Ontology Engineering -The DOGMA Approach

Mustafa Jarrar Robert Meersman

STARLab, Vrije Universiteit Brussel, Belgium,
{mjarrar | meersman}@vub.ac.be

Abstract. This chapter presents a methodological framework for ontology engineering (called DOGMA), which is aimed to guide ontology builders towards building ontologies that are both highly reusable and usable, easier to build and to maintain. We survey the main foundational challenges in ontology engineering and analyse to what extent one can build an ontology independently of application requirements at hand. We discuss ontology reusability verses ontology usability and present the DOGMA approach, its philosophy and formalization, which prescribe that an ontology be built as separate domain axiomatization and application axiomatizations. While a domain axiomatization focuses on the characterization of the intended meaning (i.e. intended models) of a vocabulary at the domain level, application axiomatizations focus on the usability of this vocabulary according to certain application/usability perspectives and specify the legal models (a subset of the intended models) of the application(s)' interest. We show how specification languages (such as ORM, UML, EER, and OWL) can be effectively (re)used in ontology engineering.

***Published as**: Mustafa Jarrar and Robert Meersman: Ontology Engineering -The DOGMA Approach. Book Chapter (Chapter 3). In Advances in Web Semantics I. Volume LNCS 4891, Springer. 2008.*

1 Introduction and motivation

The Internet and other open connectivity environments create a strong demand for sharing the semantics of data. *Ontologies* are becoming increasingly essential for nearly all computer science applications. Organizations are looking towards them as vital machine-processable semantic resources for many application areas. An ontology is an agreed understanding (i.e. semantics) of a certain domain, axiomatized and represented formally as logical theory in the form of a computer-based resource. By sharing an ontology, autonomous and distributed applications can *meaningfully* communicate to exchange data and thus make transactions *interoperate* independently of their internal technologies.
Research on ontologies has turned into an interdisciplinary subject. It combines elements of Philosophy, Linguistics, Logics, and Computer Science. Within computer science, the research on ontologies emerged "mainly" within two subcommunities: artificial intelligence (among scientists largely committed to building shared knowledge bases) and database (among scientists and members of industry who are largely committed to building conceptual data schemes, also called semantic data models [V82]). This legacy in computer science brings indeed successful techniques

and methods to enrich the art of ontology engineering. However, some confusion on how to reuse these techniques is witnessed. For example, many researchers have confused ontologies with data schemes, knowledge bases, or even logic programs.

Unlike a conceptual data schema or a "classical" knowledge base that captures semantics for a given enterprise application, the main and fundamental advantage of an ontology is that it captures domain knowledge highly independently of any particular application or task [M99b] [JDM03]. A consensus on ontological content is the main requirement in ontology engineering, and this is what mainly distinguishes it from conceptual data modeling. Neither an ontology nor its development process is a single person enterprise [KN03].

The main goal of this chapter is to present a methodological framework for ontology engineering (called DOGMA[1] [M01] [J05]) to guide ontology builders towards building ontologies that are both highly reusable and usable, easier to build, and smoother to maintain. Some parts of this research has been published in earlier articles such as [M96] [M99a] [J05a] [M99b] [JM02a] [JDM03] [J06] [JH07] [J07c]. This chapter provides an up-to-date specification of DOGMA based on [J05]. A comprehensive view on DOGMA, its formalisms, applications, and supportive tools can be found in [J05]. Others have also enriched our approach with special techniques for ontology evolution [DDM07] [DM05], community-driven ontologies [DDM06], ontology integration [DSM04], and visualization [P05] [TV06].

Before presenting the DOGMA methodological framework, we investigate and illustrate (in section 2) the main *foundational* challenges in ontology engineering. We argue that different usability perspectives (i.e. different purposes of what an ontology is made for and how it will be used) hamper reusability and lead to different or even to conflicting ontologies, although these ontologies might intuitively be in agreement at the domain level. The more an ontology is independent of application perspectives, the less usable it will be. In contrast, the closer an ontology is to application perspectives, the less reusable it will be. From a methodological viewpoint, notice that if a methodology emphasizes usability perspectives, or evaluates ontologies based only on how they fulfill specific application requirements, the resultant ontology will be similar to a *conceptual data schema* (or a classical knowledge base) containing specific –and thus less reusable– knowledge. Likewise, if a methodology emphasizes only on the independence of the knowledge and ignores application perspectives, the resultant ontology will be less usable.

To tackle such a foundational challenge, we propose a methodological framework, called DOGMA, in section 3. The idea of DOGMA, in nutshell, is that: an ontology is doubly articulated into a *domain axiomatization* and *application axiomatization*[2]. While a domain axiomatization is mainly concerned with characterizing the "intended meanings" of domain vocabulary (typically shared and public), an application axiomatization (typically local) is mainly concerned with the usability of these

[1] Developing Ontology-Grounded Methods and Applications.

[2] We use the term axiomatization in this chapter to mean an articulation or specification of knowledge, as a set of axioms about a certain subject-matter. We interchange this term with the term ontology in order to avoid the any misunderstanding of what an ontology is. For example, a conceptual data schema or an application's knowledge base is being named as ontologies, which in our opinion are only application axiomatizations. We preserve the term ontology to mean a domain axiomatization, which captures knowledge at the domain level.

vocabularies. The double articulation implies that all concepts and relationships introduced in an application axiomatization are predefined in its domain axiomatization. Multiple application axiomatizations (e.g. that reflect different usability perspectives, and that are more usable) share and reuse the same intended meanings in a domain axiomatization.
To illustrate this framework, one can imagine WordNet as a domain axiomatization, and an application axiomatization built in OWL (or RDF, ORM, UML, etc). The DOGMA methodology then suggests that all vocabulary in the OWL axiomatization should be linked with word-senses (i.e. concepts) in WordNet. In this way, we gain more consensus about application axiomatizations; we improve the usability of application axiomatizations and the reusability of domain axiomatizations; application ontologies that are built in the same way (i.e. commit to the same domain axiomatizations) will be easier to integrate, etc.
The DOGMA approach goes farther and suggests the notion of "ontology base" for representing domain axiomatizations in a manner easy to evolve and use (see section 3). The basic building block of ontology base is called *lexon*, a content-specific lexical rendering of a binary conceptual relation. As we shall explain later application axiomatizations then *commit* to an ontology base containing those lexons.

2 Fundamental Challenges in Ontology Engineering

In this section, we investigate and specify several challenges in ontology engineering. We examine to what extent one can build an ontology independently of application requirements. We discuss ontology reusability verses ontology usability, then we present the work done by other researchers in relation to these challenges. To end, we draw some important requirements for ontology engineering.
Ontologies are supposed to capture knowledge at the domain level independently of application requirements [G97] [CJB99] [M99a] [SMJ02]. We may argue this is in fact the main and most fundamental asset of an ontology. The greater the extent to which an ontology is independent of application requirements, the greater its reusability[3], and hence, the ease at which a consensus can be reached about it. Guarino indeed in [G97] posited that*"Reusability across multiple tasks or methods should be systematically pursued even when modeling knowledge related to a single task or method: the more this reusability is pursued, the closer we get to the intrinsic, task-independent aspects of a given piece of reality (at least, in the commonsense perception of a human agent)."*

Ontology application-independence is not limited to the independence of *implementation* requirements - it should also be considered at the *conceptual level.* For example, notice that application-independence is the main disparity between an ontology and a classical *data schema* (e.g. EER, ORM, UML, etc.) although each captures knowledge at the conceptual level [JDM03]. Unlike ontologies, when

[3] *Notice that ontology usability is subtly different from ontology reusability.* Increasing the reusability of knowledge implies the maximization of its usage *among several kinds* of tasks. Increasing ontology usability could just mean maximizing the number of different applications using an ontology for the *same kind* of task.

building a data schema the modeling decisions depend on the specific needs and tasks that are planned to be performed within a certain enterprise, i.e. for "in-house" usage. The problem is that when building an ontology, there always will be intended or expected usability requirements -"at hand"- which influence the independency level of ontology axioms. In the problem-solving research community, this is called the *interaction problem*. Bylander and Chandrasekaran[BC88] argue that *"Representing knowledge for the purpose of solving some problem is strongly affected by the nature of the problem and the inference strategy to be applied to the problem."*
The main challenge of usability influence is that different usability perspectives (i.e. differing purposes of *what an ontology is made for* and *how it will be used*) lead to different –and sometimes conflicting– application axiomatizations although these might agree at the domain level.

Example

The following example illustrates the influence of some usability perspectives when modeling ontologies in the Bibliography domain. Ontology A in Figure 1 and ontology B in Figure 2 are assumed to have been built autonomously; ontology A is built and used within a community of bookstores, and ontology B is built and used within a community of libraries[4]. Although both ontologies "intuitively" agree at the domain level, they differ formally because of the differences in their communities' usability perspectives. Obviously, building ontologies under strong influence of usability perspectives leads to more application-dependent, and thus less reusable ontologies.

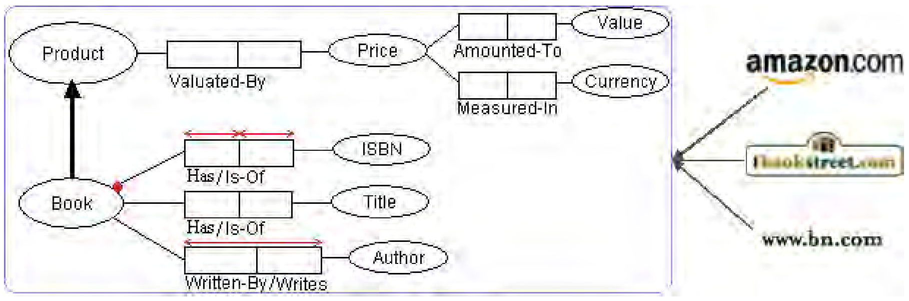


Fig. 1. Ontology A.

[4] Notice that the goal of this example is neither to discuss the Bibliography domain itself, nor to present adequate an ontology - we use it only for illustration purposes.

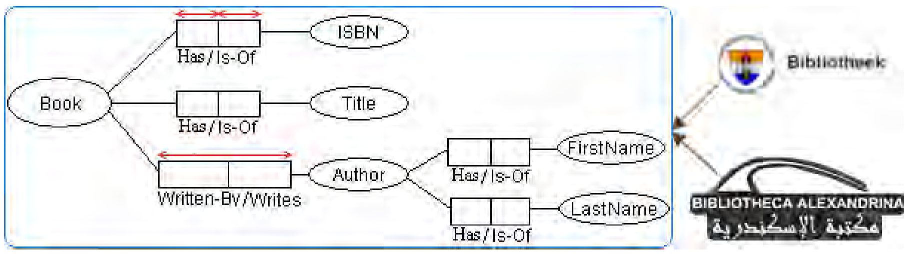


Fig. 2. Ontology B.

In the following, we examine the influence of usability perspectives on the modeling decisions of both conceptual relations and ontology rules, respectively.

Modeling conceptual relations. The concept 'Author' in ontology B is attributed with the 'First Name' and the 'Last Name' concepts. Such details (i.e. *granularity*) are not *relevant* to bookstore applications; they are not specified in ontology A. Similarly, unlike ontology A, the pricing relations {Valuated-By(Book, Price), Amounted-To(Price, Value), Measured-In(Price, Currency)} are not *relevant* for library applications, so they are not specified in ontology B.

From such differences, one can see that deciding the *granularity level* and the *scope boundaries* depend on the relevance to the intended (or expected) usability. Although such differences do not necessarily constitute a disagreement between both axiomatizations, they hamper the reusability of both ontologies. In order to reuse such ontologies, the reusing applications need to make some adaptations, *viz. introducing* the incomplete knowledge and *dismissing* the "useless" knowledge that normally distracts and scales down the reasoning/computational processes.

Modeling ontology rules. Notice that both ontologies in the example above do not agree on the notion of what is a "Book". Although both ontologies agree that the ISBN is a unique property for the concept book (see the uniqueness rules[5]), they disagree whether this property is mandatory for each instance of a book. Unlike ontology B, ontology A axiomatizes that each instance of a book *must* have an ISBN value (see the mandatory rule[6]). This rule implies for example that "PhD Theses" or "Manuals", etc. would not be considered instances of books in ontology A because they do not have an ISBN, while they would be under ontology B.

One can see from this example that modeling the ISBN as mandatory property for all instances of the concept book is naturally affected by bookstores' business perspective. Obviously, bookstores communicate only the books "that can be sold" and thus "commercially" should have ISBN, rather than perusing the notion of book at the domain level. Nevertheless, at the domain level, both bookstore and library applications intuitively share the same concept of what is really a book. For example, suppose that one assigns an ISBN for an instance of a "PhD Thesis". This instance can then be considered as a book for bookstores. If however, the ISBN is removed for an instance of a book, then this instance will no longer be a book, even though it still refers to the same real life object and is still being referred to and used as a book.

[5] The uniqueness rule in ORM is equivalent to 0:1 cardinality restriction. (notation: '⟷'), it can be verbalized as "each book must have at most one ISBN".

[6] The mandatory rule in ORM is equivalent to 1-m cardinality restriction. (notation: '●'), it can be verbalized as "each book must have at least one ISBN".

Accordingly, as ontology rules are supposed to formally specify/constrain the permitted models[7] that can necessarily hold for a given domain, determining such rules, in practice is dominated by "what is permitted and what is not" for the *intended* or *expected* usability.
Furthermore, besides the modeling decisions of ontology rules, the determination of the number and the type of these rules (the reasoning scenario) are also influenced by usability perspectives. For example, a light-weight axiomatization (e.g. with a minimum number of rules or formalities) might be sufficient if the ontology is to be accessed and used by people (i.e. not computers). Depending on the application scenario other *types* of ontology rules (i.e. modeling primitives/constructs) might be preferred over the ORM set of rules (which are easier to reason for database and XML based applications).
At this point, we conclude that even application-types might intuitively agree on the same semantics at the domain level, but the usability influence on axiomatizing this semantics may lead to different (or even conflicting) axiomatizations. An axiomatization might be more relevant for some applications than others, due to the difference of their usability perspectives. This issue presents an important challenge to the nature and the foundation of ontology engineering.

Related work

A general overview on ontology engineering methodologies is provided by [GFC04: pp.113-153], including short descriptions of the methods. A very recent list is given by [PT06], who do not provide details of the various methods but rather shortly situate them. It is nevertheless a very good starting point for further bibliographic research.

Guarino et al. have argued (in e.g. [G98a]) that in order to capture knowledge at the domain level, the notion of what is an ontology should be more precisely defined. Gruber's commonly used but somewhat schematic definition [G95] of an ontology is "an explicit specification of a conceptualization", and refers to an *extensional* ("Tarski-like") notion of a conceptualization as found e.g. in [GN87]. Guarino et al. point out that this definition *per se* does not adequately fit the purposes of an ontology. They argue, correctly in our opinion, that a conceptualization *should not be extensional b*ecause a conceptualization benefits from invariance under changes that occur at the instance level and from transitions between different "states of affairs"[8] in a domain. They propose instead a conceptualization as *an intensional* semantic structure i.e. abstracting from the instance level, which encodes *implicit rules* constraining the structure of a piece of reality. Therefore in their words, "an ontology only indirectly accounts for a conceptualization". Put differently, an ontology becomes a logical theory which possesses a conceptualization as an explicit, partial model. The resulting OntoClean methodology for evaluating ontological decisions [GW02] consists of a set of formal notions that are drawn from Analytical Philosophy and called *meta-properties*. Such meta-properties include *rigidity*, *essence*, *identity*,

[7] Also called "ontology models".
[8]A state of affairs refers to a particular instance of reality, or also called *a* possible world.

unity, and *dependence*. The idea of these notions is to focus on the *intrinsic properties* of concepts, assumed to be *application-independent.*

Clearly in the previous example the two axiomatizations should therefore not be seen as different ontologies since they only differ on their description of extensions i.e. states of affairs. Both axiomatizations implicitly share the same intensional semantic structure or conceptualization. Furthermore, the ISBN is an *extrinsic* property since it is not rigid[9] for *all* instances of the concept "book". Therefore, it cannot be used to specify the intended meaning of a book at the domain level.

An important problem of the OntoClean methodology, in our opinion, is its applicability. First of all it relies on deep philosophical notions that in practice are not easy or intuitive to teach and utilize –at least for "nonintellectual" domain experts; and secondly it only focuses on the intrinsic properties of concepts and such properties are often difficult to articulate. For example, how to formally and explicitly articulate the identity criteria of a book (or person, brain, table, conference, love, etc.)? Guarino and Welty state in [WG01]: "*We may claim as part of our analysis that people are uniquely identified by their brain, but this information would not appear in the final system we are designing*". In short, it would seem that OntoClean can be applied mainly by highly trained intellectuals for domain analysis and ontological checks[10].

Ontology usability is also important. This is another factor that should not be ignored, especially with regards to the philosophically inspired research on ontologies (or the so-called "philosophical ontology" as in [S03a]). In keeping with current views in the field of information technology, ontologies are to be shared and used collaboratively in software applications. This gives even more weight to the importance of *ontology usability*.

Conclusions

The closer an axiomatization is to certain application perspectives, the more usable it will be. In contrast, the more an axiomatization is independent of application perspectives, the more reusable it will be. In other words, *there is a tradeoff between ontology usability and ontology reusability*.

From a *methodological viewpoint,* if a methodology emphasizes usability perspectives or evaluates ontologies based on how they fulfill specific application requirements, the resulting ontology will be similar to a conceptual data schema (or a classical knowledge base) containing application specific and thus, less reusable knowledge. Likewise, if a methodology emphasizes the independency of the knowledge, the resulting ontology in general will be less *usable*, since it has no intended use by ignoring application perspectives.

Based on the above, we conclude the following ontology engineering requirement:

> *The influence of usability perspectives on ontology axioms should be well articulated, pursuing both reusability and usability.*

[9] "A property is rigid if it is essential to all its possible instances; an instance of a rigid property cannot stop being an instance of that property in a different world" [WG01].

[10] See [GGO02] for a successful application of OntoClean on cleaning up WordNet.

3 The DOGMA approach

In this section, we present the DOGMA approach for ontology engineering, which aims to tackle the engineering challenges stated in the previous section. In section 3.1 we schematically illustrate the DOGMA approach. The DOGMA philosophy is presented in section 3.2. Section 3.3 presents the formalization of the notion of Ontology Base for representing domain axiomatization, and section 3.4 presents how applications axiomatizations can built and used.

3.1 Overview

As we mentioned before, in DOGMA, we introduce the notion of *ontology base* for capturing domain axiomatizations; and we introduce the notion of application axiomatization, by which particular applications commit to an ontology base, i.e. a *domain axiomatization and its application axiomatizations*. see figure 3.

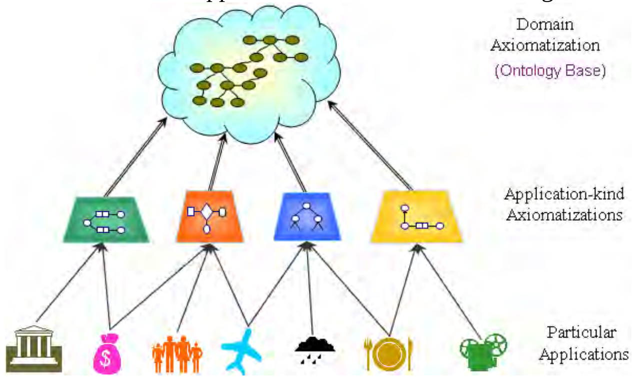


Fig. 3 DOGMA framework.

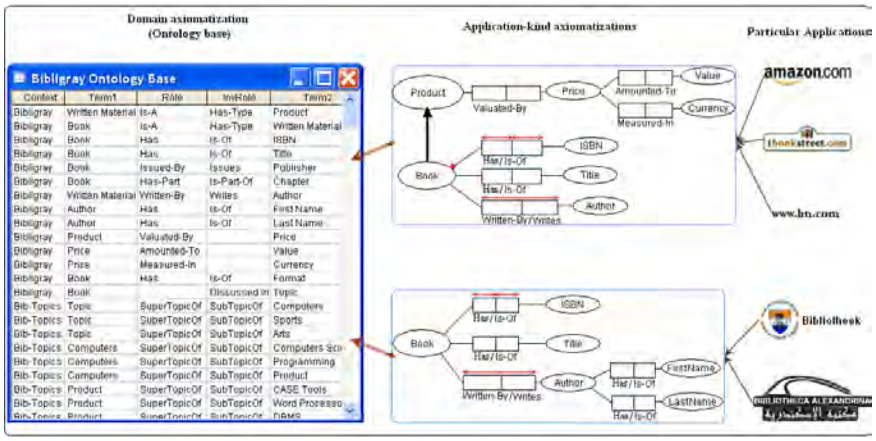
<u>*The ontology base*</u> is intended to capture "plausible" domain axiomatizations. It basically consists of a set of binary conceptual relations. The lexical rendering of a binary conceptual relation is called *lexon*. A lexon is described as a tuple of the form <γ: Term1, Role, InvRole, Term2>, where Term1 and Term2 are linguistic terms. γ is a context identifier, used to *bound* the interpretation of a linguistic term: notably, for each context γ and term T, the pair (γ, T) is assumed to refer to a uniquely identifiable *concept*. Role and InvRole are lexicalizations of the paired roles in any binary relationship, for example WorkingFor/Employing, or HasType/IsTypeOf.

Particular applications now may *commit* to the ontology base through an application axiomatization. Such a commitment is called application's *ontological commitment*[11]. Each application axiomatization consists of (1) a selected set of lexons from an ontology base; (2) a specified set of rules to constrain the usability of these lexons.

[11] We sometimes use the notion of application's "ontological commitment" and the notion "application axiomatization" interchangbly in this chapter. It is also worth to note that the notion of "ontological commitment" as found in [GG95] generally refers to a "conceptualization", literally, it is defined as "a partial semantic account of the intended conceptualization of a logical theory."

Example

Resuming the Bibliography example above, the table at the left of Figure 5 shows a Bibliography ontology base. The (ORM) graphs at the right show two application axiomatizations (Bookstore and Library axiomatizations) by which particular applications might make a commitment to and share this same Bibliography ontology base. *Notice that all conceptual relations in both application axiomatizations correspond to (or are derived from) lexons in the Bibliography ontology base. In this way, different application axiomatizations share and reuse the same intended meaning of domain concepts.* As we shall show later, application axiomatizations can be represented not only in ORM (as shown in figure 5), but in any specification language such as OWL, EML, EEU, etc.



Context	Term1	Role	InvRole	Term2
Bibilgray	Written Material	Is-A	Has-Type	Product
Bibilgray	Book	Is-A	Has-Type	Written Material
Bibilgray	Book	Has	Is-Of	ISBN
Bibilgray	Book	Has	Is-Of	Title
Bibilgray	Book	Issued-By	Issues	Publisher
Bibilgray	Book	Has-Part	Is-Part-Of	Chapter
Bibilgray	Written Material	Written-By	Writes	Author
Bibilgray	Author	Has	Is-Of	First Name
Bibilgray	Author	Has	Is-Of	Last Name
Bibilgray	Product	Valuated-By		Price
Bibilgray	Price	Amounted-To		Value
Bibilgray	Price	Measured-In		Currency
Bibilgray	Book	Has	Is-Of	Format
Bibilgray	Book		Discussed In	Topic
Bib-Topics	Topic	SuperTopicOf	SubTopicOf	Computers
Bib-Topics	Topic	SuperTopicOf	SubTopicOf	Sports
Bib-Topics	Topic	SuperTopicOf	SubTopicOf	Arts
Bib-Topics	Computers	SuperTopicOf	SubTopicOf	Computers Sci
Bib-Topics	Computers	SuperTopicOf	SubTopicOf	Programming
Bib-Topics	Computers	SuperTopicOf	SubTopicOf	Product
Bib-Topics	Product	SuperTopicOf	SubTopicOf	CASE Tools
Bib-Topics	Product	SuperTopicOf	SubTopicOf	Word Processo
Bib-Topics	Product	SuperTopicOf	SubTopicOf	DBMS

Fig. 5. Particular applications commit to a DOGMA ontology base through their respective application axiomatizations.

3.2 The DOGMA philosophy (Application vs. Domain Axiomatization)

This section presents the fundamental idea of DOGMA. We introduce the notion of domain and application axiomatizations, and the formal relationship between them (called *double-articulation*). We discuss and formalize which type of knowledge should be captured in a domain verses application axiomatization. The translation of this philosophy into a software implementation is presented in section 3.3 and 3.4.

As we have discussed in section 2, decreasing the influence of usability perspectives is a principal engineering requirement when axiomatizing *domain concepts*. To capture knowledge at the domain level, one should focus on characterizing the *intended meaning* of domain vocabularies (i.e. domain concepts), rather than on how and why these concepts will be used. A domain axiomatization becomes an axiomatic theory that captures the axioms that account for (i.e. characterizes) the intended meaning of the domain vocabularies.

This motivates us to understand the relationship between a domain vocabulary and the specification of its intended meaning in a logical theory.

In general, it is not possible to build a logical theory to specify *the complete and exact intended meaning* of a domain vocabulary[12]. Usually, the level of detail that is appropriate to explicitly capture and represent it is subject to what is reasonable and plausible for domain applications. Other details will have to remain implicit assumptions. These assumptions are usually denoted in linguistic terms that we use to lexicalize concepts, and this implicit character follows from our interpretation of these linguistic terms.

Incidentally, the study of the relationship between concepts and their linguistic terms is an ancient one; for example Avicenna (980-1037) [Q91] already argued that *"There is a strong relationship/dependence between concepts and their linguistic terms, change on linguistic aspects may affect the intended meaning… Therefore logicians should consider linguistic aspects 'as they are'. …"*[13].

Indeed, the linguistic terms that we usually use to name symbols in a logical theory convey some important assumptions, which are part of the conceptualization that underlie the logical theory. We believe that these assumptions should not be excluded or ignored (at least by definition) as indeed *they implicitly are part of our conceptualization*.

Hence, we share Guarino and Giaretta's viewpoint [GG95], that an ontology (as explicit domain axiomatization) only *approximates* its underlying conceptualization; and that a domain axiomatization should be *interpreted intensionally*, referring to the intensional notion of a conceptualization. They point out that Gruber's [G95] earlier mentioned definition does not adequately fit the purposes of an ontology since a mere re-arrangement of domain objects (i.e. different state of affairs) would correspond to *different* conceptualizations. Guarino and Giaretta argue that a conceptualization benefits from invariance under changes that occur at the instance level by transitions between merely different "states of affairs" in a domain, and thus *should not be extensional*. Instead, they propose a conceptualization as *an intensional semantic structure* (i.e. abstracting from the instance level), *which encodes implicit rules* constraining the structure of a piece of reality. Indeed, this definition allows for the focus on the meaning of domain vocabularies (by capturing their intuitions) independently of a state of affairs. See [G98a] for the details and formalisms.

3.2.1 Definition (double articulation, intended models, legal models)

Given a concept **C** as *a set of rules (i.e. axioms) in our mind about a certain thing in reality*, the set **I** of "all possible" instances that comply with these rules are called the *intended models* of the concept **C**. Such concepts are captured at the domain axiomatization level. An application A_i that is interested in a subset $\mathbf{I}_{Ai}$ of the set **I** **(**according to its usability perspectives**)**, is supposed to provide some rules to specialize **I**. In other words, every instance in $\mathbf{I}_{Ai}$ must also be an instance in **I**:

$$\mathbf{I}_{Ai} \subseteq \mathbf{I}$$

We call the subset $\mathbf{I}_{Ai}$: the *legal models* (or extensions) of the application's concept $\mathbf{C}_{Ai}$. Such application rules are captured at the application axiomatization level. Both

[12] This is because of the large number of axioms and details that need to be intensively captured and investigated, such detailed axiomatizations are difficult -for both humans and machines- to compute and reason on, and might holds "trivial" assumptions.

[13] This is an approximated translation from Arabic to English.

domain and application axiomatizations can be seen (or expressed) for example as sentences in first order logic.
We call the relationship (defined above) between a domain axiomatization and an application axiomatization: *double articulation*[14].
As we have illustrated in the previous section, bookstore applications that are interested *only* in the instances of the concept 'book' (that can be sold) need to declare the Mandatory rule that each instance of book must have an ISBN value.
In Figure 6 we show three kinds of applications specializing a domain concept.

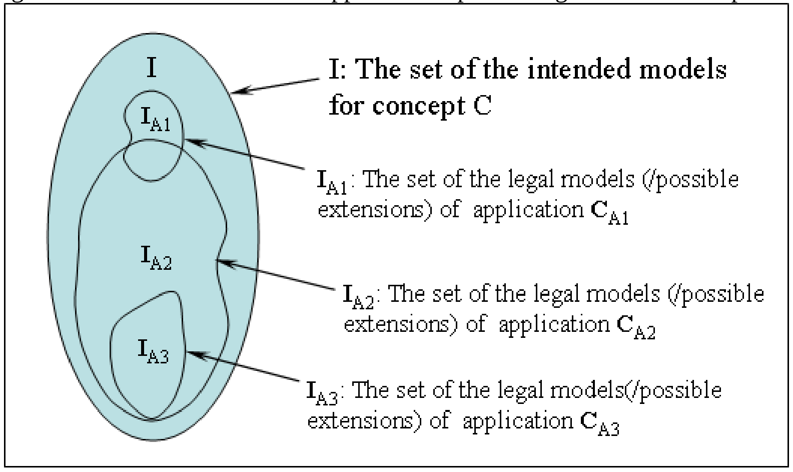

I
I_{A1}
I_{A2}
I_{A3}
I: The set of the intended models for concept C
I_{A1}: The set of the legal models (/possible extensions) of application C_{A1}
I_{A2}: The set of the legal models (/possible extensions) of application C_{A2}
I_{A3}: The set of the legal models(/possible extensions) of application C_{A3}

Fig. 6 An example of three different applications specializing a domain concept.

The differences between the legal models of these application-types illustrate their different usability perspectives:

- The intersection between the legal models of $\mathbf{C}_{A2}$ and the legal models $\mathbf{C}_{A3}$ shows that I_{A3} is a subset of I_{A2}. An example of this case could be the difference between notions of 'book' in the axiomatization of bookstores and libraries: all legal instances of the bookstores' notion are legal instances for the libraries, but not vice versa. For libraries, the instances of e.g. 'Manual' or 'Master Thesis' can be instances of a 'book'; however, they cannot be instances of 'book' for bookstores, unless they are published with an 'ISBN'.
- The difference between $\mathbf{I}_{A1}$ and $\mathbf{I}_{A3}$ shows an extreme case: two types of applications sharing the same concept $\mathbf{C}$ while their legal models are completely disjoint according to their usability perspectives. An example of this case could be the difference between notions of 'book' in the axiomatization of bookstores' and museums': Museums are interested in exhibiting and exchanging instances of old

[14] The term "double articulation" in this chapter simply means *expressing knowledge in a twofold axiomatization*. The term "articulation" in WordNet means: "Expressing in coherent verbal form", "The shape or manner in which things come together and a connection is made", etc. In the semiotics and linguistics literature, the term "double articulation" has been introduced by [N90][M55] (which has a different meaning and usage than ours) to refer to the distinction between lexical and functional unites of language or between content and expression.

‘books’, while bookstores are not interested in such ‘books’, unless for example, they are re-edited and published in a modern style.

One may wonder how *domain concepts* can be agreed upon because of the difficulty in gaining an objective insight into the nuances of another person’s thoughts. Many researchers admit that a conceptualization reflects a particular viewpoint and that it is entirely possible that every person has his/her “own” (linguistic manner of referring to) concepts. For example, Bench-Capon and Malcolm argued in [BM99] that conceptualizations are likely to be influenced by personal tastes and may reflect fundamental disagreements. In our opinion, herein lies the importance of linguistic terms.

Linguistic resources (such as lexicons, dictionaries, and glossaries) can be used as consensus references to *root* ontology concepts [J06] [M99a]. In other words, ontology concepts and axioms can be investigated using such linguistic resources and it can be determined whether a concept is influenced by usability perspectives. We explain this idea further in the following paragraphs.

The importance of using linguistic resources in this way lies in the fact that a linguistic resource renders/contains the intended meaning of a linguistic term as it is commonly "agreed" among the community of its language. The set of concepts that a language lexicalizes through its set of word-forms is generally an agreed conceptualization[15] [T00]. For example, when we use the English word 'book', we actually refer to the set of implicit rules that are common to English-speaking people for distinguishing 'books' from other objects. Such implicit rules (i.e. concepts) are learned and agreed from the repeated use of word-forms and their referents. Usually, lexicographers and lexicon developers investigate the repeated use of a word-form (e.g. based on a comprehensive corpus) to determine its underlying concept(s). See [J06] for more details about the incorporation of linguistic resources in DOGMA. Other researchers have also used linguistic resources in ontology engineering in other ways [BSZS06] [PS06].

3.2.2 On representing domain axiomatizations

In this section, we discuss some choices that are relevant for *representing* domain axiomatizations.

A domain axiomatization merely cannot be a list of linguistic terms, and their intended meanings cannot be completely implicit. The intended meaning of linguistic terms should be axiomatized and represented by means of a formal language.

From a methodological viewpoint, such a formal language should be content-oriented rather than syntax-oriented. This language should serve as a theoretical tool which guides ontology builders through its primitives, and restrict them to focus *only* on and represent the “kinds” of axioms that account for the intended meaning of domain vocabularies.

By analogy, the conceptual “data” modeling languages ORM and EER provide database designers a set of primitives with which they can be guided to build a normalized database schema. Indeed, ORM and EER can be seen as content-oriented languages, because they *restrict* the focus of database designers to the *integrity of data models*.

[15] Thus, we may view a lexicon of a language as an informal ontology for its community.

An example of the difference between conceptual data modeling primitives and the kind of primitives that account for the intended meaning of a vocabulary[16] is the difference between the "Rigid" and "Mandatory". Something can be mandatory but not rigid, as in the case of 'ISBN' which is not a rigid property for every instance of a 'book' but could be mandatory for some applications. In other words, to model something as a rigid property, it should be rigid in *all possible* applications, while what can be mandatory for an application might not be mandatory for another. See [JDM03][GHW02] for more discussions on such issues.

Current research trends on ontology languages within the Semantic Web and the description logic communities are mainly concerned with improving *logical consistency* and inference services. Such services in our opinion are more suitable for building knowledge base applications or expert systems rather than axiomatizing "domain concepts". Significant results within the description logic community have indeed been achieved in the development of expressive and decidable logics, such as DLR [CGDL01], SHOIN [HST99], etc., yet less attention has been given to the quality of ontological *content*.

"…I was annoyed by the fact that knowledge representation research was more and more focusing on reasoning issues, while the core problems of getting the right representations were not receiving that much attention…". (Nicola Guarino[17]*).*

An example of a modeling primitive in the SHOQ description logic which in our opinion, should not be allowed in axiomatizing domain concepts since it does not account for meaning, is *datatypes* [P04]. Such a primitive belongs mainly to the symbolic level. In short, description logics (and their derivative languages such as DAML+OIL, or OWL) seem to play a useful role in specifying application (rather than domain) axiomatizations.

We shall return, in section 3.4 to the use of both conceptual data modeling languages and description logic based languages, for modeling and representing application axiomatizations.

We observe two possible ways to capture formal domain axiomatizations: (1) as an arbitrary set of axioms, e.g. using description logic, or (2) through a knowledge representation model (e.g. a database). The first case is common within the Semantic Web and Artificial Intelligence communities; in this case ontology builders are responsible (i.e. unguided) to decide whether an axiom accounts for the intended meaning of a vocabulary. This way offers ontology builders more freedom and expressiveness, but the risk of encoding usability perspectives is still high. In the second case, ontology builders are restricted only to capturing and storing the kind of axioms that account for factual meaning; assuming that the representation model is well studied and designed to pursue such axioms. This way is less expressive than the first one, but it reduces the risk of mixing domain and application axioms. The second way offers scalability in accessing and retrieving axioms, which is usually a problematic issue in the first way. The second way is mostly used within the lexical semantics community, e.g. WordNet [MBFGM90], Termintography [KTT03]. Notice

[16] i.e. conceptual data modeling vs. conceptual domain modeling.

[17] An interview with Nicola Guarino and Christopher Welty (9 June 2004): http://esi-topics.com/erf/2004/june04-ChristopherWelty.html

that both ways are (or should be) well formalized and map-able to first order logic, and thus can be seen as logical theories.
We have chosen the second way for our approach. As we will show in section 3.3, we have developed a data model for capturing domain axiomatizations called an *ontology base*.

3.2.3 Summary: properties of domain axiomatization

In this section, we summarize the basic properties of a domain axiomatization: it is (1) an axiomatized theory (2) that accounts for the intended meaning of domain vocabularies; (3) it is intended to be shared and used as a vocabulary space for application axiomatizations. It is supposed to be (4) interpreted intensionally, (5) and investigated and rooted at a human language conceptualization.

3.3 The notion of an (Ontology Base), for capturing domain axiomatizations

An ontology base is a knowledge representation model for capturing domain axiomatizations. This notion is used as a core component in the DOGMA approach.
Basically, an ontology base consists of a set of lexons. A lexon is a plausible binary relationship between context-specific linguistic terms, or in other words, a lexical rendering of a –plausible– binary conceptual relation.

3.3.1 Definition (Lexon)

A lexon is a 5-tuple of the form:

$$< \gamma : \mathrm{T}_1, r, r', \mathrm{T}_2 >$$

Where:

γ is a context identifier (the notion of context will be defined shortly).

T_1 and T_2 are linguistic terms from a language L.
r and *r'* are lexicalizations of the paired roles in a binary conceptual relationship R; the role *r'* is the inverse of the role *r*. One can verbalize a lexon as (T_1 *r* T_2), and (T_2 *r'* T_1). For example, the role pair of a *subsumption* relationship could be: “Is_type_of” and “Has_type”; the role pair of a *parthood* relationship could be: “Is_part_of” and “Has_part”, and so forth.

The following is a set of lexons, as a simple example of part of an ontology base:

```
<Commerce: Person, Issues, Issued by, Order>
<Commerce: Order, Settled Via, Settles, Payment Method>
<Commerce: Money Order, Is a type of, Has type, Payment Method>
<Commerce: Check, Is a type of, Has type, Payment Method>
<Commerce: Payment Card, Is a type of, Has type, Payment Method>
<Commerce: Credit Card, Is a type of, Has type, Payment Card>
<Commerce: Credit Card, Has, Is of, Expiration Date>
```

3.3.2 Definition (Concept)

A term T within a context γ is assumed to refer to uniquely identified *concept* C:

$$(\gamma, T) \rightarrow C$$

Notice, for example, that within the context 'Commerce', the linguistic term 'Order' refers to "A commercial document used to request someone to supply something in return for payment". It may refer to other concepts within other contexts, e.g. within the context 'Military', the term 'Order' refers to "A command given by a superior that must be obeyed"[18]. Further detail about the notion of context will be discussed in the next section.

As we have discussed earlier, *a concept is circumscribed by a set of rules in our mind about a certain thing in reality*. The notion of *intended meaning* (or word meaning/sense) can be used alternatively with the notion of concept to denote something. The set of all possible instances (i.e. in all possible stats of affairs) that comply with these rules are called *intended models*.

As part of the context for each concept in DOGMA, there must be a *gloss*. A gloss is an auxiliary informal account for the commonsense perception of humans of the intended meaning of a linguistic term. The purpose of a gloss is *not* to provide or catalogue general information and comments about a concept, as conventional dictionaries and encyclopedias do [MBFGM90]. A gloss, for formal ontology engineering purposes, is supposed to render factual knowledge that is critical to understanding a concept, but that is unreasonable or very difficult to formalize and/or articulate explicitly. Although a gloss is not intended to be processed by machines, but its content is controlled by a set of well-defined guidelines (see [J06] for more details), such as: It should start with the *principal/super type* of the concept being defined; It should be written in the form of propositions; it should focus on distinguishing characteristics and intrinsic properties that differentiate the concept from other concepts; It should be consistent with the lexons and formal definitions, etc.

3.3.3 Definition (Role)

A *role* is an axiomatic entity lexically expressing how a concept (referred by a term within a context) relates to another concept. A lexon being a binary relationship always involves two roles.

A role within a context is not intended to refer to a concept; thus, $(\gamma, r) \rightarrow C$ is improper. In other words, our notion of role does not refer to a "stand alone" unary (or binary) concept. Rather, roles only lexicalize the participation of a "unary concept" in an n-ary conceptual relationship. As the notion of a lexon is a lexical rendering of a binary conceptual relationship, we formalize a lexon as two context-specific terms playing mutual roles, that both refers to a *binary concept (typically called binary conceptual relation)*:

$$< (\gamma, T, r), (\gamma, T, r) > \quad \rightarrow \quad C^2$$

The notation of a context-specific term playing a role $\left(\gamma, T, r\right)$ is called *concept-role*.

For practical purposes, we shall not require for both roles to be explicitly lexicalized within a lexon. We assume that at least one role is to be lexicalized, represented as <Bibliography, Book, is-a, , Written Material>.

[18] These two definitions of the term "Order" are taken from WordNet, (May 2004) http://www.cogsci.princeton.edu/cgi-bin/webwn.

A DOGMA ontology base contains only binary relationships. This does not deny the existence of ternary (or *n*-ary) relationships. Ternary resp. *n*-ary relationships may always be converted into an equivalent set of 3, resp. *n*, binary relationships, possibly with the introduction of new terms/concepts. In practice relationships are however mainly binary.

3.3.4 Definition (Mapping lexons into first order logic)

With each lexon $<\gamma\!: \mathrm{T}_1, r, r', \mathrm{T}_2>$ in the ontology base there correspond three statements in first order logic, as follows:

$$\forall x\, T_1(x) \rightarrow (\forall y\, r(x,y) \rightarrow T_2(y))$$
$$\forall y\, T_2(y) \rightarrow (\forall x\, r'(y,x) \rightarrow T_1(x))$$
$$\forall x, y\ r(x,y) \leftrightarrow r'(y,x)$$

For example, the mapping of the lexon <Commerce: Person, Issues, IssuedBy, Order> into first order logic produces:

$$\forall x\ Person(x) \rightarrow (\forall y\ Issues(x,y) \rightarrow Order(y))$$
$$\forall y\ Order(y) \rightarrow (\forall x\ IssuedBy(y,x) \rightarrow Person(x))$$
$$\forall x, y\ \ Issues(x,y) \leftrightarrow IssuedBy(y,x)$$

Notice that Context is not part of our formal mapping of lexons. As we shall discuss in the next section, a context for our purposes here is a mostly *informal* notion used to link unambiguously (i.e. bound) the interpretation of a linguistic term to a concept. Linguistic terms, e.g. ‘Person’, ‘Order’, etc. can be seen as unambiguous terms (i.e. concepts) within the lexon formal mapping. A lexon (or its formal mapping) is assumed to be plausible (i.e. to be a *weak axiom*) within its context, see section 3.3.5. In section 3.3.6 we shall discuss how to introduce further formal axiomatizations at the ontology base level, for targeting systematic ontological quality.
Finally, our formal lexon mapping assumes unique role names. Each role label (or InvRole) should be unique within the formal mapping of lexons. As this is might not be the case in practice, one can provide an “internal” naming convention, for example, by renaming ‘Issues’ as ‘Issues_Order’ and ‘IssuedBy’ as ‘IssuedBy_Person’.

At this point, we have established how lexons are the basic building blocks of an ontology base and that they express basic domain facts. *The principal role of an ontology base is to be a shared vocabulary space for application axiomatizations.* As sharing lexons means sharing the same concepts and their intended models, semantic interoperability between classes of autonomous applications can be achieved, basically, by sharing a certain set of lexons[19] and agreeing on their interpretation (*commitments*, see below in 3.4).

[19] As we shall show in section 3.4, a class of interoperating applications may need to agree on and share some rules that constrain the use of a concept, i.e. share the same legal models.

3.3.5 The notion of context

The notion of context has been, and still is, the subject of occasionally intense study, notably in the field of Artificial Intelligence. It has received different interpretations. Commonly, the notion of context has been realized as a set of formal axioms (i.e. a theory) about concepts. It has been used among other things: to localize or encode a particular party's view of a domain, cf. C-OWL [BHGSS03]; as a background, micro-theory, or higher-order theory for the interpretation of certain states of affairs [M93]; and to facilitate the translation of facts from one context to another, as in KIF [PFP+92].

In our approach, we shall use the notion of context to play a "*scoping" role* at the ontology base level. We say a term *within* a context refers to a concept, or in other words, that *context is an abstract identifier that refers to implicit (or maybe tacit*[20]*) assumptions, in which the interpretation of a term is bounded to a concept.*

Notice that a context in our approach is not *explicit* formal knowledge. In practice, we define context by referring to a source (e.g. a set of documents, laws and regulations, informal description of "best practice", etc.), which, by *human understanding*, is assumed to "contain" those assumptions. Lexons are assumed (by that same human understanding) to be "plausible within their context's source". Hence, a lexon is seen as a (weak) domain *axiom*.

Note. For ease of readability, we will in lexons continue to use a (unique) mnemonic label such as Commerce or Bibliography to denote a context rather than an abstract context identifier (pointer to a resource) in the representations of lexons below. To wit, Bibliography will point to a document in which a human interpreter, possibly assisted by programs, finds sufficient "context" to at least disambiguate both terms used in the lexon.

3.3.6 Further formal axiomatizations (incorporating upper level ontologies)

In order to achieve *a systematic ontological quality and precision*[21] on the specification of the intended meanings of linguistic terms, these specifications might need to receive more formal restrictions, than just mapping lexons into logical statements.

For example, without introducing further formal restrictions to the following lexons:

```
<Bibliography: Man, Is-a, Person>
<Bibliography: Author, Is-a, Person>
<Bibliography: John, Is-a, Person>
```

[20] The difference between implicit and tacit assumptions, is that the implicit assumptions can, in principle, be articulated but still they have not, while tacit assumptions are the knowledge that cannot be articulated. it consists partially of technical skills -the kind of informal, hard-to-pin-down skills captured in terms like "know-how", and "we know more than we can tell or put in words". However, even though tacit assumptions cannot be articulated, they can be transferred through other means over than verbal or formal descriptions [Inn+03] [N94].

[21] The notion of "ontological precision" is defined by Aldo Gangemi in [G04] as "*the ability to catch all and only the intended meaning*".

The ontological difference (or rather, the misuse of 'is-a') cannot be systematically detected[22]. In this section, we discuss how a formal axiomatic system can be introduced into an ontology base.
As we have chosen to represent formal domain axiomatization in a data model (i.e. ontology base), arbitrary and expressive formal definitions are restricted (see our discussion on this issue in section 3.2.3). Therefore, we extend the ontology base model to incorporate primitives of upper level ontologies. Our incorporation of upper level ontologies in this chapter is fairly simplistic; the deeper philosophical arguments that are necessary for such incorporation are presented schematically for the sake of completeness only. It is important to note that upper ontologies are still very much exponents of work in progress. Upper level ontologies are formal axiomatic systems that describe the most general categories of reality. Such ontologies are not only application and task independent but represent also domain (and possibly language) independent axiomatizations [DHHS01] [G98b].
Based on the literature of upper level ontologies as found for example in [DHHS01] [G98b], we introduce in our approach the notion of *upper-form.* Each term within a context should have an upper-form, likewise, each lexon should have an upper-form.

Term upper-forms
Term upper-forms are superior types of concepts, such as substantial, feature, abstract, region, event, process, type, role, property, particular, etc. The notation of a term upper-form declaration is:

$$\gamma(T) : < UpperFormName >$$

Examples:
Bibliography(Person):Substantial,
Bibliography(Author):Substantial,
Bibliography(First-Name):Property

A term can have several upper-forms, denoted $\gamma(T) : \{UpperForm, ...\}$. Examples:
Bibliography(Person):{Substantial, Type},
Bibliography(Author):{Substantial, Role},
Bibliography(John):{Substantial, Instance}.

Lexon upper-forms
Lexon upper-forms are relationship kinds, also called "basic primitive relations", such as parthood, dependence, property-of, attribution, subsumption, etc. Such relationship kinds are carefully and formally axiomatized in upper level ontologies, and they are general enough to be applied in multiple domains. Our notation of a lexon upper-form declaration is

$$< \gamma : \mathrm{T}_1, r, r', \mathrm{T}_2 >: < UpperFormName >$$

For instance, the lexon "<Bibliography: Book, Is-a, HasType, Written Material>: Subsumption" is declared as a subsumption relationship where the concept 'Book' formally

[22] By assuming that the 'is-a' refers to a subsumption relationship (i.e. Sub-Type of), only the first lexon is correct. The 'is-a' in the second lexon should interpreted as "is role of", because 'Author' is a role of 'Person' and not a type of a 'Person'; and obviously, the last lexon refers to 'is instance of'. See [GW02] for more details on this issue.

subsumes the concept 'Written Material'. Similarly the declaration "<Bibliography: Book, Has-Part, Is-Part-Of, Chapter>: Parthood" now states that the lexon lexically expresses a particular *parthood* (meronymy) relationship, where within the context of Bibliography, an instance of the concept 'chapter' is a part of an instance of the concept 'Book'. And "<Bibliography: Author, Has, Is-Of, Name>: Property" declares the lexon as a *property-of* relationship, where the concept 'Name' is a property of the concept 'Author', and so forth.
Upper-forms may carry with them a formal axiomatization and reasoning mechanism, as defined in an associated *upper level ontology*,.They may thus be used to add a *formal account* to lexons. For example, a formal account of the lexon "<Bibliography, John, instance-of, Author>: Instantiation" may include or be induced from a (partial) formal axiomatization of the instantiation relationship such as found in [GGMO01]:

$$\text{Instantiation}(x,y) \rightarrow \neg\text{Instantiation}(y,x)$$
$$(\text{Instantiation}(x,y) \wedge \text{Instantiation}(x,z)) \rightarrow (\neg\text{Instantiation}(y,z) \wedge \neg\text{Instantiation}(y,z))$$
$$\text{Particular}(x) =_{def} \neg\exists y\ (\text{Instantiation}(y,x))$$
$$\text{Universal}(x) =_{def} \neg\text{Particular}(x)$$

Fig. 7 A formal axiomatization of Instantiation,, from [GGMO01].

Similarly Parthood, Subsumption etc. may become interpreted from their formal axiomatizations in [GGMO01]. For instance, under the formal axiomatization of the 'Subsumption' relationship in [GGMO01], the following lexon would be inadmissible because its declared upper-form conflicts with the adopted axiomatization of the term upper-forms, where Role may no longer subsume Type:

Bibliography(Person):{Substantial, Type}
Bibliography(Author):{Substantial, Role}
<Bibliography: Author, Is-a, , Person>: Subsumption

Notice that formal axiomatizations of such upper forms are not necessarily to be used at runtime by applications that use or share lexons. Their main purpose could be as theoretical tools to help achieve verifiable quality during development and maintenance time of an ontology.
Note that our methodological principles and their implementation prototypes are not dependent on a particular choice of upper level ontology. This is left to ontology builders. However, libraries of upper-ontology plug-ins may be envisaged complete with predefined reasoning, for ontology builders to apply on selected sets of lexons as part of so-called *commitments*. This constitutes the foundational principle of *application axiomatization* in DOGMA as explained in the following section.

3.4 Application axiomatization

The notion of an ontology base was introduced in order to capture domain axiomatizations independently of usability perspectives. In this section, we introduce the second part of the double articulation: *application axiomatizations*. First, we discuss the general properties of these axiomatizations; then, we introduce the key notion of an *application's ontological commitment*.

While the axiomatization of domain knowledge is mainly concerned with the characterization of the "intended models" of concepts, the axiomatization of application knowledge is mainly concerned with the characterization of the "legal models" of these concepts (see Fig. 6). Typically, as domain axiomatizations are intended to be shared, public, and highly reusable at the domain level, application axiomatizations are intended to be local and highly usable at the task/application-kind level.

As we have discussed earlier, applications that are interested only in a subset of the intended models of a concept (according to their usability perspective) are supposed to provide some rules to specialize these intended models. Such a specialization is called an *application axiomatization*. The vocabulary of concepts used in application axiomatization is restricted to the vocabulary defined in its domain axiomatization. (Note that this "specialization" should therefore not be confused with the subsumption relationship discussed earlier.) As clarified below, an application axiomatization comes defined as a set of rules that constrain the use of the domain vocabulary. More specifically, these rules declare what must necessarily hold in any possible interpretation for a given class of applications.

A particular application is said to *commit ontologically* to an intended meaning of a domain vocabulary (stored as lexons in an ontology base) through its application axiomatization., i.e. the latter may be considered a formal specification of such a commitment.

An application axiomatization typically consists of: (1) an *ontological view* that specifies which domain concepts in an ontology base are relevant to include in this axiomatization. These concepts can be explicit lexons or derived from lexons, (2) a set of rules that defines the legal models of the ontological view in the classical model-theoretic semantics sense, i.e. it formally specifies what must or must not hold in any possible world (interpretation) for the applications sharing this axiomatization.

We say that a particular extension of an application (i.e. a set of instances) commits to an ontology base through an application axiomatization if it conforms to or is consistent with the ontological view and the rules declared in this axiomatization (cf. model-theoretic semantics).

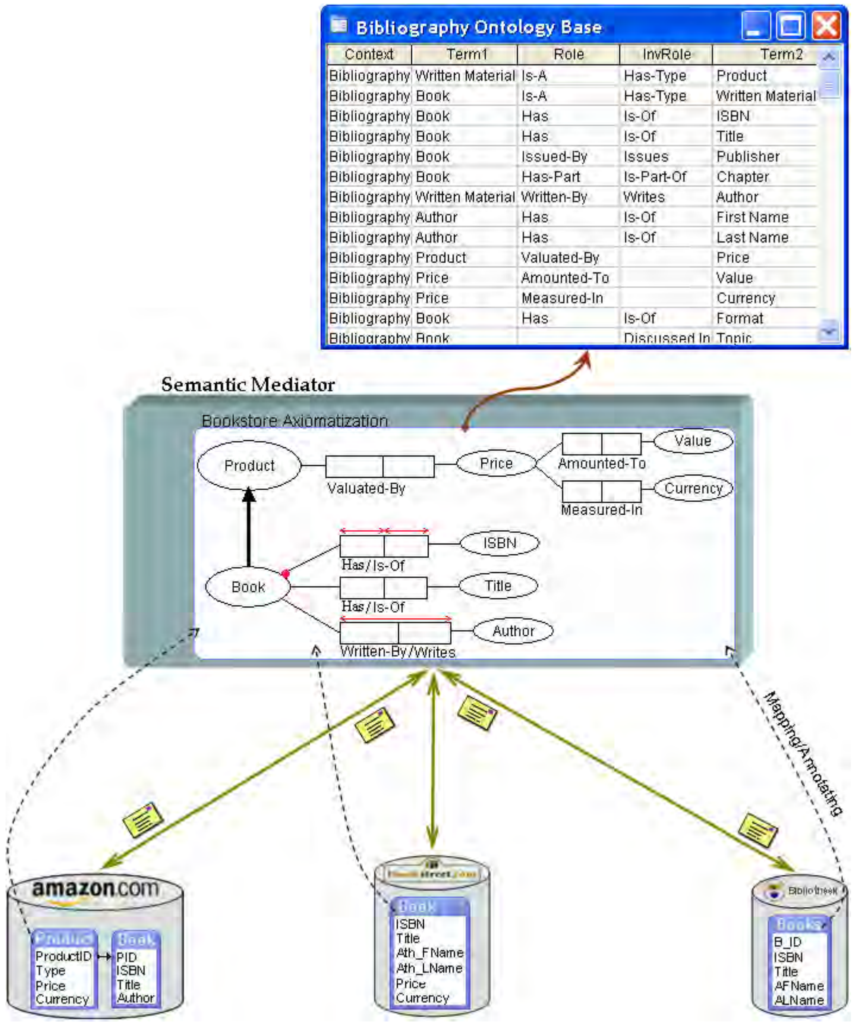
Speaking in operational terms, the eventual execution of an ontological commitment by an interpreter (or reasoner) provides an enforcer for the constraints on that application's intended behavior with its concepts, as laid down in the application's axiomatization. It is important to realize at this point that applications can run meaningfully *without* such an executed formal commitment to an ontology case –in fact most of today's legacy applications do– but clearly this happens "at their own risk" of producing –literally– meaningless results now and then.

3.4.1 Example

This example is based on that presented in Section 3.1. In this application scenario software agents wish to interoperate through a semantic mediator in order to exchange data messages and share business transactions (see Figure 9). The interoperation is enabled by the sharing of the same Bookstore axiomatization, i.e. as a global and legal

data model[23]. The data source (or its "export schema" [ZD04]) of each agent is mapped into the shared axiomatization. All exchanged data messages (e.g. those formed in XML, RDF, etc.) may be validated whether they conform to the rules and the ontological view declared in the Bookstore axiomatization, for example under classical first-order model-theoretic semantics [R88].
The ontological view of the above bookstore axiomatization specifies which concepts are relevant for the task(s) of this application scenario. These concepts correspond to explicit lexons in the ontology base, or they might be derived from these lexons. One can see in the ontology base that a 'Book' is not explicitly a 'subtype of' a 'Product' as specified in the Bookstore axiomatization. This subsumption is derived from the two lexons {<Bibliography: Book, Is-A, Has-Type, Written Material>, <Bibliography: Written Material, Is-A, Has-Type, Product>}. Based on these subsumptions, some inheritance also might be derived; for example, 'Book' inherits the relationship <Bibliography: Book, Written-By, Writes, Author> from its Written Material supertype. The choice of which concepts and relations should be committed to is an application-specific issue or may be subject to a usability perspective. See our discussion on this in Section 2.
In the Bookstore axiomatization, four rules are declared and may be conveniently verbalized: 1) each Book Has at least one ISBN; 2) each Book Has at most one ISBN; 3) each ISBN Is-Of at most one Book; 4) a Book may be Written-by several Author/s)and 5) an Author may Write/s several Book/s.
Notice that this approach enables usability perspectives to be encountered and encoded outside domain axiomatization. In turn, this indeed increases the usability of application axiomatizations as well as it increases the reusability of the underlying domain axiomatization.

[23] This way of sharing and using axiomatizations (as global schema) seems more applicable to data integration and mediation systems [ZD04]. They can also be used to describe web services [NM02]. For example, an axiomatization could be specified for each web service (to describe the "static" information provided to/by a web service), so that all agents accessing a web service share the same axiomatization.


Bibliography Ontology Base

Context	Term1	Role	InvRole	Term2
Bibliography	Written Material	Is-A	Has-Type	Product
Bibliography	Book	Is-A	Has-Type	Written Material
Bibliography	Book	Has	Is-Of	ISBN
Bibliography	Book	Has	Is-Of	Title
Bibliography	Book	Issued-By	Issues	Publisher
Bibliography	Book	Has-Part	Is-Part-Of	Chapter
Bibliography	Written Material	Written-By	Writes	Author
Bibliography	Author	Has	Is-Of	First Name
Bibliography	Author	Has	Is-Of	Last Name
Bibliography	Product	Valuated-By		Price
Bibliography	Price	Amounted-To		Value
Bibliography	Price	Measured-In		Currency
Bibliography	Book	Has	Is-Of	Format
Bibliography	Book		Discussed In	Topic

Fig. 9 Meaningful semantic interoperation between Bookstore applications.

Depending on the application scenario, application axiomatizations may be used in different ways. For example, in the Semantic Web and information search/retrieval scenarios, declaring rules might be not important because the main idea of these scenarios is to expand (rather than to constrain) queries. Filtering the unwanted results (i.e. illegal models) usually then falls within the responsibility of the application itself. In [J05] we presented such an application scenario of an ontology-based user interface, in which complaint web forms meaningfully share, through explicit ontological commitments, a vocabulary of relationships stored as a DOGMA Server lexon base.

To increase usability of application axiomatizations, they might be specified in multiple specification languages, such as DAML+OIL, OWL, RuleML, EER, UML, etc. Figure 10 shows the above Bookstore axiomatization expressed in OWL.

```
….
<owl:Class rdf:ID="Product" />
<owl:Class rdf:ID="Book">
  <rdfs:subClassOf rdf:resource="#Product" />
</owl:Class>
<owl:Class rdf:ID="Price" />
<owl:Class rdf:ID="Value" />
<owl:Class rdf:ID="Currency" />
<owl:Class rdf:ID="Title" />
<owl:Class rdf:ID="ISBN" />
<owl:Class rdf:ID="Author" />
<owl:ObjectProperty rdf:ID="Valuated-By">
  <rdfs:domain rdf:resource="#Product" />
  <rdfs:range  rdf:resource="#Price" />
</owl:ObjectProperty>
<owl:DataProperty rdf:ID=" Amounted-To .Value">
  <rdfs:domain rdf:resource="#Price" />
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DataProperty>
<owl:DataProperty rdf:ID="Measured-In.Currency">
  <rdfs:domain rdf:resource="#Price" />
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DataProperty>
<owl:DataProperty rdf:ID="Has.ISBN">
  <rdfs:domain rdf:resource="#Book" />
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#integer "/>
</owl:DataProperty>
<owl:DataProperty rdf:ID="Has.Title">
  <rdfs:domain rdf:resource="#Title" />
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DataProperty>
<owl:ObjectProperty rdf:ID="Written-By">
  <owl:inverseOf rdf:resource="#Writes "/>
  <rdfs:domain rdf:resource="#Book" />
  <rdfs:range  rdf:resource="#Author" />
</owl:ObjectProperty>
<owl:Restriction>
  <owl:onProperty rdf:resource="# Has.ISBN " />
  <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardinality>
</owl:Restriction>
….
```

Fig. 10. An OWL representation of the Bookstore ontological commitment.

Although both representations share the same intended meaning of concepts at the domain (/ontology base) level, notice the disparities between ORM and OWL in representing the Bookstore axiomatization. For example, ORM does not distinguish between DataProperties and ObjectProperties as does OWL. This is an example of an epistemological difference[24]. The ORM uniqueness constraint that spans over "Written-By/Writes" cannot be expressed in OWL, as it is implied by definition[25].

[24] Epistemology level: The level that deals with the knowledge structuring primitives (e.g. concept types, structuring relations, etc.). [G94].

[25] The formalization of ObjectProperties in OWL does not allow the same tuple to appear twice in the same set, such as Written-By = {<author1, book1>, < author1, book1>,…}.

The other uniqueness and mandatory constraints are all expressed as a cardinality restriction in OWL.
The logical and epistemological disparities described above (which are induced by the difference between the formalizations and the constructs of both languages) illustrate different ways of characterizing the legal models of application axiomatizations. The choice of which language is more suitable for specifying application axiomatizations depends on the application scenario and perspectives. For example, ORM and EER are mainly suitable for database and XML (-based) application scenarios since they are comprehensive in their treatments of the integrity of data sets. For inference and reasoning application scenarios, description logic based languages (such as OWL, DAML, etc.) seem to be more applicable than other languages, as they focus on the expressiveness and the decidability of axioms. See [J07][J07b] for the complete formalization of ORM in description logics. In this work we also identify which ORM constructs cannot be mapped into OWL. We have also implemented this formalization to reason about ORM diagrams using Racer (See [JD06]).
Allowing different languages, optimized techniques, or methodologies to be deployed at the application axiomatization level will indeed increase the usability of these axiomatizations. A recent application axiomatization language called Ω-RIDL [VDM04] has been developed within the DOGMA framework. It is claimed to better suited to the database applications' commitment to an ontology base.

4 Discussion and conclusions

In this chapter, we have presented a comprehensive view of the DOGMA approach for ontology engineering. We have shown how application verses domain axiomatizations can be well articulated. We have introduced the notion of an ontology base for capturing domain axiomatizations, and the notion of application axiomatizations by which particular applications commit to the intended meaning of domain vocabulary.
In the following we summarize the main advantages of our approach:
Increase reusability of domain axiomatization, as well as usability of application axiomatizations. As we have shown in this chapter, the application-independence of an ontology is increased by separating domain and application axiomatizations. Usability perspectives have a negligible influence on the independence of a domain axiomatization, because ontology builders are prevented from encoding their application-specific axioms. In other words, domain axiomatizations are mainly concerned with the characterization of the "intended models" of concepts, while application axiomatizations are mainly concerned with the characterization of the "legal models" of these concepts.
Allows different communities to create and maintain domain axiomatization (typically public) and application axiomatizations (typically local). Indeed, domain experts, lexicographers, knowledge engineers, and even philosophers, may contribute to the development, maintenance, and review phases of domain axiomatizations. It is needless for them to know why and how these axiomatizations will be used. Application-oriented experts can also contribute to and focus on the development phases of application axiomatizations, without needing to know about the correctness

of domain axioms. Hence, we offer an ontology representation model that is capable of distributed and collaborative development.

Allows the deployment of differently optimized technologies and methodologies to each articulation. For example, relational database management systems can be used (with high scalability and performance) to store and retrieve large-scale ontology bases. Natural language parsing and understanding techniques can be employed for extracting lexons from texts. Different specification languages can be used to specify application axiomatizations and these increase the usability of these axiomatizations.

Furthermore, *the importance of linguistic terms in ontology engineering is observed and incorporated* in our approach [SD04]. Not coincidentally, our approach allows for the adoption and reuse of many available lexical resources to support (or to serve as) domain axiomatizations. Lexical recourses (such as lexicons, glossaries, thesauruses and dictionaries) are indeed important recourses of domain concepts. Some resources focus mainly on the morphological issues of terms, rather than categorizing and clearly describing their intended meanings. Depending on its description of term meaning(s), its accuracy, and maybe its formality[26], a lexical resource can play an important role in ontology engineering.

An important lexical resource that is organized by word meanings (i.e. concepts, or called synsets) is WordNet [MBFGM90]. WordNet offers a machine-readable and comprehensive conceptual system for English words. Currently, a number of initiatives and efforts in the lexical semantic community have been started to extend WordNet to cover multiple languages. As we have discussed in section 3.2.1, the consensus about domain concepts can be gained and realized by investigating these concepts at the level of a human language conceptualization. This can be practically accomplished e.g. by adopting the informal description of term meanings that can be found in lexical resources such as WordNet, *as glosses*.

Acknowledgement. We are in debt to all present and former colleagues in STARLab for their comments, discussion, and suggestions on the earlier version of this work.

References

[BC88] Bylander, T., Chandrasekaran, B.: Generic tasks in knowledge-based reasoning: The right level of abstraction for knowledge acquisition. In Knowledge Acquisition for Knowledge Based Systems. Vol. 1. Academic Press, London. (1988) pp. 65–77

[BHGSS03] Bouquet, P., van Harmelen, F., Giunchiglia, F., Serafini, L., Stuckenschmidt H.: C-OWL: Contextualizing ontologies. Proceedings of the second International Semantic Web Conference - ISWC'03, Sanibel Island, Florida. October (2003)

[BM99] Bench-Capon T.J.M., Malcolm G.: Formalising Ontologies and Their Relations. Proceedings of DEXA'99. (1999) pp. 250–259

[BSZS06] Bouquet, P., Serafini, L., Zanobini, S., and Sceffer, S. 2006. Bootstrapping semantics on the web: meaning elicitation from schemas. In Proceedings of the World Wide Web Conf (WWW '06). ACM Press, New York, NY, 505-512. (2006).

[CGDL01] Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Identification constraints and functional dependencies in description logics. In Proceedings of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI 2001), pages 155–160, 2001.

[26] i.e., the discrimination of term meanings in a machine-referable manner.

[CJB99] Chandrasekaran, B., Johnson, R., Benjamins, R.: Ontologies: what are they? why do we need, them?. IEEE Intelligent Systems and Their Applications. 14(1). Special Issue on Ontologies. (1999) pp. 20–26.

[DDM06] De Moor, A.; De Leenheer, P.; Meersman, M.: DOGMA-MESS: A Meaning Evolution Support System for Interorganizational Ontology Engineering. In the 14th International Conference on Conceptual Structures (ICCS 2006), Volume 4068, Aalborg, Denmark, p.189-203, (2006)

[DDM07] De Leenheer, P., de Moor, A., and Meersman, R.: Context Dependency Management in Ontology Engineering: a Formal Approach. Journal on Data Semantics VIII, LNCS 4380, Springer-Verlag, pp. 26-56, (2007).

[DHHS01] Degen, W., Heller, B., Herre, H. and Smith, B.: GOL: Towards an Axiomatized Upper-Level Ontology. In Formal Ontology in Information Systems. Proceedings of the FOIS 2001. ACM Press. New York: October (2001) pp. 34–46

[DM05] De Leenheer P., Meersman R., Towards a formal foundation of DOGMA ontology Part I: Lexon base and concept definition server, TR STAR-2005-06, Brussel, 2005

[DSM04] De Bo, J., SpynsP., Meersman, R. : Assisting Ontology Integration with Existing Thesauri. In, On the Move to Meaningful Internet Systems 2004. In, Meersman R., Zahir T. et al.,(eds.), p.801-818,LNCS 3290, (2004)

[G04] Gangemi, A.: Some design patterns for domain ontology building and analysis. An online presentation (http://www.loa-cnr.it/Tutorials/OntologyDesignPatterns.zip April 2004)

[G94] Guarino, N.: The Ontological Level. In R. Casati, B. Smith and G. White (eds.), Philosophy and the Cognitive Science. Hölder-Pichler-Tempsky, Vienna: 443-456. (1994)

[G95] Gruber, T.: Toward principles for the design of ontologies used for knowledge sharing. International Journal of Human-Computer Studies, 43(5/6) (1995)

[G98a] Guarino, N.: Formal Ontology in Information Systems. Proceedings of FOIS'98, IOS Press, Amsterdam. (1998) pp. 3–15

[G98b] Guarino, N.: Some Ontological Principles for Designing Upper Level Lexical Resources. In: A. Rubio, N. Gallardo, R. Castro and A. Tejada (eds.): Proceedings of First International Conference on Language Resources and Evaluation. ELRA - European Language Resources Association, Granada, Spain. (1998)

[GG95] Guarino, N. and Giaretta, P.: Ontologies and Knowledge Bases: Towards a Terminological Clarification, in: Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing, N. Mars (ed.), pp 25-32, IOS Press, Amsterdam (1995).

[GGMO01] Gangemi, A., Guarino, N., Masolo, C., and Oltramari, A.: Understanding toplevel ontological distinctions. Proceedings of IJCAI-01 Workshop on Ontologies and Information Sharing. AAAI Press. Seattle, USA, (2001) pp. 26–33

[GGO02] Gangemi, A., Guarino, N., Oltramari A., Borgo, S.: Cleaning-up WordNet's top-level. Proceedings of the 1st International WordNet Conference. January (2002)

[GHW02] Guizzardi, G., Herre, H., Wagner G.: Towards Ontological Foundations for UML Conceptual Models. proceedings of the 1st International Conference on Ontologies, Databases and Application of Semantics (ODBASE'02), Lecture Notes in Computer Science, Vol. 2519, Springer-Verlag, Berlin. (2002) pp. 1100–1117

[GN87] Genesereth, M.R., Nilsson, N.J.: Logical Foundation of Artificial Intelligence. Morgan Kaufmann. Los Altos, California. (1987)

[GW02] Guarino, N. and Welty, C.: Evaluating Ontological Decisions with OntoClean. Communications of the ACM, 45(2). (2002) pp. 61–65

[HST99] Ian Horrocks, Ulrike Sattler, and Stephan Tobies. Practical reasoning for expressive description logics. In H. Ganzinger, D. McAllester, and A. Voronkov, editors, Proceedings of the 6th International Conference on Logic for Programming and Automated Reasoning (LPAR'99), number 1705 in Lecture Notes in Artificial Intelligence, pages 161–180. Springer-Verlag, 1999.

[Inn+03] Persidis A., Niederée C., Muscogiuri C., Bouquet P., Wynants M.: Innovation Engineering for the Support of Scientific Discovery. Innovanet Project (IST-2001-38422), deliverable D1. (2003)

[J05] Jarrar, M.: Towards Methodological Principles for Ontology Engineering. PhD thesis, Vrije Universiteit Brussel, May 2005.

[J05a] Jarrar, M.: Modularization and automatic composition of Object-Role Modeling (ORM) Schemes. In OTM 2005 Workshops, proceedings of the International Workshop on Object-Role Modeling (ORM'05). Volume 3762, LNCS, Pages (613-625), Springer. ISBN: 3540297391. November 2005.

[J06] Jarrar, M.: Towards the notion of gloss, and the adoption of linguistic resources in formal ontology engineering. In proceedings of the 15th International World Wide Web Conference (WWW2006). Scotland. Pages 497-503. ACM Press. ISBN: 1595933239. May 2006.

[J07c] Jarrar, M.: Towards Effectiveness and Transparency in e-Business Transactions, An Ontology for Customer Complaint Management. A book chapter in "Semantic Web Methodologies for E-Business Applications". IGI Global. ISBN: 978-1-60566-066-0. Chapter 7. (October 2008)

[J07] Mustafa Jarrar: Towards Automated Reasoning on ORM Schemes. -Mapping ORM into the DLR_idf description logic. Proceedings of the 26th International Conference on Conceptual Modeling (ER 2007). Volume 4801, LNCS, Pages (181-197), Springer. ISBN:9783540755623. New Zealand. November 2007

[J07b] Mustafa Jarrar: Mapping ORM into the SHOIN/OWL Description Logic- Towards a Methodological and Expressive Graphical Notation for Ontology Engineering. In OTM workshops, proceeding of the International Workshop on Object-Role Modeling (ORM'07). Volume 4805, LNCS, Pages (729-741), Springer. ISBN: 9783540768890. Portogal. November, 2007

[JDM03] Jarrar, M., Demy, J., Meersman, R.: On Using Conceptual Data Modeling for Ontology Engineering. Journal on Data Semantics, Special issue on "Best papers from the ER/ODBASE/COOPIS 2002 Conferences", LNCS Vol. 2800, Springer. ISBN: 3-540-20407-5. October (2003) pp. 185–207

[JE06] Jarrar, M., Eldammagh, M.: Reasoning on ORM using Racer. Technical report, Vrije Universiteit Brussel, Brussels, Belgium, August 2006.

[JH07] Mustafa Jarrar and Stijn Heymans: Towards Pattern-based Reasoning for Friendly Ontology Debugging. Journal of Artificial Intelligence Tools. Volume 17. No.4. World Scientific Publishing. August 2008.

[JM02a] Jarrar, M., Meersman, R.: Formal Ontology Engineering in the DOGMA Approach. In proceedings of the International Conference on Ontologies, Databases, and Applications of Semantics (ODBase 2002). Volume 2519, LNCS, Pages: 1238-1254, Springer. ISBN:3540001069. October 2002.

[KN03] Klein, M., Noy.: A component-based framework for ontology evolution. Technical Report IR-504, Vrije Universiteit Amsterdam. March (2003)

[KTT03] Kerremans, K., Temmerman, R. and Tummers, J.: Representing multilingual and culture-specific knowledge in a VAT regulatory ontology: support from the termontography approach. In OTM 2003 Workshops. Tübingen: Springer Verlag. (2003)

[M55] Martinet, A.: Economie des changements phonétiques, Berne: Francke, (1955) pp. 157-158

[M93] McCarthy, J.: Notes on Formalizing Context. Proceedings of IJCAI'93. Morgan-Kaufmann. (1993)

[M96] Meersman, R.: An essay on the Role and Evolution of Data(base) Semantics. In: Meersman, R., Mark L. (eds.): Proceeding of the IFIP WG 2.6 Working Conference on Database Applications Semantics (DS-6). CHAPMAN & HALL. Atlanta, USA. (1996)

[M99a] Meersman R.: The Use of Lexicons and Other Computer-Linguistic Tools. In: Zhang Y., Rusinkiewicz M, & Kambayashi Y. (eds.): Semantics, Design and Cooperation of

Database Systems, The International Symposium on Cooperative Database Systems for Advanced Applications (CODAS'99). Springer Verlag. Heidelberg. (1999) pp. 1–14

[M99b] Meersman R., Semantic Ontology Tools in Information System Design. In, Ras, Z. & Zemankova, M.,(eds.), Proceedings of the ISMIS 99 Conference, LNCS 1609, Springer Verlag. (1999) pp. 30–45

[M01] Meersman R., Ontologies and Databases: More than a Fleeting Resemblance, in d'Atri A. and Missikoff M. (eds), OES/SEO 2001 Rome Workshop, Luiss Publications, 2001

[MBFGM90] Miller, G. Beckwith, R., Fellbaum, F., Gross, D., Miller, K.: Introduction to wordnet: an on-line lexical database. International Journal of Lexicography, 3(4). (1990) pp. 235–244

[N90] Nöth, W.: Handbook of Semiotics. Bloomington, IN : Indiana University Press (1990)

[NM02] Nakhimovsky, A., Myers, T.: Web Services: Description, Interfaces and Ontology. In: Geroimenko, V., Chen, C. (eds.): Visualizing the Semantic Web. Springer. ISBN 1-85233-576-9. (2002) pp. 135–150.

[P05] Pretorius A. J. , Visual Analysis for Ontology Engineering. Journal of Visual Languages and Computing , 16(4) : 359 - 381, 2005.

[PFP+92] Patil, R., Fikes, R., Patel-Schneider, P., McKay, D., Finin, T., Gruber, T., Neches, R.: The DARPA Knowledge Sharing Effort: Progress Report. Proceedings of Knowledge Representation and Reasoning. (1992) pp. 777–788

[PS06] Pazienza, M., Stellato, A.: Linguistic Enrichment of Ontologies: a methodological framework Second Workshop on Interfacing Ontologies and Lexical Resources for Semantic Web Technologies (OntoLex2006), Italy, 24-26 May 2006

[Q91] Qmair, Y.: Foundations of Arabic philosophy. Dar al-Shoroq. Beirut, ISBN 2-7214-8024-3. (1991)

[R88] Reiter, R.: Towards a Logical Reconstruction of Relational Database Theory. In: Readings in AI and Databases. Morgan Kaufman. (1988)

[S03a] Smith, B.: Ontology. In: Floridi, L. (eds.): Blackwell Guide to the Philosophy of Computing and Information. Oxford: Blackwell. (2003) pp. 155–166

[S95] Shapiro, S.: Propositional, First-Order And Higher-Order Logics: Basic Definitions, Rules of Inference, Examples. In: Iwanska, L., Stuart, S., Shapiro, (eds.): Natural Language Processing and Knowledge Representation: Language for Knowledge and Knowledge for Language. AAAI Press/The MIT Press, Menlo Park, CA. (1995)

[T00] Temmerman, T.: Towards New Ways of Terminology Description, the sociocognitive approach. John Benjamins Publishing Company. Amsterdam. ISBN 9027223262. (2000)

[TV06] Trog D., Vereecken J.: Context-driven Visualization For Ontology Engineering. Computer Science, Brussels, p.237, (2006)

[SMJ02] Spyns, P., Meersman, R., Jarrar, M.: Data modelling versus Ontology engineering. SIGMOD Record 31(4):12-17. ISSN: 01635808. March 2002.

[V82] Van Griethuysen, J.J., (Eds.): Concepts and Terminology for the Conceptual Schema and Information Base. International Standardization Organization, Publication No. ISO/TC97/SC5- N695. (1982)

[VDM04] Verheyden, P., De Bo, J., Meersman, R.: Semantically unlocking database content through ontology-based mediation . InProceedings of the 2nd Workshop on the Semantic Web and Databases (in conjuction with the 30th International Conference on Very Large Databases), LNCS 3372, Springer Verlag. (2004)

[WG01] Welty, C., Guarino, N.: Support for Ontological Analysis of Taxonomic Relationships. Journal of Data and Knowledge Engineering. 39(1). October (2001) pp. 51–74

[ZD04] Ziegler, P., Dittrich, K.: User-Specific Semantic Integration of Heterogeneous Data: The SIRUP Approach. In Proceeding of the International Conference on Semantics of a Networked World. LNCS, Springer, Paris, France. June (2004) pp. 14–44 .