# Axioms & templates: Distinctions & transformations amongst ontologies, frames, & information models

# Axioms & Templates: Distinctions & Transformations amongst Ontologies, Frames, & Information Models

Alan Rector
School of Computer Science, University of Manchester, Manchester M13 9PL, UK
+44 (0) 114 255 3660    rector@cs.man.ac.uk

## ABSTRACT

The relationships between "ontologies", knowledge bases, and information models – and correspondingly between OWL/Description Logics, frames and UML – remains confusing to many developers. Understanding which to use when and developing effective hybrid systems that exploit the potential synergies requires clarifying key distinctions: between ontology, background knowledge, and information models; between axiom-based and template-based systems; and between logical definitions and queries. As a step towards a more coordinated approach to knowledge-rich systems and a platform for incorporating additional technologies, we propose factoring systems into "ontology (narrow sense)", the rest of the "background knowledge base", and the "information model", with clear distinctions, mutual derivations and interfaces amongst them and clear understanding of the semantics and limitations of each.

## Categories and Subject Descriptors

I.2.4 [**Computing methodologies**]: Artificial intelligence – *knowledge representation formalisms & methods*
H.2.3 [**Information systems**]: Database management – *data description languages*

## General Terms

Design Human Factors, Standardization, Languages

## Keywords

OWL, UML, Ontologies, Frames, Protégé, Description logics

## 1. INTRODUCTION

The relationship between representations in OWL, Frames, and UML remains problematic. "How do I convert OWL to UML/or vice versa?" remain amongst the most frequently asked questions in tutorials on OWL. Analogous issues extend to RDF Schema and SKOS. Furthermore, those working primarily with either frames or OWL often find formulating knowledge in the other difficult, so that the groups tend to work in isolated "silos".

There are numerous papers using the various representations together but no widely accepted guidelines or standards. The OMG Meta model [18] describes OWL constructs in terms of UML, but does not capture OWL's DL semantics. The same is true work such as Brockman *et al.* [8], although it is clearer. Methods such as those of Milanović [26] or Franconi [10] tend to be too heavyweight for many applications and require mastering mechanisms only relevant to advanced features of UML and OWL.

All this despite the fact that OWL is often used as part of the design process for information systems to be specified, ultimately, in UML (*e.g.* [7]), that OWL and frames are both used to represent "ontologies", that OWL developed from attempts to formalize frames, and that RDF Schema, SKOS, and OWL are all part of the Semantic Web "stack" [4].

These problems are now acute in the Healthcare Informatics community, which is trying to reach joint standards for terminologies and Electronic Health Records that require the interaction of very large terminologies represented in OWL/DLs (10K-500K classes) to be used with information models represented in UML or similar formalisms.

Furthermore, in the eyes of at least some researchers, despite the many advances in specific areas, the overall confusion has made knowledge modelling more, rather than less, difficult. We have heard major researchers from both the health informatics[1] and the broader knowledge representation[2] communities complain that our current tools are in many ways less useful and usable than the tools in the mid 1980s.

This paper contends that confusion over at least four distinctions contributes to these difficulties:

- Between "ontology", background knowledge, and information models.
- Between formalisms based on logical axioms – OWL, description (and other) logics – and formalisms based on templates – *e.g.* UML and Frames – and arguably RDF Schema.
- Between class expressions and queries in OWL/Description logics.
- Between the model of the domain and the model of the information system for data about that domain.

These issues are exacerbated because the usage of the word "ontology" varies amongst authors. Some use it so broadly as to encompass the entire background knowledge base; others confine it to definitions and related "universal" or "necessary" truths.

This paper sets out to:

- Advocate a well-defined usage of "ontology (narrow sense)" and establish the usefulness of factoring out the "ontology (narrow sense)" from the rest of the background knowledge base and information systems model.
- Articulate the differences between axiom-based and template-based formalisms and illustrate simple practical derivations between important subsets of each – specifically between OWL, UML and Protégé-frames.

[1] Zak Kohane, Personal communication, 2012
[2] John Sowa, personal communication and email post, 11 July 2012, Ontolog Forum.

- Highlight the difference between class expressions and queries in OWL/DLs.
- Illustrate issues in binding ontologies and information models and argue that, for the key use case of "value sets," the binding of information models to axiom-based ontologies should be to queries rather than to class expressions.
- Suggest an approach to hybrid systems based on these insights.

(Notation: throughout the this paper, we use Manchester syntax for OWL [14] with key words in italics and abbreviations for conciseness: → and ←→ for *SubClassOf* and *EquivalentTo* respectively, *inv ()* for *inverse (),* and & for *and*. Symbols for classes begin in upper case; for individuals and properties in lower case. "First order logic" is abbreviated "FoL".)

## 2. ONTOLOGY (NARROW SENSE)

The original meaning of "ontology" in Aristotelian philosophy was the study of being – or of what exists and their definitions. Borrowed by information systems this corresponds naturally to the entities and used in an information system, their definition and necessary truths about them. Expressed in logic, this corresponds to the set of positive universal statements, *i.e.* of the form:

**FoL:** $\forall x.\ C(x) \ldots \rightarrow \ldots$ (1

Description logics (DLs) and OWL-DL capture a fragment of such axioms, limited to two variables and with various further restrictions depending on the description logic profile [1, 27]. For example, the OWL axioms:

**OWL:** B *SubClassOf* A (2
**OWL:** B *SubClassOf* p *some* C (3
**OWL:** B *EquivalentTo* (A & p *some* C) (4
**OWL:** B *EquivalentTo* (A & p *value* c) (5

are equivalent to, respectively:

**FoL:** $\forall x.\ B(x) \rightarrow A(x)$ (6
**FoL:** $\forall x.\ B(x) \rightarrow \exists y.C(y) \wedge p(x,y)$ (7
**FoL:** $\forall x.\ B(x) \leftrightarrow A(x) \wedge \exists y.C(y) \wedge p(x,y)$ (8
**FoL:** $\forall x.\ B(x) \leftrightarrow A(x) \wedge p(x,c)$ (9

Examples of corresponding statements in natural language are "All pneumonias are infections", "All infections are sited in anatomical structures", "Any infection sited in a lung is a pneumonia", "A Mancunian is any person living in Manchester". In this paper we shall call such axioms "necessary" or describe them as representing "necessities"[3] (where "necessary" includes "necessary and sufficient"). We shall say that an "ontology (narrow sense)" may only consist axioms that may be expressed as variants of (1-9).

Importantly, this excludes statements that are commonly approximated in logic as axioms of the form:

**FoL:** $\exists xy.\ B(x) \wedge C(y) \wedge p(x,y)$ (10

For example: "Some pneumonias are caused by bacteria" or, alternatively, "Pneumonia may be caused by bacteria".

This definition also excludes simple facts. For example: "John lives in Manchester" or "John is a student", *i.e.* statements that follow the pattern in FoL and OWL respectively:

**FoL:** p (a, b) (11
C(a)
**OWL:** a *type* (p *some* b)
a *type* C

[3] We avoid the word "universal" because of the potential for confusion with its use in "universal restriction", "universal quantifier" and its philosophical usage for the entities in ontologies as "universals". Another alternative is "essential", although it raises issues for some philosophers.

However, it does not exclude the use of an individual to define a class as in expressions (5) & (9) – known as "nominals" in DL parlance. By this definition, an "ontology (narrow sense)" is part of the what is known in OWL/DLs as the "T-Box" or "terminology box", but excludes facts about individuals – what OWL/DLs call the "A-Box" or "Assertion-Box" (see [1]) – and excludes as well as statements of the form in (10), which cannot be expressed directly in OWL at all.

This illustrates an important point. Despite its name, OWL – or at least OWL DL and its profiles – are simply logic languages, syntactic variants and embellishments of description logics. Not everything expressed in OWL is, or need be, an ontology (narrow sense), and conversely, not every ontology need be, or can be, expressed in OWL. "Represented in OWL" and "ontology" should not be confused. The phrase "OWL ontology" is often used to refer to anything represented in OWL. This is misleading.

(Other authors, *e.g.* Smith [40], would further restrict the use of "ontology" to just a subset of such statements. That discussion is beyond the scope of this paper. )

The knowledge that is not "necessary" we refer to as "contingent".[4] Variants of expressions of the form shown in (10) we refer to as "contingent generalisations"; simple statements about individuals as in (11) we refer to simply as "contingent facts" – or more simply just as "generalisations" and "facts".

Defined in this way, the ontology (narrow sense) provides the definitions and foundations for a knowledge base. It may be thought of as a "conceptual coat rack" on which to hang other information – a colorful phrase that goes back to Woods [43].[5]

This gives a factoring of the system as shown in Figure 1, where the "Background Knowledge Base" includes the "Ontology", "Contingent", and possibly other forms of knowledge about the domain itself, and the "Information Model" specifies the schemas to hold data in information systems pertaining to the domain.

To make clear the distinction between domain knowledge in the Background Knowledge Base and schemas in the Information model, consider the example of body temperature. Every body necessarily has a temperature (even though it may be ambient). However, not every information system about bodies need include body temperature. Furthermore, a "missing body temperature" is an oxymoron if speaking about the domain; a "missing entry for body temperature" is perfectly reasonable if speaking about data.

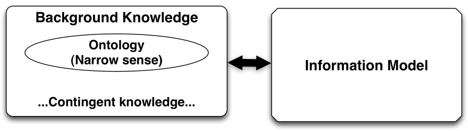


**Figure 1: Proposed factoring**

The "background knowledge base" needed for semantic interoperability and collaborative development often contains many more contingent than necessary statements, *i.e.* the ontology (narrow sense) is just a small part of the background knowledge.

[4] Other authors use "particular" after Aristotle, although this is easily confused with what we here call "facts". "Contingent" here means merely contingent on circumstances in this world, not expressions of the form "if…then…". We have been able to find no consensus on terminology, but "necessary" and "contingent" roughly follow other modern usage, *e.g.* Kripke, and seem to cause the least confusion.

[5] All of the examples above use existential restrictions, (*some*), because this is the only restriction permitted in the OWL profile most widely used in biomedical ontologies, OWL-EL (the OWL variant of $EL^{++}$). However, it can be extended to include universal restrictions (*only*), although to do so is beyond the scope of this paper.

Finally, note that, although ontologies are often presented as hierarchies of subclasses, not all hierarchical structures qualify as ontologies (narrow sense). In particular, thesauri, library catalogues such as the MeSH headings [23], and SKOS [20] use "broader than" and "narrower than" specifically to indicate that no logical inferences are to be drawn from the hierarchy. Additionally, this definition excludes "classifications", such as the International Classification of Diseases (ICD) [44], which contain non-logical mechanisms to avoid "double counting" – *e.g.* "exclusions" and "residual categories" such as "other".

## 3. DL/OWL AXIOMS VS TEMPLATES

There are two broad families of knowledge representations – those based on templates and those based on logical axioms. The most familiar axiom-based system is OWL. Systems based on templates include frame systems such as Protégé-Frames, UML class diagrams, Archetypes [2] (a specification language for data structures and forms much used for Electronic Health Records), and RDF Schema, at least as it is commonly used.

Superficially both axiom-based and template-based systems consist of hierarchies of classes and subclasses, instances of those classes, and relations between classes and/or their instances named variously "properties" (OWL), "slots" (frames), or "associations" (UML) (We shall ignore UML attributes here.)

However, these superficial similarities conceal major differences. Most critically for applications:

- Axiom systems support definitions, logical expressions, composite concepts[6] and their classification by reasoners; most template systems do not, where by "definition" we mean axioms using EquivalentTo as in the manner shown in
- Axiom systems support overall consistency checking by reasoners; template systems do not.
- Template systems support representation of contingent and meta-knowledge; OWL/DLs do not, at least not straightforwardly.
- In template systems, what can be said about any class is well defined by the slots in the template. Axiom systems specify what cannot be said – what would be "unsatisfiable" (a contradiction). There are no fixed semantics for what can be said – *i.e.* for which properties can be used with which classes (sometimes called "sanctioning"). Anything not inconsistent can be asserted.

More fundamentally:

- Templates define structures to be queried. Logical axioms provide assertions from which inferences can be drawn. Beyond simple inheritance down a fixed hierarchy, the semantics of template systems rests largely in the queries used. The semantics of logic systems is fixed by the logic itself, although some also are associated with query languages that can be used to provide additional or alternative semantics.
- Templates permit; logical axioms restrict (although supplementary mechanisms may add constraints to either). Each entry in a template system provides a slot/field/association, which can be filled. Therefore, the more you know about an object, the more fields there are, and, therefore, the more you can say. Axioms restrict; every (nonredundant) axiom reduces the number of valid "models". Therefore, the more you know the less you can say.
- DL/OWL axioms are always disguised necessary statements about the members of a class, as illustrated in (2-9) above. DL/OWL properties always represent relations between individuals. Templates can specify the fields either for information about classes themselves or their members.
- Entries in templates are local to a class and its subclasses. Axioms are global and can affect inferences about any class or property. In particular, new subclass relations can be inferred which may alter the hierarchy drastically.
- Template systems are normally queried under the closed world assumption, with negation as failure. Inference in axiom systems uses open world reasoning with negation as contradiction.
- Template systems often contain constraints – *e.g.* on domain and range – that are local and normally checked at the time of entry. Axioms are global restrictions and checked as part of inference, which is usually a separate step.
- Axiom systems can infer existence from underspecified statements such as "John has a sister"; template systems can only deal with explicit entries: "John's sister is Mary", (although dummy values – "skolem constants" – can provide an approximation of existential quantification).

Critical practical differences include:

- Template systems are normally built in a single step. What you assert is what you get. Axiom systems usually employ a reasoner to draw inferences. What you get is the result of the inference, which may be very different from was asserted. For many purposes, the reasoner may be viewed as an "Ontology compiler", with the asserted form[7] analogous to the source code and the inferred form to the object code.
- Validation of template systems is immediate and straightforward. Violations of constraints are local and easily identified. Validation of axiom systems requires examining the results of inference when the reasoner is run. The results are neither immediate nor local, and unexpected inferences can be difficult to understand [13, 15].
- The semantics of transitive relations are built into most axiom-based systems; they must be simulated in the formulation of queries in most template-based systems.

## 4. OUTLINE PROPOSAL – AXIOMS FOR TEMPLATES

### 4.1 OWL & UML: basic outline

What is proposed here is a set of transformations between the most commonly used constructs in OWL, UML, and frames plus caveats concerning their limitations. The goal is to provide something straightforward for common cases and in so doing to clarify the critical distinctions. The transformations do not aim to be comprehensive and are inevitably lossy. Within these limits, we aim to:

- Provide an easy path between OWL, frames, and UML for common cases.
- Establish the basis for a hybrid environment preserving the value of each representation while allowing users to express themselves intuitively that:
  - Retains axiom systems' capacity for definitions, expressions, consistency checking, and transitive relations

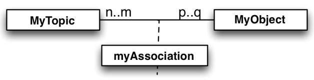
[6] The process of forming what are here called "composite concepts" termed "composition", but we avoid that term because of possible confusion with its use in UML.

[7] SNOMED CT uses the phrase "stated form" for the asserted form. It's inferred form is not distributed directly, but is the basis for its distribution files.

– Retains template systems' capacity to represent contingent and meta knowledge
– Retains template systems' capacity to specify which relations/properties can be used with which classes ("sanctioning").

For presentation purposes, we start by describing a means for representing the core notions of UML in OWL, and then identify a subset of OWL for which the process is reversible, and a further subset for which it can be approximated. (The pattern is related to work by Berardi *et al*. [3] and Severi *et al.* [33].)

The basic pattern is shown below starting with the observation that, although not usually shown in diagrams, every UML association is has a linked class that we represent explicitly in the OWL in a form that we will refer to as "transformed OWL", whose transformation can be reversed in a subset of cases to give "conventional OWL".

**UML**:  (12

**Transformed OWL Schema**:
*Property* to *functional*. (13
*Property* from *functional*
*Class* DomainEntity (14
*Class* Association → (15
  to *some* DomainEntity & (16
  from *some* DomainEntity (17
  has*_key* (to from) (18
*Disjoint*(Association, DomainEntity) (19

**Transformed OWL:**
*Class* MyAssociation *SubClassOf* Association & (20
  to *some* MyObject &
  from *some* MyTopic
*Class* MyTopic *SubClassOf* DomainEntity & (21
  *inv*(from) *min* p MyAssociation &
  *inv*(from) *max* q MyAssociation
*Class* MyObject *SubClassOf* DomainEntity &(22
  to *min* n MyAssociation &
  to *max* m MyAssociation

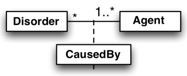
**Conventional OWL**: *Property* myAssoc
  *Domain* MyTopic *Range* MyObject (23
**(if p=1)** MyTopic → myAssoc *some* MyObject (24
**(if p≥1)** MyTopic → myAssoc *min p* MyObject (25
**(if n=1)** MyObject→ *inv*(myAssoc) *some* MyTopic (26
**(if n≥1)** MyObject→ *inv*(myAssoc) *min n* MyTopic (27

The original UML in (12) associates the classes MyTopic and MyObject via myAssociation with the multiplicities shown – *e.g.* we might associate Disorder to Agent via CausedBy. The "transformed OWL" Schema represents the UML almost literally. The properties to and from are declared and made functional – *i.e.* single valued (13). A class DomainEntity is declared (14). A class Association is declared (15) and is asserted to be linked by the properties to and from to Domain Entities (16-17). (Since to and from are functional, this also means that Associations can *only* link to or from DomainEntities; no additional domain axiom is required.[8]) Association and DomainEntity are made disjoint (19) – *i.e.* nothing can be both an Association and a DomainEntity, which is useful for validation. Furthermore, the properties to and from are made the key to the class Association (18). This means that if any two instances of Association link the same instances of Disorder and Agent, then they are the same. This corresponds to the usual practice of banning duplicate rows in a database.

Note that the combination of declaring the properties to and from to be functional – *i.e. max* 1 – (13) and asserting that every Association is from *some* – *i.e. min* 1 – and to *some* DomainEntity is equivalent to saying that each Association is to and from *exactly* 1 DomainEntity. Note also that in OWL, *min* 1 and *some* are equivalent. Further note that maximum multiplicities of ..* and minimum multiplicities of 0.. in UML require no corresponding axiom in the OWL as neither represents a semantic restriction[9].

The transformation on to "conventional OWL" using a property myAssoc to correspond to the UML association – what would usually appear in an ontology implemented in OWL – is only possible in case the minimum multiplicity of MyTopic and/or the MyObject are greater or equal to one, because only in these cases to the statements refer to all instances of the class.

Consider, briefly, a typical example:

**UML**  (28

**Transformed OWL:** *Class* Disorder *SubClassOf* DomainEntity & (29
  *inv*(from) *some* CausedBy (30
*Class* Agent *SubClassOf* DomainEntity
*Class* CausedBy *subClassOf* Association & (31
  from *some* Disorder & (32
  to *some* Agent (33
**Conventional OWL:** Disorder *SubClassOf* causedBy *some* Agent (34

The UML (28) states that every Disorder must be associated with one or more Agents, but that an Agent may, or may not, be associated with any number of Disorders. The transformed OWL asserts that every instance of Disorder must be linked by the inverse of from to some instance of the association CausedBy (30), which must in turn be linked by the property to some instance of Agent (33). The conventional OWL axiom (34) asserts likewise that all Disorders are caused by at least one Agent, and makes no assertion about the inverse relation of Agents to Disorders.

The result captures most of the meaning of the UML in OWL and *vice versa*. It is useful for many purposes, but there are important differences in meaning that will be discussed in Sections 5-8.

An additional advantage of the transformed OWL representation is that it makes it possible to use a class expression to retrieve all associations to or from any domain class, for example, the class expressions in (35) below will be inferred to subsume the association class CausedBy in (32).

Association & to *some* Agent (35

Finding all the relations pertinent to a class[10] ("sanctioning") is a common requirement for those authoring and editing knowledge models that most expect to be supported by an "ontology".

## 4.2 Extension to frames

Since there is no official standard for Frames, we shall refer here to Protégé-frames (version 3.1) as this is currently the most widely used frame system, at least in biomedicine. The semantics are specified operationally in the OKBC draft standard [9]. However, although the standard distinguishes between primitive and non-primitive (*i.e.* defined) classes, neither Protégé nor any other system of which we are aware implements the distinction or supports inference of the classification hierarchy. Indeed, Protégé-frames historically has used "ontology" a broad sense roughly equivalent to "background knowledge base" [30] without distinguishing the "ontology (narrow sense)" as used in this paper.

In Protégé, links between classes, or "slots", are a special form of class, so the structure is closer to the transformed OWL

[8] This schema assumes that all associations are binary. Extension to any finite n-ary model are straightforward by adding additional properties analogous to to and from and extending the has_key axiom.

[9] Although *min* 0 is a legal construct in OWL, no inferences follow from it. There is no OWL construct analogous to *max* *

[10] Sometimes referred to as the relations "sanctioned" for a class.

described above than to conventional OWL. (In addition, Protégé supports a special class BinaryRelation almost exactly parallel to the Association classes in the transformed OWL.) Much of what has been said about UML can, therefore, be carried over directly to Protégé-frames. However, there are important differences.

## 5. METADATA & HIGHER ORDER KNOWLEDGE

Metadata – or data about data – was a key feature in most early knowledge representation systems (*e.g.* see [37]) and is a key feature of Protégé-Frames, although its various functions were not always well distinguished. We here distinguish three:

- Editorial information and comments, including versioning, provenance, authority, intellectual property, etc.
- Extending the functionality of the system – *e.g.* for calculations, pointers to external resources, hints to the user interface, etc.
- Higher order domain knowledge about the category represented by the class (See 5.1)

Most systems, including OWL, include mechanisms for annotations or comments for editorial knowledge, and this mechanism is often co-opted for extending functionality. Where there is a difference between template- and axiom-based systems is in the representation of higher order domain knowledge.

### 5.1 Higher order knowledge in frames

The key mechanisms for metadata in Protégé-Frames are "own slots" and the use of "classes as values". "Own slots" are slots that apply to the class itself and are not inherited by subclasses. This is in contrast to the more common "template slots", which, as the name implies, form the templates for information about all instances of the class and, therefore, are inherited by subclasses. Template classes correspond to properties in OWL and to associations in UML.

The use of own slots that concerns us here is to support higher order statements about the domain, *e.g.* "This class represents a species", "This species was first described by Alfred Russell Wallace in 1847", or "The prevalence of this disease in 2012 in the United Kingdom was 15/100000 of the population." None of these statements apply to any individual member of the class; they apply to the category represented by the class.

The converse of the use of own slots that carry information about a class, as opposed to all its members, is the use of the class itself as a value, again to convey higher order information: for example, "Books about Pneumonia" – *i.e.* the disease pneumonia – as opposed to books about some specific cases (instances) of pneumonia. In addition classes are often used as values in expressions such as "Smith has Pneumonia".

### 5.2 Higher order knowledge in OWL

OWL 1 provided no construct for higher order domain knowledge other than as annotations. From early in OWL's history, how best to approximate the higher order knowledge and the use of classes as values was an issue [31].

"Puns" were introduced into OWL 2 [29] as a weak mechanism for representing higher order knowledge. OWL 2 allows the same name to be used for a class, property and individual – a pun on the name. However, it explicitly excludes any formal connection between the different entities represented by the same name. Hence, in OWL 2, we could represent "Book & is_about value Pneumonia" or the fact "Pneumonia prevalence 0.00015" the reasoner would make no connection between the individual Pneumonia in these statements and the class Pneumonia in "Pneumonia SubClassOf LungDisorder". In principle, the connection can be made by a query language, although currently the most obvious query language, the OWL entailment regime of SPARQL 1.1 [11] does not support such queries straightforwardly, so that such queries may be more easily programmed via the OWL API.

## 6. CONTINGENT GENERALISATIONS

Consider a typical statement from a medical knowledge base, which we here term a "contingent generalization":

"Pneumonia may be caused by Bacteria" (36

Typically, in logic textbooks, this would approximated by something like "Some pneumonia is caused by some bacteria"

$$\exists xy.\ \mathrm{Pneumonia}(x) \wedge \mathrm{Bacteria}(y) \wedge \mathrm{causedBy}(x,y) \qquad (37$$

The FoL statement in (37) is reciprocal – *i.e.* if pneumonia may be caused by bacteria, then bacteria may cause pneumonia. However, weaker than our usual natural language of the corresponding of (36), which we usually take as having a further meaning that bacteria are a significant or noteworthy cause of pneumonia. To go beyond this requires going beyond simple first order logic to some formalism incorporating uncertainties, probabilities or fuzziness. However, for many practical purposes, the FoL statement (37) is an adequate approximation. Provided we accept the extra-logical semantics that only "interesting" such statements are made, we can answer questions such as "What are the causes of pneumonia?"

Note that contingent generalisations are not "inherited" – there are kinds of pneumonia that are not caused by bacteria, and there are kinds of bacteria that do not cause pneumonia.

### 6.1 Contingent generalisations in Frames

Protégé-frames makes no explicit distinction between necessary and contingent knowledge. However, much of what is implemented in typical knowledge bases in Protégé is contingent.

In most knowledge bases in Protégé frames, there would be an entry in a slot "CausedBy" for the class Bacteria, and the slot would be many-valued. This might be supplemented by a default value of Bacteria, but this simply means that for any instance, this is the value initially instantiated. Furthermore, the information would be inherited by all kinds of pneumonias. The multiplicities of the slot might also make it optional, so that there was no requirement to enter any cause for a pneumonia.

This captures neither the reciprocity of the statement nor the fact that the value should not necessarily be inherited. These issues are typically dealt with in the query and constraint languages associated with frame systems.

Alternatively, the binary associations class provides a means to achieve reciprocal relations, although the rules for inheritance remain problematic. Alternatively still, such statements may be treated as higher order statements about the classes themselves, as discussed in 5.1.

Despite these caveats, because frames offer primarily data structures with limited semantics, queries, scripts or programs can and are used to capture contingent generalisations with appropriate semantics for specific applications.

### 6.2 Contingent generalisations in OWL

OWL was not designed to, and does not naturally, support contingent generalisations. Simple expressions such as that in the conventional OWL in (34) are always necessary.

However, the transformed OWL schema from (13-19) provides a useful approximation analogous to that for optional associations in UML:

```
Class CausedByPneumoniaPneumococcus →                    (38
      from some Pneumonia &
      to some Bacteria
```

Literally, there is a class of causal associations between pneumonias and pneumococci. This statement is even weaker than the FoL statement (37), since it does not even imply that the class CausedByPneumoniaPneumococcus has any instances. However, just as in the case of the FoL approximation, we can use a collection of such axioms to get the expected answers, to questions such as "What are the causes of pneumonia?" The extra assumptions required to use the OWL approximation are, arguably, no more onerous than those for the FoL.

## 7. OTHER ISSUES

### 7.1 Queries vs class expressions

In axiom-based systems, it is important to distinguish between class expressions and queries, even though class expressions can be used analogously to queries in some cases.

Class expressions are first order expressions about the domain and subject to all the restrictions on the DL language and hence cannot refer to the classes themselves – *i.e.* cannot explore higher order information. Queries are fundamentally about the representational artifact itself. Their variables bind to symbols in that artifact. There is no requirement for complete algorithms for subsumption between queries. Therefore they may contain constructs excluded from class expressions, *e.g.* equality, functions between classes, and negation as failure. In particular, variables can bind direct to classes as values so that it is possible to query an OWL ontology for a class used as a value. However, the query cannot be used to form a class expression that will persist in the ontology and be classified by the reasoner.

The differences are fundamental but not always obvious in practice. They can, therefore, be used as an approximation to capture many higher order notions, *e.g.* all classes annotated as "species", or "a species first described by Darwin". An important query is for a class except for its subclasses subsumed by some other class (as opposed to being disjoint with that class). Consider, to find all the subclasses of hypertension explicitly caused by pregnancy, we can use the OWL expression in (39) below which will be inferred to subsume the expected classes:

Hypertension & causedBy *some* Pregnancy (39

However, to find the subclasses of hypertension excluding those classified as being caused by pregnancy – an important category in the ICD – the expression in (40 below) does not work as expected, because it only retrieves the subclasses of hypertension asserted or inferred to be disjoint with causedBy *some* Pregnancy – *i.e.* those kinds of Hypertension *necessarily not* caused by Pregnancy.

Hypertension & *not* (causedBy *some* Pregnancy) (40

Most kinds of hypertension do not have any relation to pregnancy, causal or otherwise. Some might be caused by pregnancy in some cases and not others. To retrieve the required classes, we need a query about the ontology itself, to find the classes not classified under Hypertension causedBy *some* Pregnancy (40), *i.e.* those *not necessarily* caused by pregnancy. Therefore, we need negation as failure against the content of the ontology rather than negation as contradiction against the domain. In SPARQL 1.1 [11], the core of such a query would be:

Hypertension *MINUS* (causedBy *some* Pregnancy) (41

The result of the query in (41) would be just those classes subsumed by Hypertension but not by (causedBy some Pregnancy). Such queries are particularly important when using the OWL-EL profile, which excludes negation in any form.

More generally, distinction between the DL expression for *necessarily not* and the query for *not necessarily* is critical when the ontology is being used to expand queries against a database. To retrieve all individuals who definitely do not have a characteristic requires querying for the union of those that *necessarily* do not have it, based on the concept definitions in the ontology, plus those for which it has been explicitly excluded in the database (and *vice versa* for those that *necessarily* do have the characteristic and those that are explicitly stated to have it.) What is inferred from the ontology to be necessary is often not explicit in the database because it seems redundant to users.

### 7.2 Constraints vs restrictions

Both OWL and Protégé support constructs for domain and range ("allowable values" in Protégé). In UML they are implicit, but can be checked by most database systems. However, it is important to remember that in an axiom-based language, domain and range statements are axioms to be used for inference rather than constraints to be checked. Because the properties to and from are functional, the axioms in (15) will be used to infer that anything in that fills one of them is a subclass of DomainEntity. If this inference causes a contradiction – *e.g.* if the class is provably disjoint with DomainEntity – the reasoner would find the association to be inconsistent ("unsatisfiable"). However, this would only occur when the reasoner was run. Furthermore, the inference that led to the conclusion that the association was a subclass of two disjoint classes might depend on a complex chain of reasoning and be hard to locate and correct.

More seriously, if no inconsistency were encountered, the association class would simply be added to an additional place hierarchy – a change that can be difficult to spot. Misunderstandings over domain and range constraints are one of the most common sources of difficulties in dealing with OWL.

When used for validating instances, these consideration make using OWL on its own problematic. Users expect closed world reasoning, and few OWL models are, or can be, sufficiently restricted to infer all invalid instances to be unsatisfiable.

There is work on combining closed world constrains with OWL [28], but although included in one classifier, Hermit [19], it is not part of any standard or widely supported by tools.

### 7.3 Transitive relations and property paths

A particularly useful feature of OWL is the ability to declare properties as transitive, something that can be implied in UML diagrams but must be embedded explicitly in queries of the resulting databases. A reasonable approximation can be retained in the transformed OWL thanks to the property path mechanism in OWL 2. To outline the mechanism briefly:

What we would like to say is that there is an Association, IsPartOf and the path below implies a property "is_part_of:

*inv*(from)…IsPartOf…to → is_part_of (42

However, OWL 2 only supports paths along properties. There can be no intervening classes and no cycles. So the best approximation is to define subproperties of to and from for each transitive association and use tools to enforce that the correct subproperties are used with the correct associations, for example defining ipo_to and from_ipo as subproperties of to and from respectively, then we can use a property path:

*inv*(from_ipo) o ipo_to → is_part_of (43

This says that any chain of properties across inv(from_ipo) and ipo_to alternatively implies the property is_part_of. (The small circle symbol is read "composed with"). Unfortunately, this means that the intervening associations are also parts, so the statement is weaker than we would like. This can be circumvented by exclude Associations explicitly in queries. Given that DomainEntity and Association are disjoint, expressions of the form (44) below subsume all and only the expected results.

Anatomic_structure & is_part_of *some* Lung. (44

## 8. INFORMATION MODELS AND KNOWLEDGE BASES

Since the Background Knowledge Base and Information Model are different, the interface – or "binding" – between them is important. There are at least three types of binding. The ontology may act as: i) "value sets" for the data structures [35], ii) indexes to entities in the information model, or iii) a means of generating the information model dynamically [34].

Only the use as "value sets" is within the scope of this paper. This issue of is particularly acute currently in healthcare because standard information models – *e.g.* HL7 or Archetypes – are developed independently from the terminologies/ontologies intended to provide their value sets. However, our experience is that it is equally true for more coordinated developments.

Consider a data structure for "provisional diagnosis". It is to be filled only with subclasses of the class Disorder. If any disorder is to be allowed, this is easy to capture by a class expression. However, the typical requirement is that there are only certain subclasses of Disorder that are allowed, whether for clinical, human factors, or administrative reasons.

There is no way in an OWL expression to identify specific classes without including all of their subclasses. In OWL-EL, which does not support negation, there is no way to exclude classes at all. Furthermore, the requirement is often to exclude classes not proven to be subclasses rather than proven not to be subclasses – to express *not necessarily* rather than *necessarily not*. To achieve the required flexibility requires using queries (See 7.1) rather than class expressions, at least for such difficult cases. Our proposal is that if OWL is used for the ontology, then, for uniformity, all such bindings to data structures be via queries.

## 9. DISCUSSION & CONCLUSIONS

Before the advent of OWL and description logics, the distinctions in this paper could be left ill defined. Because frame systems and information models are both template-based, and because template-based systems allow blurring of the boundary between necessary and contingent knowledge and between domain knowledge and information model, ignoring the distinctions had few consequences. Given the importance of OWL/ today, this is no longer true.

We advocate is a tripartite division of the representation into:

- Ontology (narrow sense) – necessary knowledge
- Contingent and other knowledge – the rest of the Background Knowledge Base
- Information model – data structures capture and manage the information based on the ontology and background knowledge.

Our goals are to achieve an environment and language that provides a high level of abstraction over the underlying formalisms. Our key requirements are to:

- Provide standard methods of combining ontologies (narrow sense), broader knowledge representations, and object oriented modelling – and correspondingly OWL, Frames, and UML – that respect the semantics of each.
- Retain the ability to compose expressions and composite concepts and classify them with a reasoner. Otherwise, maintenance is difficult, and we are condemned to combinatorial proliferation of concepts – ludicrous examples of which have even reached the Wall Street Journal [25].
- Provide means of capturing and querying contingent and higher order knowledge with sound semantics, not just as annotations.
- Provide a standard means for the interface (binding) between the ontology and background knowledge base and information model.

There are alternative approaches to such a programme: *e.g.* Sowa's Conceptual Graphs [41], F-Logic [21], Common Logic, or GRAIL [36]. This paper builds on OWL, Frames and UML because of the large number of applications already committed to these technologies and ability to take advantage of existing and likely future advances – *e.g.* the proposed "Rich Annotations" [24] mechanisms or layered OWL extensions, *e.g.* OWL-FA [32].

Within OWL, contingent knowledge such as "Pneumonia caused by bacteria" might alternatively be represented by defining explicit subclasses and showing that they were satisfiable – *e.g.*

Pneumonia_caused_by_bacteria ←→ (45
    Pneumonia *and* caused_by *some* Bacteria

However, this is not inherently reciprocal which is the natural meaning of the statement and entailed by the FoL in (10) in Section 2. If "Pneumonia may be caused by bacteria" then "Bacteria may cause pneumonia".

Bacteria_that causes_pnuemonia ←→ (46
    Bacteria *and* causes *some* Pneumonia

This would require tools to enforce the constraint that such subclasses were always created in pairs analogous to (45-46). More seriously, classes created simply to indicate contingent knowledge could rapidly clutter the hierarchy of domain entities. These considerations, combined with the natural links to frames and UML, and the fact that it provides a method for determining which relations are "sanctioned" for each class, lead us to prefer the strategy using "transformed OWL" as in Section 4.

This proposal is motivated by experience in working with medical terminologies – SNOMED CT, the new ICD revision, and their harmonization [38, 39] – and clinical systems in commercial collaborations past [5, 6, 22] and present.[11]

We have also built small demonstration systems using "transformed OWL" as described here plus UML using *ad hoc* combinations of the scripting language OPPL [16, 17], grammar transformations, Protégé-Owl, Protégé-frames, and standard UML tools. We hope experiment in the future with replacing some of the use of OPPL with SPARQL 1.1 [11] as implemented in the Hermit reasoner [19] and the use of the knowledge exploration extensions to the FaCT++ reasoner [42]. However, to develop these methods further requires an integrated environment making the derivations and transformation transparent to the user, something we would hope this paper will help to stimulate. Our experience leads us to believe that building such an environment is feasible. It would provide a natural meeting point for those working in frames, OWL, or UML. No new breakthroughs are required. Indeed, the individual technologies are well established.

Such environments might eventually go beyond the limited list of technologies that space has allowed us to discuss here. The first priority is to extend the derivations and experiments to RDF Schema. We would make other high priority incorporation of rules, whose relation with description logics and OWL is far from straightforward but at least partially understood [12]. Following that, at least for the biomedical community, the next priority would be to interface with modern Bayesian probabilistic methods, which would require serious research.

However, even without such an environment – indeed especially without it – developers need to understand the distinctions made here. Otherwise, they will continue to try to use representations for purposes for which they are not appropriate

[11] Details of the commercial collaborations are, unfortunately, currently confidential.

and be frustrated when they do not perform as expected, and they will continue to work in "silos" – often making *ad hoc* extensions and "reinventing wheels" rather than reusing standard methodologies.

## ACKNOWLEDGMENTS

This work supported in part by collaboration with Siemens Health Services. Our thanks to the Description Logic and Clinical Systems Groups at University of Manchester and to the IHTSDO-WHO Joint Advisory Group on integration of SNOMED CT and ICD11.

## REFERENCES

[1] Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D., and Patel-Schneider, P. F. 2007 The description logic handbook, edition 2. Cambridge University Press.

[2] Beale, T. 2002. Archetypes: Constraint-based domain models for future-proof information systems. OOPSLA-2002 Workshop on behavioural semantics.

[3] Berardi, D., Calvanese, D., and De Giacomo, G. 2005. Reasoning on UML Class Diagrams. Artificial Intelligence. 168, 70-118.

[4] Berners Lee, T. 2003. WWW past & future. http://www.w3.org/2003/Talks/0922-rsoc-tbl/ (accessed 2012)

[5] Bouamrane, M. M., Rector, A., and Hurrell, M. 2009. A hybrid architecture for a preoperative decision support system using a rule engine and a reasoner on a clinical ontology. Web Reasoning and Rule Systems. 242-253.

[6] Bouamrane, M. M., Rector, A., and Hurrell, M. 2011. Using OWL ontologies for adaptive patient information modelling and preoperative clinical decision support. Knowledge and information systems. 29, 2, 405-418.

[7] Brachman, R. J. and Levesque, H. J. 1984. The tractability of subsumption in frame-based description languages. AAAI-84. 34-37.

[8] Brockmans, S., Colomb, R., Haase, P., Kendall, E., Wallace, E., Welty, C., and Xie, G. 2006. A model driven approach for building OWL DL and OWL Full ontologies. The Semantic Web-ISWC 2006. 187-200.

[9] Chaudhri, V. K., Farquhar, A., Fikes, R., Karp, P. D., and Rice, a. 1998. OKBC: A programmatic foundation for knowledge base interoperability. 15th National Conference on Artificial Intelligence (AAAI 98). 600-607.

[10] Franconi, E., Mosca, A., and Solomakhin, D. 2012. ORM2 encoding into Description Logics. 2012 International Description Logics workshop (DL-2012), Rome, Italy, DL-2012.

[11] Harris, S. and Seaborn, A. 2012. The SPARQL 1.1 Query Language (W3C recommendation 21 March 2013. http://www.w3.org/TR/sparql11-query/ (accessed 26 Mar 2013)

[12] Hitzler, P. and Parsia, B. 2009. Ontologies and rules. Handbook on Ontologies. 111–132.

[13] Horridge, M., Parsia, B., and Sattler, U. 2010. Justification oriented proofs in OWL. International Semantic Web Conference (ISWC 2010). 354-369.

[14] Horridge, M. and Patel-Schneider, P. F. 2009. OIWL 2 Web Ontology language: Manchester Syntax. http://www.w3.org/TR/owl2-manchester-syntax/

[15] Horridge, M., Parsia, B., and Sattler, U. 2009. Computing Explanations for Entailments in Description Logic Based Ontologies. 16th Automated Reasoning Workshop (ARW 2009), ARW 2009.

[16] Iannone, L., Aranguren, M. E., Rector, A., and Stevens, R. 2008. Augmenting the expressivity of the ontology pre-processor language. OWL Experiences and Directions (OWLEd 2008).

[17] Iannone, L., Rector, A., and Stevens, R. 2009. Embedding knowledge patterns into OWL. European Semantic Web Conference (ESWC 2009). 218-232.

[18] IBM, Sandpiper Software Inc. 2005. Ontology Definition Metamodel: Third revised submission to OMG. 310.

[19] Information Systems Group Department of Computer Science University of Oxford 2012. Hermit OWL Reasoner Home Page. http://hermit-reasoner.com/ (accessed 2012)

[20] Isaac, A. and Summers, E. 2009. SKOS Simple Knowledge Organization System Primer. http://www.w3.org/TR/skos-primer/

[21] Kifer, M. and Lausen, G. 1989 F-logic: a higher-order language for reasoning about objects, inheritance, and schemas. ACM Press.

[22] Kirby, J. and Rector, A. L. 1996. The PEN&PAD Data Entry System: From prototype to practical system. AMIA Fall Symposium. 709-713.

[23] Lipscomb, C. E. 2000. Medical subject headings (MeSH). Bulletin of the Medical Library Association. 88, 3, 265.

[24] Luciano, J. and Parsia, B. RichAnnotations. http://code.google.com/p/owl1-1/wiki/RichAnnotations

[25] Matthews, A. W. 2011. Walked Into a Lamppost? Hurt While Crocheting? Help Is on the Way. New Medical-Billing System Provides Precision; Nine Codes for Macaw Mishaps. Wall Street Journal. The A-HED, 13 Sep 2011 http://tinyurl.com/5tpxhjc.

[26] Milanović, M., Gašević, D., Giurca, A., Wagner, G., and Devedžić, V. 2006. On interchanging between owl/swrl and uml/ocl. Proceedings of 6th Workshop on OCL for (Meta-) Models in Multiple Application Domains (OCLApps) at the 9th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems (MoDELS), Genoa, Italy. 81–95.

[27] Motik, Cuenco-grau, *et al*. 2009. OWL 2 Web Ontology Language Profiles. http://www.w3.org/TR/owl2-profiles/ (accessed 2011)

[28] Motik, B., Horrocks, I., and Sattler, U. 2007. Adding integrity constraints to OWL. Third OWL Experiences and Directions Workshop (OWLEd-2007).

[29] Motik, B., Patel-Schneider, P. F., and Parsia, B. 2009. OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax. (accessed 2011)

[30] Musen, M. A. 1998. Domain ontologies in software engineering: use of Protégé with the EON architecture. Methods of Information in Medicine. 37, 4, 540-550.

[31] Noy, N. 2005. Representing classes as property values on the semantic web. 2005(March 2006), March 2006,. http://www.w3.org/TR/swbp-classes-as-values/

[32] Pan, J. Z., Horrocks, I., and Schreiber, G. 2005. OWL FA: A metamodeling extension of OWL DL. Proc International workshop on OWL: Experiences and Directions (OWL-ED2005).

[33] Parreiras, F. S., Staab, S., and Winter, A. 2007 TwoUse: Integrating UML models and OWL ontologies. Citeseer.

[34] Pulestin, C. and Parsia, B. 2012. The HOBO hybrid modelling framework. OWL Experiences and Directions (OWLEd 2012). http://ceur-ws.org/Vol-849/paper_15.pdf.

[35] Rector, A., Qamar, R., and Marley, T. 2009. Binding ontologies and coding systems to electronic health records and messages. Applied Ontology. 4, 1, 51-69.

[36] Rector, A. L., Bechhofer, S., Goble, C. A., Horrocks, I., Nowlan, W. A., and Solomon, W. D. 1997. The GRAIL concept modelling language for medical terminology. Artificial intelligence in Medicine. 9, 2, 139-171.

[37] Ringland, G. A. and Duce, D. A. 1988 Approaches to Knowledge Representation: An Introduction. John Wiley.

[38] Schulz, S., Rector, A., Rodrigues, J. M., Chute, C. G., Üstün, B., and Spackman, K. 2012. Ontology-based convergence of medical terminologies: SNOMED CT and ICD 11. eHealth 2012 - Health Informatics meets eHealth.

[39] Schulz, S., Spackman, K., and Jame, A. 2011. Scalable representations of diseases in biomedical ontologies. Journal of Biomedical Semantics.

[40] Smith, B. 1998. The basic tools of formal ontology. Formal Ontology in Information Systems (FOIS). 19-28.

[41] Sowa, J. 1985 Conceptual Structures: Knowledge Representation in Mind and Machine. John Wiley & Sons.

[42] Tsarkov, D. and Palmisano, I. 2012. Divide et impera: Metareasoning for large ontologies. OWL Experiences and Directions (OWLEd 2012). http://ceur-ws.org/Vol-849/paper_3.pdf.

[43] Woods, W. A. 1975 What's in a link: Foundations for semantic networks. In Representation and Understanding: Studies in Cognitive Science, D. G. Bobrow and A. M. Collins, Eds. Academic Press.

[44] Organization, W. H. 1993 The ICD-10 classification of mental and behavioural disorders: diagnostic criteria for research. World Health Organization.