UNDERSTANDING, BUILDING, AND USING ONTOLOGIES

A commentary to "Using Explicit Ontologies in KBS Development", by van Heijst, Schreiber, and Wielinga

Nicola Guarino
LADSEB-CNR, National Research Council
Corso Stati Uniti 4, I-35127 Padova, Italy
guarino@ladseb.pd.cnr.it

1. Introduction

In their paper on "Using Explicit Ontologies in KBS Development", van Heijst and colleagues[1] seem to take for granted Bylander and Chandrasekaran's hypothesis on the strong dependence of knowledge represesentation on the nature and the inference strategy of the problem at hand, the so-called *interaction problem*:

> 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.
> [Bylander and Chandrasekaran 1988]

The fact that the van Heijst and colleagues don't attempt to explore in detail the arguments sustaining this hypothesis is particularly puzzling, since they admit that it contradicts one of the main assumptions of their well-known KADS approach [Schreiber *et al.* 1993], namely the separation between domain knowledge and problem-solving knowledge. They report two reasons brought by Bylander and Chandrasekaran to support their hypothesis: "Firstly, the application task determines to a large extent which *kinds* of knowledge should be encoded. (...) Secondly, the knowledge must be encoded in such a way that the inference strategy used can reason efficiently".

In fact, at a closer inspection, the statement from Bylander and Chandrasekaran reported above mentions the problem of *representing* knowledge, and it is related therefore to the *symbol level*. Now, it is certainly true that the interaction problem exists at this level, but it seems plausible to assume that its importance decreases at the knowledge level, to which the whole issue of ontology belongs. Let us try for instance to re-state Bylander and Chandrasekaran's statement at the knowledge level: "the knowledge required to solve some problem is strongly affected by the nature of the problem...". Put in this way, it sounds even trivial. Notice however that this formulation doesn't refer to the way this knowledge is encoded, but simply to the relevance relationship between the knowledge and the problem. In other words, at the knowledge level the interaction problem reduces to the first of the two "reasons" reported above. Of course, a specific piece of knowledge may be

1 In the following, I shall refer to this paper as "the paper", and to its authors as "the authors".

more or less *relevant* for a particular task, but nothing tells us that this knowledge is peculiar, *specific* of such task.

I will defend here the thesis of the independence of domain knowledge. This thesis should not be intended in a rigid sense, since it is clear that – more or less – ontological commitments always reflect particular points of view; rather, what I would like to stress is the fact 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).

In this systematic quest for reusability, the potential role of a discipline like *formal ontology* appears evident. I have explored elsewhere [Guarino 1995] how a stronger connection between formal ontology, conceptual analysis and knowledge engineering can contribute to establish the foundations of the emerging field of "ontological engineering". Following the lines of the paper by van Heijst and colleagues, I shall discuss here how the principles of ontological engineering can be used in the practice of knowledge-based systems building, focusing in particular on the interplay between ontologies and problem-solving knowledge and on the ways to build and update ontologies. I will first analyze in section 2 the various definitions of the term "ontology" proposed by the authors, trying to make clear the problems bound to the formal relationships between ontologies and conceptualizations. Then, in section 3, I will address the role of ontologies in the knowledge engineering process. A crucial issue in this respect is the relationship between the ontology library and the application ontology, and the role played by the latter in the update of the former. The vision I will defend is that of application ontologies as specializations of a more general library, which includes *task* and *method* ontologies [Falasconi and Stefanelli 1994, Gennari *et al.* 1994] as well as domain ontologies. The ontology-based knowledge modelling methodology proposed by the authors will be discussed in detail in section 4, following their example taken from the medical field. Finally, in the conclusions I will stress the role of domain analysis, often absent in current methodological proposals where the task analysis is strongly privileged.

2. Understanding Ontologies

2.1 Ontologies and Conceptualizations

Before discussing the principles for ontology libraries construction, the authors report various definitions of the term "ontology" appeared in the literature, trying to establish a comprehensive definition. Together with Pierdaniele Giaretta, I have analyzed this terminological problem in detail in [Guarino and Giaretta 1995], focusing in particular on the possibility of giving a formal interpretation to the most cited definition of an ontology in the knowledge sharing community, i.e. Gruber's definition:

(1) An ontology is an explicit specification of a conceptualization.

[Gruber 1994]

The main problem of this definition is that it is claimed to be based on the formal notion of "conceptualization" introduced in [Genesereth and Nilsson

1987], while it can only be accepted in terms of an *intuitive* understanding of that term. The origin of this problem, in my opinion, lies in the bad use of the term "conceptualization" made by Genesereth and Nilsson.

It may be useful here to report some of the discussion appeared in [Guarino and Giaretta 1995]. Let us consider the example given by Genesereth and Nilsson. They take into account a situation where two piles of blocks are resting on a table (Fig. 1a). According to them, a possible conceptualization of this scene is given by the following structure:

<{*a*, *b*, *c*, *d*, *e*}, {*on*, *above*, *clear*, *table*}>

where {*a*, *b*, *c*, *d*, *e*} is the *universe of discourse*, consisting of the five blocks we are interested in, and {*on*, *above*, *clear*, *table* } is the set of the relevant relations among this blocks, of which the first two, *on* and *above*, are binary and the other two, *clear* and *table*, are unary[2]. The authors make clear that objects and relations are extensional entities. For instance, the *table* relation, which is understood as holding of a block if and only if that block is resting on the table, is just equal to the set {*c*, *e*}. It is exactly such an extensional interpretation that originates our troubles.

Let us notice first that Genesereth and Nilsson used natural language terms (like *on*, *above*) in the metalanguage chosen to describe a conceptualization. This could perhaps be seen as nothing more than a didactical device. However, these linguistic terms do convey essential information in order to understand the criteria used to consider some sets of tuples as the *relevant* relations. Such an extra information cannot be accounted for by the conceptualization itself.

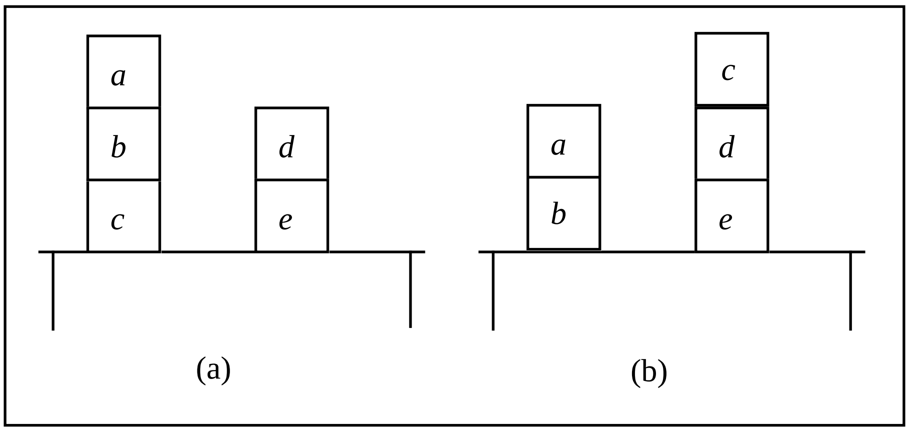

Fig. 1. Blocks on a table (from [Guarino and Giaretta 1995]).
(a) A possible arrangement of blocks.
(b) A different arrangement. Also a different conceptualization?

Referring to the example given, consider a different arrangement of blocks, where *c* is on the top of *d* and *a* and *b* form a separate stack standing

2 In the original example also a function is considered, but we omit it here for the sake of simplicity.

on the table (Fig. 1b). The corresponding structure would be different from the previous one, generating therefore a different conceptualization. Of course there is nothing wrong in such a view, if one is only interested in isolated snapshots of the block world. But the meanings of the terms used to denote the relevant relations are still the same, since they are invariant with respect to the possible configurations of blocks. In fact, in the metalanguage adopted in their book, Genesereth and Nilsson would adopt the same symbols (*on*, *above*, *clear*, *table*) to denote the new conceptualization. We prefer to say in this case that the *states of affairs* are different, but the conceptualization is the same. The structure proposed by Genesereth and Nilsson seems to be more apt to represent a state of affairs rather than a conceptualization.

In order to capture such intuitions, the linguistic terms we have used to denote the relevant relations cannot be thought of as mere comments, informal extra-information. Rather, the formal structure used for a conceptualization should somehow account for their *meaning*. As the logico-philosophical literature teaches us, such a meaning cannot coincide with an extensional relation. In [Guarino and Giaretta 1995] we have presented a way to represent this meaning in terms of an *intensional structure* inspired to Montague's semantics. According to this intensional interpretation, a conceptualization accounts for the intended meanings of the terms used to denote the relevant relations. These meanings are supposed to remain the same if the actual extensions of the relations change due to different states of affairs. This means that, for instance, the actual extensions of the relation *on* in the two examples of Fig. 1a and 1b belong to the *same* conceptualization. Intuitively, we can see a conceptualization as a set of informal rules that constrain the structure of a piece of reality, which an agent uses in order to isolate and organize relevant objects and relevant relations: the rules which tell us whether a certain block is *on* another one remain the same, independently of the particular arrangement of the blocks.

2.2 What are ontologies: a still debated issue

Hoping to have clarified the sense of the term "conceptualization", let us now analyze the various definitions of "ontology" appearing in the paper by van Heijst, Schreiber and Wielinga. Besides Gruber's definition, they report two more definitions taken from the literature:

(2) A (AI-) ontology is a theory of what entities can exist in the mind of a knowledgeable agent.

[Wielinga and Schreiber 1993]

(3) An ontology for a body of knowledge concerning a particular task or domain describes a taxonomy of concepts for that task or domain that define the semantic interpretation of the knowledge.

[Alberts 1993]

The definition (2) is similar to the classical notion of ontological commitment introduced by Quine [Quine 1961][3]. According to him, a logical the-

3 At least, in the sense that the ontological commitment of a logical theory is intended as a set of individuals. Expressions like "can exist in the mind" are however extraneous to Quine.

ory is ontologically committed to the entities it quantifies over. As discussed in [Guarino *et al.* 1994], such a notion however is too weak for our purposes, since we want not only an account of what exists, but also an account of the *structure* of what exists. This structure is implied in the *language* we use: this is the reason why, as noticed by the authors, the term "ontology" is often used as a synonym of "terminology" in the AI community.

The definition (3) is more problematic. Although Van Heijst and colleagues correctly observe that it is the semantic interpretation of the terms of a domain that constitutes an ontology, the formulation reported is misleading, since "the semantic interpretation of the knowledge ... concerning a particular task or domain" doesn't regard the taxonomy only, but it also involves the factual situations holding in that domain. The distinction between *domain knowledge* and *domain ontology* made by the authors (p. 12) is therefore not caputred by this definition. Moreover, according to definitions (1) and (2), an ontology can be much more than a taxonomy of concepts, involving in particular constraints and interrelations among concepts. Hopefully, it should also concern *more* than one particular task or domain. Alberts' definition seems therefore both partial and inaccurate, and I cannot see how the authors consider it as "not contradictory" with (1) and (2), coming up with the following "unifying" definition:

(4) An ontology is a explicit knowledge level specification of a conceptualization, (...) which may be affected by the particular domain and task it is intended for.

[van Heijst *et al.* 1996]

Despite the difficulties of recognizing definitions (2) and especially (3) as present in (4), this new formulation clarifies a little bit Gruber's definition (under the assumption of the correct interpretation of "conceptualization" discussed above), stressing that ontologies belong to the knowledge level and that they *may* depend on particular points of view. We must observe however that it is exacly the degree of such dependence which determines the reusability and therefore the *value* of an ontology.

There is another, nicer and more recent definition of ontologies proposed by Tom Gruber in a message to the SRKB (Shared Reusable Knowledge Bases) mailing list, reported in a recent work by Uschold and Gruninger [Uschold and Gruninger 1996]:

(5) Ontologies are agreements about *shared* conceptualizations. Shared conceptualizations include conceptual frameworks for modelling domain knowledge; content-specific protocols for communication among inter-operating agents; and agreements about the representation of particular domain theories. In the knowledge sharing context, ontologies are specified in the form of definitions of representational vocabulary. A very simple case would be a type hierarchy, specifying classes and their subsumption relationships. Relational database schemata also serve as ontologies by specifying the relations that can exist in some shared database and the integrity constraints that must hold for them.

(Tom Gruber, 1994, SRKB Mailing list)

The nice thing of this formulation is that ontologies and conceptualizations are kept clearly distinct. An ontology in this sense is not a *specification* of a conceptualization, but a (possibly incomplete) agreement *about* a conceptualization. Therefore, as suggested in [Guarino and Giaretta 1995], we can have different degrees of detail in this agreement depending on the purpose of the ontology (see § 2.3). Formulation (5) agrees very well with our refined version of (1):

(6) An ontology is an explicit, partial account of a conceptualization.
[Guarino and Giaretta 1995]

I consider this definition as quite satisfactory from my point of view. Since it relies however on the revised notion of conceptualization discussed above, it may result obscure for somebody. Hoping to clarify things more, I would like to suggest the following further definition:

(7) An ontology is a logical theory that constrains the indended models of a logical language.

To be precise, I refer here to the set of non-logical symbols (predicates and functions) of a logical language (what is usally called the *signature* of the language), used as "primitives" for a particular representation purpose. An example of this signature is the set of symbols used by Genesereth and Nilsson to denote what they call a conceptualization: {*on*, *above*, *clear*, *table*}. An ontology in this case would provide the axioms which constrain the meaning of these predicates, like, for example, $\neg$on(X,X).

Finally, there is still a last definition of ontology that the authors consider as compatible with Gruber's (and our) one:

(8) An ontology is an explicit, partial specification of a conceptualization that is expressible as a meta-level viewpoint on a set of possible domain theories for the purpose of modular design, redesign and reuse of knowledge-intensive system components.
[Schreiber *et al.* 1995]

In short, an ontology is considered in this case as "a meta-level description of a knowledge representation" (p. 10). In my opinion, this definition introduces a source of confusion, due to the fact that the "meta-level view" is considered by the authors as *intrinsic* to their notion ontology. Indeed, the ontologies they have used in their work on KAKTUS [Wielinga *et al.* 1994] and on the VT-domain [Schreiber and Terpstra 1996] are meta-level ontologies, since their domain is the meta-level domain of representation primitives. However, the ontologies present in the core library built by Falasconi and Stefanelli [Falasconi and Stefanelli 1994] can hardly be seen as "meta-level".

In other words, ontologies can be either "meta-level" or not, depending on the nature of their domain. In their experience on KAKTUS and the VT domain, the authors have brilliantly shown how to use meta-level ontologies for knowledge reuse purposes, exploiting mapping rules between an ontology and another [Schreiber *et al.* 1995]; what they call "representational meta-models" in the paper discussed here (p. 70) are again ontologies, developed for the particular purpose of knowledge transformation: their domain is constituted by the "types of expressions allowed in a knowledge representation formalism". It is important to remark here that these meta-level theories can be still conceived as logical theories (see remark at the end of section 3.2).

2.3 Ontology kinds

I will now briefly comment the classification of ontology proposed by the authors. They distinguish two dimensions, "the amount and type of structure of the conceptualization and the subject of the conceptualization" (p. 11).

The *first dimension* is far from being clear. First of all, it is hard to see how what they call "information ontologies" can be considered as ontologies at all. A "specification of the record structure" of a database cannot be considered as an ontology according to the definition given by the authors, since it belongs to the *symbol level*. A database schema can be seen as an ontology as long as it is a *conceptual* database schema, while a *logical* database schema belongs again to the symbol level. Level 1 of the PEN&PAD model [Rector *et al.* 1993] can't be seen as an ontology since it describes *factual knowledge* (medical records report "Observations - What the agents have heard, seen and done"). Considering this as an ontology would violate the distinction made by the authors between domain knowledge and domain ontology. Rather, what consitutes an ontology is the *vocabulary* used to describe medical records, but this collapses into what have been called “terminological ontologies”.

In turn, the distinction between terminological and knowledge-modelling ontologies is also not clear. Due to the problems of the information ontologies, the contrast between them and knowledge-modelling ontologies is misleading, and the meaning of the “richer internal structure” of the latter remains vague. The reference to the level 2 of the PEN&PAD model increases the confusion, since this seems to refer only to meta-level knowledge related to the ways of observing and relating medical facts.

In conclusion, I believe that there is no reason to hypothesize a distinction among ontologies on the basis of “the amount and type of structure of their conceptualization”. Maybe, as suggested above, a distinction can be made among different ontologies on the basis of the degree of detail used to characterize a conceptualization. A very detailed ontology gets closer to specifying the intended conceptualization (and therefore may be used to *establish consensus* about the utility of sharing a particular knowledge base which commits to that ontology), but it pays the price of a richer language. A very simple ontology, on the other hand, may be developed with particular inferences in mind, in order to be shared among users which *already agree* on the underlying conceptualization. We may distinguish therefore between *documenting ontologies* and *shareable ontologies,* or maybe *off-line* and *on-line ontologies*. Very simple ontologies like lexicons can be kept on-line, while sophisticated theories accounting for the meaning of the terms used in a lexicon can be kept off-line.

The *second dimension* is much clearer: depending on the subject of the conceptualization, the authors distinguish between *application ontologies, domain ontologies*, *generic ontologies* and *representation ontologies.* Before discussing in detail the relationships between the former three kinds in the next section, I would like to comment here briefly on the notion of *representation ontology*. In this case, the underlying conceptualization addresses representation primitives, like those defined in Ontolingua’s *Frame Ontology* [Gruber 1993]. According to the discussion made in the previous section, a representation ontology is therefore an example of meta-level ontology. I must remark however that the citation to the work done together with Luca Boldrin [Guarino and Boldrin 1993] about the supposed ontological neutrality of such primitives is incorrect, since in that paper we argued *against* this neutrality, “which makes possible, for instance, to interpret arbitray unary predicates either as classes or properties, and arbitrary binary predicates either as slots or relations” (p. 2). In short, it is perfectly valid to adopt ontologically neutral representation primitives to build a particular knowledge base, but to build a reusable ontology it may be necessary to assign a more restricted semantics to the representation primitives, taking into account the ontological distinctions that can be made within unary and binary relations. This position has been further discussed in [Guarino 1994, Guarino 1995], where I distinguished between a neutral *epistemological level* and a non-neutral *ontological level;* ontological distinctions between unary primitives have been discussed in [Guarino *et al.* 1994].

3. Ontologies in the Knowledge Engineering Process

3.1 The interaction problem

As mentioned in the introduction, van Heijst and colleagues postulate a strong influence, in the ontology development process, of the particular application at hand. However, the interaction problem does not hold to the same extent for all concepts; they suggest therefore to distinguish between an *ontology library,* that contains more or less reusable knowledge across different applications, and an *application ontology,* containing the definitions specific to a particular application.

Surprisingly, they don’t introduce a *method ontology,* as done in [Falasconi and Stefanelli 1994, Gennari *et al.* 1994]. Rather, they propose to introduce two attributes, *domain-specificity* and *method-specificity,* “to determine to what extent and under which circumstances a concept can be reused”. Once a large ontology library has been built, this indexing scheme can surely simplify the construction of application ontologies; the key issue however regards *the methodology used to update an incomplete ontology library while building a particular application ontology*.

The risk is to give too much importance to the interaction problem, considering a new concept introduced in the application ontology to be *specific* of a certain domain and a certain method (i.e., of the application ontology at hand), without making any attempt to generalize it in such a way to be reused for more general tasks and domains. As mentioned in the introduction, in fact, a concept may be *relevant* for a particular task whithout being necessarily *specific* of that task.

3.2 Using application ontologies to update the ontology library

The risk mentioned above is especially evident when considering the methodology suggested by the authors for building and updating the ontology library. The key role in this process appears to be played by the *application ontology*. The notion of application ontologies has been introduced in [Gennari *et al.* 1994], for the purpose of i) reducing the gaps between domain and method ontologies, and ii) allowing the domain expert to use the same language adopted in the application at hand, which may be different from the language used in the ontology library. In the work made by the PROTÉGÉ group, the application ontology is mainly used to produce a tool used to “populate” the application knowledge base, while in the paper by van Heijst and colleagues the authors propose to exploit application ontologies *also* for the task of updating the ontology library. In both cases, the construction of the application ontology is mainly a “creative process”, with a very limited support for what concern the content of the ontology itself. What distinguishes the two groups is the kind of link established between the application ontology and the ontology library: in the PROTÉGÉ group, a method ontology is intended to be part of the ontology library besides the domain ontology, and the link with the application ontology is handled by explicit mapping rules acting as “mediators”; in the KADS group this link is handled by an indexing mechanism.

The former solution appears to me more powerful, since it makes explicit the way the application ontology is related to the method ontology: a simple indexing mechanism may be unsatisfactory for this purpose, since the choice of the particular specificity index for a given concept remains obscure[4]. The two solutions may be considered as roughly equivalent (as noticed indeed in [Gennari *et al.* 1994]) if the purpose is to build an application ontology by exploiting an existing ontology library, or to facilitate integration of different representation formalisms (section 6); the matter is however completely different if we want to update the ontology library while building the application ontology. This latter goal is of course highly desirable, as underlined by van Heijst and colleagues (section 3.3), but it has been not addressed until now due to its difficulty.

The hard issue is to limit the effects of the interaction problem, separating the domain knowledge from the method knowledge. To this purpose, the relevance relationship between domain concepts and methods must be captured. With the explicit introduction of a *method ontology*, this relevance relationship can be represented by means of a “mapping relation” between the application ontology and the method ontology, where the role played by each single concept within a particular method is made explicit. In this way, the effects of the interaction problem can be limited by representing the nature of the interaction, rather than assuming its effects as intrinsic to the concepts being modeled.

However, the technique based on mapping relations developed by the PROTÉGÉ group is still limited for ontology building purposes, since it is mainly based on a syntactic mapping. I believe that we can push further this approach, seeing an application ontology as a specialization of both the domain and the method ontology.

Consider for example the concept `cost` appearing in the CASNET application ontology reported by the authors (p. 26). It is not clear why it gets the method-specificity “CASNET ranking”, and not the more general “ranking by weight to cost ratio” reported in the method ontology shown at page 25. Presumably, the authors think it may be “dangerous” to assign a more general meaning to such a concept, which is assumed to be dependent on the particular application. No attempt to generalize is foreseen by the proposed methodology in this phase, and the reusability of `cost` remains restricted to the CASNET application.

4 By the way, van Heijst already use the technique of mapping rules for the task of knowledge-based integration (section 6), and I do not see why they don’t use the the same technique to link the application ontology to the method ontology.

A different conception of the application ontology is reported in Fig. 2. The application ontology is built by specializing both the domain and the method ontology. The concept `CASNET-cost` is the cost of an observation leading to a pathophysiological state, which plays the role of the cost of an hypothesis within the method "ranking by weight-to-cost-ratio". What motivates its presence in the application ontology is the fact that it plays a specific *role* in CASNET's ranking procedure. The ontological requirements of this procedure are represented explicitly in the method ontology, and the CASNET-specific `cost` satisfies the range restriction of the attribute `has-cost` of the concept `hypothesis` belonging to the method ontology.

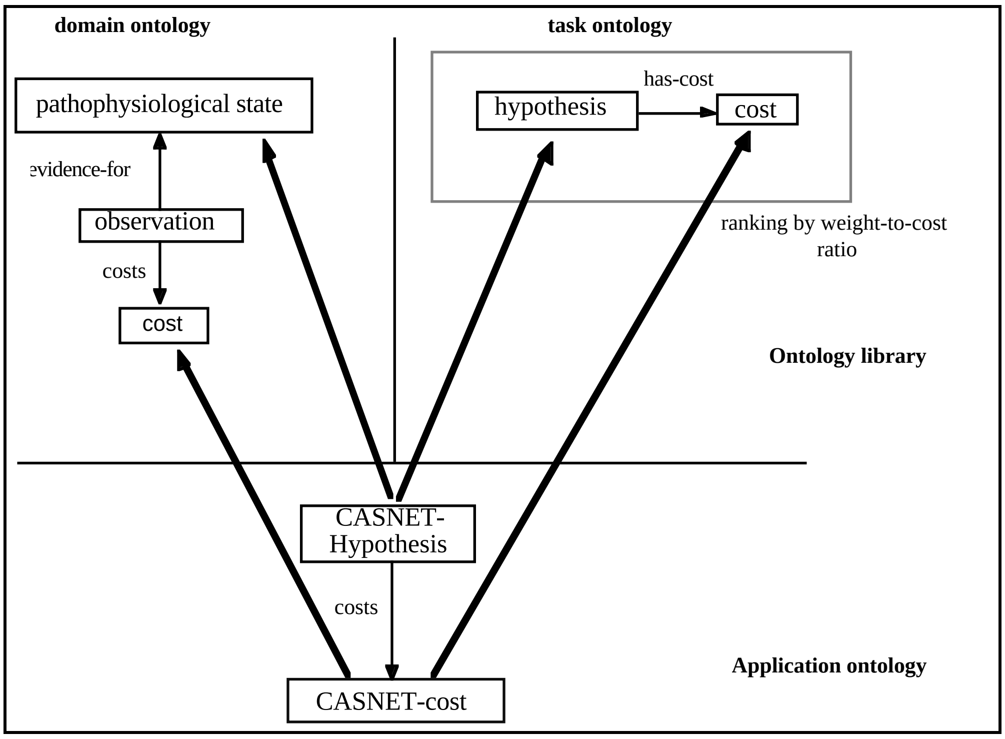

Fig. 2. Application ontology as a specialization of the ontology library. Thick arrows represents subsumption links.

According to this view, all concepts appearing in the application ontology reported by the authors are specializations of *both* the domain ontology and the method ontology[5], and the "mapping rules" between the two ontologies would be extremely simple. Focusing on the application ontology amounts to highlighting those concepts which are *relevant* for a particular application, being specializations of its method(s) and its domain: the application ontology is just a *view* of the more general ontology.

5 More in general, any concept of the application ontology would be either a specialization or an instance of a (meta)concept in the ontology library. Of course, this choice would imply a richer structure both in the application ontology and the ontology library.

Notice that, while the left part of Fig. 2 (the domain ontology) can be considered as relatively static, the bottom and right parts change when the problem solving strategy changes. In this way, the ISA arcs linking the application ontology to the task ontology “can be seen as *attributing context-specific semantics* to domain knowledge elements” ([Schreiber *et al.* 1995], section 3.1). However, once the application ontology is fixed, it has a rigorous model-theoretic semantics, in contrast with the approach based on syntactical mapping relations discussed in [Schreiber *et al.* 1995].

In sum, I believe that the specific role played by single items of domain knowledge into the decision-making process should be made explicit in the application ontology. The indexing mechanism proposed by the authors appears to be to coarse for this purpose. As admitted by the authors (p. 25), it makes the task of populating a largely incomplete ontology library by scoring the newly defined concepts in the application ontology particulary difficult, or almost impossible in presence of strong interaction problems.

4. An Example of Ontology-Based Modelling Methodology

I will briefly comment in the following the concrete example of a knowledge modelling methodology sketched by the authors in section 5.1 and discussed in section 7, in order to elucidate the ideas and the criticisms reported above. In this example, I will assume the existence of two (imaginary) ontologies, a domain ontology and a task and method ontology, built upon principles slightly different from those adopted by the authors.

4.1 Informally describing the domain and the task

It is important here to isolate the domain andtask*vocabulary*. Informal methods such *concept maps* {Gaines ...} may turn to be very useful.

4.2 Modelling the task

A suitable *task and method ontology* should supplement in my opinion the “inference and task model” realized with QUITE. It is important that this ontology, accordingly to [Gennari *et al.* 1994], includes all the concepts necessary to describe the inferential process, from the very abstract concepts related to the inference scheme to the more specialized concepts specific for single methods. For the sake of simplicity, I will call this ontology *method ontology*.

4.3 Modelling the domain

This step is not present in the methodology proposed, since the authors assume that a domain model already exists in the ontology library. However, a phase in the modeling process where the basic structure of the domain is analyzed seems to be as important as the task analyisis accomplished in the previous step.

4.4 Building the application ontology

Here the authors propose a number of guidelines, presented more as heuristic criteria than formal steps. The presence of a task ontology makes it possible to specify these steps in a more rigorous way. We comment here some of the steps reported by the authors, presenting a different modelling strategy.

Diagnostic hypotheses

Here me must find, in the domain ontology, the specializations of the concept “hypothesis” (belonging to the method ontology) which are relevant for the domain at hand. The method ontology imposes general constraints on such concepts (say, an hypothesis must be a *disorder*). We look here at the domain vocabulary. If we find a term suitable to be considered as an hypothesis, we try to find it in the domain ontology, either by direct syntactic matching or by means of suitable linguistic tools like thesauri (Wordnet, UMLS). If we don’t find it, we may consider to introduce a definition for it, and to classify the new concept in the domain ontology according to such definition. Then, we must check whether this concept can be also classified as an “hypothesis”. If this test fails, we must either change the definition of the newly created concept or consider some other term as a candidate.

In the example given, the misleading term “syndrome” used in section 7.3.1 does not appear in the domain description of section 7.1. Sticking to the originally elicited vocabulary helps therefore to avoid the introduction of uncontrolled terms. In fact, a simple *linguistic analysis* of the last two sentences of section 7.1. allows us to conclude that the “therapeutic action” (which is the *goal* of the task at hand) is driven by “the gradings of the lesions to each of the systems”. The term “grading of a lesion” is therefore a good candidate for an hypothesis[6]. Since “grading” is obviously an attribute of “lesion”, we look for the main concept “lesion” in the domain ontology, which has, say, the attribute “severity”, restricted to “grading”. We consider therefore “severity of a lesion” as a candidate. But the severity of a lesion is nothing else than a grading, i.e. a qualitative value. In the method ontology, the hypotheses of a diagnostic task are restricted to be “disorders”, and these are disjoint from gradings. We exclude therefore “severity of a lesion”, and consider the the possible specializations of “disorder”. Here we find “lesion”, which admits any lesion of a particular severity as a subconcept. Since “severity” is an attribute of “lesion”, and not a slot, all the lesions whith different severity are mutually exclusive due to the semantics of the representation primitives used (defined in the frame ontology). In conclusion, the hypotheses for the ARS applications are specializations of “lesion” (in fact, “organ-system lesions”). Notice that no “lesion-subtype” relation is used; notice also that “finding” is immediately excluded as a candidate since it does not satisfy the restrictions on “hypothesis”.

Patient findings

Since the term “patient findings” appears explicitly in the task model, the corresponding concept belongs in my opinion to the method ontology and not to

6 We may use “lesion-grading”, but hyphenation may be misleading in this stage. The term “grading of a lesion” makes clear what is the main concept and what is the attribute. See [Guarino 1992].

the domain ontology as assumed by the authors (see however their note at page 82). None of the terms used for the informal domain description (section 7.1) appears to be suitable as a candidate. The method ontology is thererore used to elicit the concepts needed for the application ontology. As discussed in the paper, particular expressions named “lesion-indications” are assumed to be specializations of the concept “finding”.

Diagnostic data

The authors observe that “one aspect that distinguishes raw data from findings in general is that data are directly observable”. This fact should be modeled in the method ontology, which can be used therefore to elicit the concepts related to diagnostic data in the application ontology. It is also important, in my opinion, that these concepts reflect the information contained in the raw data available for the application, without any implicit abstraction process. In the example given, it seems to me a much better strategy to represent a single datum as `(ars-datum (erythema (location head-and-neck) (degree 2)))` rather than `(ars-datum head-and-neck-erythema = 2)`. Among other things, this choice blocks the possibility to exploit general knowledge related to the location of a finding.

Diagnostic abduction

Due to the choice made when determining the diagnostic hypotheses, the authors are not able to specialize the relation `manifestation-of` supposed to exist in the domain ontology, since it holds for disorders while `ars-lesion-gradings` have been defined as expressions. We don’t have this problem, since - as we have seen - hypotheses are restricted to be disorders by the method ontology. However, what deserves attention is the way out adopted by the authors: they introduce an application-specific concept called `ars-manifestation-of`, which is modification of `manifestation-of`[7]. Now this approach of modifying a concept as a result of a type mismatch, introducing a new concept with a very similar name, seems to me extremely dangerous. First, I believe that a rigorous naming discipline should be part of the methodology. A very natural criterion in this respect is the following: any concept whose name is X-C should be a specialization of C[8]. It is easy to see how violations to this criterion generate confusion and compromise readability. Second, the introduction of an ad-hoc concept in the application ontology which is not a specialization of an existing concept in the domain ontology violates the (refined) definition of application ontology that I have proposed.

5. Conclusions

I will try here to summarize the observations made in this commentary paper, giving at the same time an overall assessment of the main issues related to ontology-based knowledge modelling.

---

7 By the way, how can an ARS-manifestation be a manifestation of an expression?

8 See [Guarino 1992] for a similar criterion applied to attributes, called attribute consistency postulate: any X of a Y must be a X.

First, I would say that ontologies can be of some help for building knowledge-based systems if the interaction problem is not taken too seriously. Fortunately, as I have argued in the introduction, this problem mainly regards the symbol level, and does not affect the knowledge level too much. Ontologies, on the other hand, need to be described at the knowledge level, and sometime their full translation to the symbol level is not even necessary: their purpose is to characterize a conceptualization, limiting the possible interpretations of the non-logical symbols of a logical language in order to establish consensus about the knowledge described by that language. I hope to have contributed to a clarification of the related meanings of "ontology" and "conceptualization" in section 2.

In order to avoid the effects of the interaction problem, a greater emphasis to *domain analysis* should be given. In my opinion, the attention deserved to domain analysis, conceived as an independent activity, should be greater or equal to that devoted to task analysis. Confirming a tendence largely present in the KA literature, the paper by van Heijst and colleagues gives in my opinion too much importance to task analysis, avoiding however at the same time to introduce an explicit task and method ontology. As shown with the simple example reported in Fig. 2, an explicit representation of task and method knowledge, along the lines of [Gennari *et al.* 1994] and [Falasconi and Stefanelli 1994] can help to systematically analyze the *knowledge roles* [McDermott 1988] played by the domain knowledge within a particular problem solving strategy, resulting in a very simple link between the application ontology and the ontology library, aimed to maximize abstraction, reusability and semantic coherence.

Suitable tools and techniques need still to be developed for domain analysis, and for ontology building in general. The authors admit that "the creative aspect of ontology construction" (i.e., that related to the *content* of ontology itself) "remains a task for the user" (p. 49), and assume that a "largely complete" ontology library already exists. This assumption however is far from being satisfied in many cases, and the crucial task is exactly to build the ontology library. It is clear that in this case the vision of "Model-Based KA" proposed by the authors (p. 30) must be abandoned in favour of "KA as Modeling": we cannot insist too much on model instantiation as a good strategy when we don't have good enough models.

Under this vision of "KA as Modeling", domain-modeling tools need to address content-related issues. This can be done by exploiting: i) linguistic resources such as thesauri, and ii) analysis techniques based on formal ontological principles.

I don't understand why linguistic analysis is almost absent in the literature related to ontology building for knowledge-based applications (besides the ontologies built for specific NL purposes, like PENMAN [Bateman *et al.* 1990], PANGLOSS [Knight and Luk 1994] or Mikrokosmos [Mahesh 1996]). If not a linguistic ontology, at least some on-line thesaurus like Wordnet would be of great help for an ontology building tool, allowing at the same time to i) pursue generality; ii) identify ambiguities and subtle differences in meaning; iii) enforce readability and consistency by means of linguistic discipline.

These NL-based analysis techniques should be integrated by formal ontological principles. For example, questions related to the mereo-topological structure and the dependence relationships holding for a particular concept or individual should be systematically asked; I have shown elsewhere [Guarino 1992, Guarino *et al.* 1994, Guarino *et al.* 1994] how formal properties like rigidity, countability, dependence can help a lot to clarify the ontological nature of a concept.

In conclusion, the elicitation of the intrinsic structure of domain-knowledge should be the main task of ontology-building tools. The goal of so-called ontological engineering is to develop theories, methodologies and tools suitable to elicit and organize domain knowledge in a reusable and "transparent" way. This *cognitive transparency* is in my opinion the main “added value” of an ontology.

Acknowledgements

This work has been done in the framework of a Special CNR Project on ontological and linguistic tools for conceptual modelling (Progetto Coordinato "Strumenti Ontologico-Linguistici per la Modellazione Concettuale”). I am grateful to Pierdaniele Giaretta and Massimiliano Carrara for their precious comments.

Bibliography

Alberts, L. K. 1993. *YMIR: an Ontology for Engineering Design.* University of Twente

Bateman, J. A., Kasper, R. T., Moore, J. D., and Whitney, R. A. 1990. A General Organization of Knowledge for Natural Language Processing: the PENMAN upper model. USC/Information Sciences Institute, Marina del Rey, CA.

Bylander, T. and Chandrasekaran, B. 1988. Generic tasks in knowledge-based reasoning: The right level of abstraction for knowledge acquisition. In B. R. Gaines and J. H. Boose (eds.), *Knowledge Acquisition for Knowledge Based Systems*. Academic Press, London.

Falasconi, S. and Stefanelli, M. 1994. A Library of Medical Ontologies. In *Proceedings of ECAI94 Workshop on Comparison of Implemented Ontologies*. Amsterdam, The Nederlands, European Coordinating Committee for Artificial Intelligence (ECCAI): 81-92.

Genesereth, M. R. and Nilsson, N. J. 1987. *Logical Foundation of Artificial Intelligence.* Morgan Kaufmann, Los Altos, California.

Gennari, J. H., Tu, S. W., Rothenfluh, T. E., and Musen, M. A. 1994. Mapping Domains to Methods in Support of Reuse. *IJHCS*, **41**: 399-424.

Gruber, T. 1994. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. *IJHCS*, **43**(5/6): 907-928.

Gruber, T. R. 1993. A translation approach to portable ontology specifications. *Knowledge Acquisition*, **5**: 199-220.

Guarino, N. 1992. Concepts, Attributes and Arbitrary Relations: Some Linguistic and Ontological Criteria for Structuring Knowledge Bases. *Data & Knowledge Engineering*, **8**: 249-261.

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

Guarino, N. 1995. Formal Ontology, Conceptual Analysis and Knowledge Representation. *International Journal of Human and Computer Studies*, **43**(5/6): 625-640.

Guarino, N. and Boldrin, L. 1993. Ontological Requirements for Knowledge Sharing. In *Proceedings of IJCAI workshop on Knowledge Sharing and Information Interchange*. Chambery, France: 1-5.

Guarino, N., Carrara, M., and Giaretta, P. 1994. Formalizing Ontological Commitment. In *Proceedings of National Conference on Artificial Intelligence (AAAI-94)*. Seattle, Morgan Kaufmann.

Guarino, N., Carrara, M., and Giaretta, P. 1994. An Ontology of Meta-Level Categories. In D. J., E. Sandewall and P. Torasso (eds.), *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference (KR94)*. Morgan Kaufmann, San Mateo, CA: 270-280.

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

Knight, K. and Luk, S. 1994. Building a Large Knowledge Base for Machine Translation. In *Proceedings of American Association of Artificial Intelligence Conference (AAAI-94)*. Seattle, WA.

Mahesh, K. 1996. Ontology Development for Machine Translation: Ideology and Methodology. New Mexico State University, Computing Research Laboratory MCCS-96-292.

McDermott, J. 1988. Preliminary steps toward a taxonomy of problem-solving methods. In S. Marcus (ed.) *Automating Knowledge Acquisition for Expert Systems*. Kluwer Academic Publishers.

Quine, W. O. 1961. *From a Logical Point of View, Nine Logico-Philosophical Essays.* Harvard University Press, Cambridge, Mass.

Rector, A. L., Nowlan, W. A., Kay, S., Goble, C. A., and Howkins, T. J. 1993. A Framework for Modelling the Electronic Medical Record. *Methods of Information in Medicine*, **32**: 109-119.

Schreiber, A. T. and Terpstra, P. 1996. Sysyphus-VT: a CommonKADS Solution. *IJHCS*, **44**: 373-402.

Schreiber, G., Wielinga, B., and Breuker, J. 1993. *KADS: A Principled Approach to Knowledge-Based System Development.* Academic Press, London.

Schreiber, G., Wielinga, B., and Jansweijer, W. 1995. The KAKTUS View on the 'O' Word. In *Proceedings of IJCAI95 Workshop on Basic Ontological Issues in Knowledge Sharing*. Montreal, Canada.

Uschold, M. and Gruninger, M. 1996. Ontologies: Principles, Methods and Applications. *The Knowledge Engineering Review*, **(in press)**.

van Heijst, G., Schreiber, A. T., and Wielinga, B. J. 1996. Using Explicit Ontologies in KBS Development. *International Journal of Human and Computer Studies*(this issue).

Wielinga, B., Schreiber, A. T., Jansweijer, W., Anjewierden, A., and van Harmelen, F. 1994. Framework and formalism for expressing ontologies. ESPRIT Project 8145 KACTUS, Free University of Amsterdam deliverable DO1b.1.

Wielinga, B. J. and Schreiber, A. T. 1993. Reusable and sharable knowledge bases: a European perspective. In *Proceedings of Proceedings of First International Conference on Building and Sharing of Very Large-Scaled Knowledge Bases*. Tokyo, Japan Information Processing Development Center.