR}_{ifd}$ [48] is an unusual DL in that it has features that are commonly claimed as not doable and it is deemed the best fit for a logic-based reconstruction of conceptual data modelling languages such as UML Class Diagrams. OWL 2 DL [20] is included in the comparison because it has popularised ontologies in government and industry and was standardised by the W3C. Traditional FOL is included because it is a common logic foundation theoretically at least. HOL was added because it is the most expressive family of logics.

Their comparisons on ontological commitments and computational features are included in Tables 2 and 3, respectively. In Table 2, a feature is considered primitive ('yes') when there is a corresponding logical symbol in the language; it is 'possible' if there is a direct and simple reconstruction of the meaning of the feature in the language without altering the essential properties of the language (in terms of those defined in step 3). For example, subsumption is possible to be defined in terms of FOL primitives; in contrast, fuzzy extensions of OWL are possible, but they change significantly the designed properties of the language, so we deem this a 'no'. If the feature is not completely represented in the language, it is 'partial'; e.g., $\sqsubseteq$ in DLs does have property inheritance in the semantics for class subsumption, but there are no different symbols for properties of classes as illustrated with the Cat, Animal, and Mouse above.

FOL and HOL turned out to have the same answers in Table 2 and therefore were merged into one column; SKOS is not used for automated reasoning and therefore omit-

Table 2. Selection of logics for ontologies and vocabularies and their ontological commitments and features. 3D/4D: 3/4-dimensionalism; NL: natural language

Feature	**OBO**	**SKOS**	**KL-ONE**	$\mathcal{DLR}_{ifd}$	**OWL2DL**	**FOL/HOL**
Standard view	yes	yes	yes, with some positionalist	position-alist	yes	yes
3- or 4-dimensionalism	3D	3D	3D	3D	3D	3D or 4D
NL$\leftrightarrow$logic separation	yes	possible	no	no	possible	no
Type(s) of unaries	class	concept	generic, primitive, defined, and individual concepts	concept	class	unary predicate
Concept subsumption as primitive	yes	no[‡]	yes	partial	partial	possible
Parthood as primitive	yes[*]	possible	no	no	no	no
Relationship subsetting and sub-typing as primitives	no	subsetting possible	yes	subsetting	subsetting	no
Relationship definition	no	no	composition	no	limited chains	yes
n-aries (where $n \geq 3$)	no	no	no	yes	no	yes
Attributes with datatypes	no	partial	no	yes	yes	no
Functions or procedures	no	no	procedures may be attached	no	no	functions
Multi-valued (including fuzzy, rough)	no	no	no	no	no	possible
Semantics	graph	none / variable	variable	model-theoretic	model-theoretic	model-theoretic
Open World Assumption	no[†]	no[†]	yes	yes	yes	yes
Unique Name Assumption	yes[†]	yes	yes	no	no	no

[*] depends on the OBO file format version: v1.4 does not have it so as to be compatible with OWL.
[†] not explicitly stated in the documentation to be the case, but likely given other information.
[‡] there is `skos:broader` and `skos:narrower`, but they do not equate sub-/super-concept.

ted from Table 3. As can be seen from the values in the columns, there are varied ontological and computational commitments, except for a clear predominance of a 3-dimensionalist stance and crisp logics, and to some extent also Open World Assumption, parthood not as primitive, and on not scoring well on scalability; or: lacking 4-dimensionalism, uncertainty, CWA, parthood, and scalability. Why this is so is a question that an experimental philosopher may wish to investigate.

4.2. Designing one's preferred language

Commencing the design process with *Step 1*, the scope is 'use case' of which the benefits are the evaluation of the applicability of the design approach to ontology language design. Then, to determine the requirements of our fictitious *FKOL* language (*Step 2*), we need to avail of a requirements catalogue. Since it did not exist, we first created a draft catalogue by combining the published requirements from OWL [8] and CL [15], added the ontology-motivated ones, and a few more that seemed possibly relevant for

Table 3. Selection of logics for ontologies and their computational features.

Feature	OBO	KL-ONE	$\mathcal{DLR}_{ifd}$	OWL2 DL	FOL	HOL
Complexity (sat., subsumtions)	depends*	undecidable	ExpTime-complete	N2ExpTime-complete	undecidable	highly undecidable
Scalability	yes	not scalable	± scalable TBox	± scalable TBox	not scalable	not scalable
Serialisation	yes	no current one	no	RDF/XML	yes, e.g., CLIF	yes, e.g., Isabelle
Reasoner	no	no current one	no	multiple	few	very few
Interoperability	yes	no	no	yes, ample	little	very little
Usable with DOL	no	n/a	no	yes	yes	yes
Modular ontologies	no	no	no	yes	possible with DOL	possible with DOL

* depends on the version and format (whether as graph or as stored in a database).

such a catalogue. This preliminary catalogue of 56 possible requirements is available online and open for comments and extensions[3]. After establishing the catalogue, the authors independently chose a subset of features they like most or would be most interesting to experiment with, which were then combined into a joint list. The joint list did not have conflicting requirements and is included in Fig. 2. Use case scenarios include specification of the subject domain knowledge that it has to be able to represent, such as that all canonical humans have exactly two legs as part (Case1), and crisp subsumption reasoning and classification, in that it should be able to compute that, e.g., elephants are herbivores (Case2). Due to space limitations, we assign equal priority to each selected requirement.

With respect to the ontological analysis (*Step 3*), the following. The reason for positionalism (O-5 in the requirements catalogue) is a combination of several arguments: 1) the assumption that when things relate, there is one relation in reality, not as many as humans have words for it, 2) it allows for more detailed constraints, so possibly resulting in a better ontology, and 3) it has a better link with popular conceptual data modelling languages. The reasons for parthood as primitive (O-6) are because of its pervasiveness throughout ontologies and when it is a separate primitive, like the aggregations association in UML class, it increases analysis and use and thus may improve ontology quality. Subsumption (O-6) goes hand-in-hand with the requirement for subsumption reasoning/taxonomic classification (UC-8) and that millennia-old notion. The choice for a crisp world (O-7) is based on the argument that any vagueness can be attributed to language or lack of knowledge, rather an inherent vagueness. One can, of course, easily argue differently, but these are our—i.e., the language designers'—choices for the purpose of the evaluation.

The performance trade-offs (*Step 3b*) for *FKOL* will not be good. If we take parthood to mean including the axioms of Ground Mereology, reasoning over a partonomy would be undecidable due to antisymmetry of parthood, so let us ignore antisymmetry. Then the complexity trade-offs depend largely on the list of constraints in E-2, which are fairly straight-forward and can be obtained with languages that are, at most, ExpTime-complete and probably less, since, say, $\mathcal{ALCQI}$ is ExpTime-complete. The other require-

[3]`https://keet.wordpress.com/2020/04/10/a-draft-requirements-catalogue-for-ontology-languages/`

Req-E	Expressiveness/constructs/modelling features:	
	E-1	Equipped with basic language elements: *n*-ary predicates ($n \geq 1$, as classes and relationships), roles, and individuals.
	E-2	Equipped with language features: domain & range axioms and cardinality constraints.
Req-F	Features of the language as a whole:	
	F-9	Should not make arbitrary assumptions about semantics.
	F-11	Extendable (e.g., regarding adding more axioms to same ontology, add more vocabulary, and/or in the sense of importing other ontologies).
	F-16	Use Unique Name Assumption.
Req-UC	Usability by computer:	
	UC-1	Be an (identifiable) object on the Web.
	UC-8	Able to be used by tools that can do subsumption reasoning/taxonomic classification.
	UC-9	Able to be used by tools that can detect inconsistency.
Req-HU	Human usability:	
	HU-5	Such that a modeller is free to invent new names and use them in published content.
	HU-6	Have clearly defined syntactic sugar (CNL or diagrammatic)
Req-I	Interaction with outside:	
	I-3	Compatible with existing standards (e.g., RDF, OWL, XML, URIs, Unicode).
Req-O	Ontological decisions:	
	O-1	3-Dimensionalist commitment.
	O-5	Positionalist relations and relationships.
	O-6	Have additional primitives: parthood, subsumption.
	O-7	Statements are either true or false.

Figure 2. Combined requirements selected from the requirements catalogue by the authors, for which a logic would have to be designed.

ments can be met without making the language less well-behaved computationally, but it is actually incomplete, still. Notably, it does not say which semantics should be chosen in F-9 and which existing standards in I-3, but let us assume a model-theoretic semantics and Unicode for now.

Since, to the best of our knowledge, there is no ontology language that meets all these requirements, one would have to proceed to *Step 4*: specifying the syntax and semantics, a glossary, and, optionally, a metamodel. We will not pursue this here, since that easily can take up the space of a paper and the focus is on the process of language design and the ontological analysis for certain choices.

The process finalises with the evaluation of the language defined against the set of requirements (Fig. 2) and devising test cases. The test cases should match with the use case scenarios specified in *Step 2b*. Case1 could be tested by examining whether one can represent exactly that: with parthood as $\preceq$, then, say, $\mathsf{Human} \preceq_{=2} \mathsf{Leg}$ in some made-up DL-like syntax, which the language would allow if it meets requirements E-1 (classes), E-2 (for domain and range declaration), and O-6 (parthood). Case2 tests are more elaborate to realise, testing whether requirements E-1 (classes and relationships), E-2 (to declare properties needed for classification), UC-8 (classification) and O-6 (subsumption) have been implemented.

What this use case evaluation has shown is that it is feasible to step through the process of language design, including ontological analysis. It also illustrates that more research and method development could assist this process further, such as possible dependencies between requirements, and how to systematically match use cases to requirements and to test cases.

5. Conclusions

We have outlined an engineering approach to the design of an ontology language, with a particular emphasis on the ontological commitments that hitherto were typically brushed over in the language design stage. It has put more and less common topics and debates in ontology in a different context for a new way of using them: before the actual investigation and representation of a universe of discourse. These results presented may contribute to reflecting on the language as enabler or inhibitor to formally characterising an ontology or an ontological investigation, and spur research into the effects of the representation language on what is eventually represented in the ontology.

References

[1] Kendall EF, McGuinness DL. Ontology Engineering. Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool Publishers; 2019.

[2] Gal A. Ontology Engineering. In: Liu L, Özsu MT, editors. Encyclopedia of Database Systems, Second Edition. Springer; 2018. .

[3] Keet CM. An Introduction to Ontolgy Engineering. vol. 20. College Publications; 2018.

[4] Tudorache T. Ontology engineering: Current state, challenges, and future directions. Semantic Web. 2020;11(1):125–138.

[5] Shoval P, Shiran S. Entity-relationship and object-oriented data modeling—an experimental comparison of design quality. Data and Knowledge Engineering. 1997;21:297–315.

[6] Keet CM, Fillottrani PR. An analysis and characterisation of publicly available conceptual models. In: Proc. of ER'15. vol. 9381 of LNCS. Springer; 2015. p. 585–593. 19-22 Oct, Stockholm, Sweden.

[7] Brachman R, Schmolze J. An overview of the KL-ONE Knowledge Representation System. Cognitive Science. 1985;9:171–216.

[8] Horrocks I, Patel-Schneider PF, van Harmelen F. From SHIQ and RDF to OWL: The making of a web ontology language. Journal of Web Semantics. 2003;1(1):7.

[9] Distributed Ontology, Model, and Specification Language; 2018. Available from: `http://www.omg.org/spec/DOL/`.

[10] Frank U. Domain-Specific Modeling Languages - Requirements Analysis and Design Guidelines. In: Reinhartz-Berger I, et al., editors. Domain Engineering: Product Lines, Conceptual Models, and Languages. Springer; 2013. p. 133–157.

[11] de Kinderen S, Ma Q. Requirements engineering for the design of conceptual modeling languages. Applied Ontology. 2015;10(1):7–24.

[12] Karsai G, Krahn H, Pinkernell C, Rumpe B, Schindler M, Völkel S. Design Guidelines for Domain Specific Languages. In: Proceedings of the 9th OOPSLA Workshop on Domain-Specific Modeling (DSM'09); 2009. Orlando, Florida, USA, October 2009.

[13] Guarino N. Formal Ontology and Information Systems. In: Guarino N, editor. Proceedings of Formal Ontology in Information Systems (FOIS'98). Amsterdam: IOS Press; 1998. p. 3–15.

[14] Baader F, Calvanese D, McGuinness DL, Nardi D, Patel-Schneider PF, editors. The Description Logics Handbook – Theory and Applications. 2nd ed. Cambridge University Press; 2008.

[15] Common Logic (CL): a framework for a family of logic-based languages [Standard]; 2007. Available from: `https://www.iso.org/standard/39175.html`.

[16] Motik B, Grau BC, Horrocks I, Wu Z, Fokoue A, Lutz C. OWL 2 Web Ontology Language Profiles. W3C; 2009. Available from: `http://www.w3.org/TR/owl2-profiles/`.

[17] Fillottrani PR, Keet CM. Evidence-based Languages for Conceptual Data Modelling Profiles. In: Proc. of ADBIS'15. vol. 9282 of LNCS. Springer; 2015. p. 215–229. 8-11 Sept, 2015, Poitiers, France.

[18] Keet CM, Kutz O. Orchestrating a Network of Mereo(Topo)Logical Theories. In: Proceedings of the Knowledge Capture Conference. K-CAP 2017. New York, NY, USA: ACM; 2017. p. 11:1–11:8.

[19] Horrocks I, Kutz O, Sattler U. The Even More Irresistible $\mathcal{SROIQ}$. Proc of KR'06. 2006;p. 452–457.

[20] Motik B, Patel-Schneider PF, Parsia B. OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax. W3C; 2009. Http://www.w3.org/TR/owl2-syntax/.

[21] Mungall C, Ruttenberg A, Horrocks I, Osumi-Sutherland D. OBO Flat File Format 1.4 Syntax and Semantics; 2012. Available from: `http://purl.obolibrary.org/obo/oboformat/spec.html`.

[22] Gene Ontology Consortium. Gene Ontology: tool for the unification of biology. Nature Genetics. 2000;25:25–29.

[23] Artale A, Franconi E, Guarino N. Open Problems for Part-Whole Relations. In: Proc. DL'96. Cambridge, MA: AAAI Press; 1996. p. 70–73.

[24] Calvanese D, De Giacomo G, Lenzerini M. On the decidability of query containment under constraints. In: Proc. of PODS'98; 1998. p. 149–158.

[25] Toman D, Weddell GE. On Adding Inverse Features to the Description Logic $\mathit{CFD}^{\forall}{}_{\mathrm{nc}}$. In: 13th Pacific Rim International Conference on Artificial Intelligence (PRICAI'14); 2014. p. 587–599.

[26] Fine K. Neutral Relations. The Philosophical Review. 2000;109(1):1–33.

[27] Keet CM. Positionalism of relations and its consequences for fact-oriented modelling. In: Proc. of ORM'09. vol. 5872 of LNCS. Springer; 2009. p. 735–744. Vilamoura, Portugal, Nov 4-6, 2009.

[28] Keet CM, Chirema T. A model for verbalising relations with roles in multiple languages. In: Blomqvist E, et al., editors. Proc. of EKAW'16. vol. 10024 of LNAI. Springer; 2016. p. 384–399. 19-23 Nov 2016, Bologna, Italy.

[29] Leo J. Modeling relations. Journal of Philosophical Logic. 2008;37:353–385.

[30] Leo J. Coordinate-free logic. The Review of Symbolic Logic. 2016;9(3):522–555.

[31] Donnelly M, Bittner T. Summation relations and portions of stuff. Philosophical Studies. 2009;143:167–185.

[32] Guizzardi G. On the representation of quantities and their parts in conceptual modeling. In: Proc. of FOIS'10. IOS Press; 2010. 11th-14th May 2010, Toronto, Canada.

[33] Keet CM. Relating some stuff to other stuff. In: Blomqvist E, et al., editors. Proc. of EKAW'16. vol. 10024 of LNAI. Springer; 2016. p. 368–383. 19-23 Nov 2016, Bologna, Italy.

[34] Masolo C, Borgo S, Gangemi A, Guarino N, Oltramari A. Ontology Library; 2003. Http://wonderweb.semanticweb.org. WonderWeb Deliverable D18 (ver. 1.0, 31-12-2003).

[35] Artale A, Franconi E, Wolter F, Zakharyaschev M. A temporal description logic for reasoning about conceptual schemas and queries. In: Proc. of JELIA'02. vol. 2424 of LNAI. Springer; 2002. p. 98–110.

[36] Batsakis S, Petrakis E, Tachmazidis I, Antoniou G. Temporal Representation and Reasoning in OWL 2. Semantic Web Journal. 2017;8(6):981–1000.

[37] West M, Partridge C, Lycett M. Enterprise Data Modelling: Developing an Ontology-Based Framework for the Shell Downstream Business. In: Proc. of FOMI'10; 2010. p. 71–84. 14-15 Dec., 2010, Trento, Italy.

[38] Guarino N, Welty C. An Overview of OntoClean. In: Staab S, Studer R, editors. Handbook on Ontologies. Springer Verlag; 2009. p. 201–220.

[39] Buitelaar P, Cimiano P, editors. Towards the Multilingual Semantic Web: Principles, Methods and Applications. Springer; 2014.

[40] Cimiano P, McCrae JP, Buitelaar P. Lexicon Model for Ontologies: Community Report. W3C; 2016. Available from: `https://www.w3.org/2016/05/ontolex/`.

[41] Rosch E. Principles of Categorization. In: Margolis E, Lawrence S, editors. Concepts: Core readings. MIT Press; 1978/1999. p. 186–206.

[42] Pawlak Z, Skowron A. Rudiments of rough sets. Information Sciences. 2007;177(1):3–27.

[43] Prinz J. Vagueness, Language, and Ontology. Electronic Journal of Analytical Philosophy. 1998 Spring;6. Available from: `http://ejap.louisiana.edu/EJAP/1998/prinz98.html`.

[44] Sorensen R. Vagueness. In: Zalta EN, editor. The Stanford Encyclopedia of Philosophy; 2003. Available from: `http://plato.stanford.edu/archives/fall2003/entries/vagueness/`.

[45] Lukasiewicz T, Straccia U. Managing Uncertainty and Vagueness in Description Logics for the Semantic Web. Journal of Web Semantics. 2008;6(4):291–308.

[46] Guarino N, Oberle D, Staab S. What Is An Ontology? In: Staab S, Studer R, editors. Handbook on Ontologies. Springer; 2009. p. 1–17.

[47] Miles A, Bechhofer S. SKOS Simple Knowledge Organization System Reference. World Wide Web Consortium (W3C); 2009. Available from: `http://www.w3.org/TR/skos-reference/`.

[48] Calvanese D, De Giacomo G, Lenzerini M. Identification Constraints and Functional Dependencies in Description Logics. In: Proc. of IJCAI'01. Morgan Kaufmann; 2001. p. 155–160. Seattle, USA, Aug. 4-10, 2001.