Using the WordNet Concept Catalog and a Relation Hierarchy for Knowledge Acquisition

Philippe MARTIN

INRIA - ACACIA project
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex France
Phone: (33) 93.65.76.45 Fax: (33) 93.65.77.83 - E-mail:phmartin@sophia.inria.fr

Abstract. In order to guide a knowledge engineer in the design of ontologies, CGKAT (Martin, 1995), our knowledge acquisition tool, exploits the terminological knowledge base WordNet. Since the top-level concept types of this large general ontology are poorly structured, we subordinated/merged them into an extension of the top-level ontology adviced by Sowa (1992). The result is proposed in the initial concept type lattice of CGKAT. Since WordNet is an on-line system, only the WordNet top-level concept types need to be included in the lattice: CGKAT enables the user to search the WordNet ontology by browsing or lexical search, and dynamically places the retrieved concept types in the lattice. Any part of the lattice may be reorganized without loosing access to the WordNet ontology. Thus, only the WordNet part that is useful for an application has to be definitely included in the lattice. This part, which is very detailed, eases interpretation, validation, reuse and automatic inferences on knowledge of the application. In this article, we detail the mechanism of dynamic inclusion of the WordNet ontology in the lattice, and its interests for knowledge representation and reuse, and we give the rationales behind our top-level ontology. Lastly, we sum up the interests of the use of basic relations for KA (in (Martin, 1995) we have already presented a top-level ontology for 200 basic relation types gathered from the hypertext and knowledge representation literature).

1 Introduction

Buiding an ontology, that is a taxonomic catalogue of concept types and relation types, is a difficult, long and crucial part of the Knowledge Acquisition (KA) process. Most of the natural language concepts and relations may appear in documents which are sources of expertise (technical documents, interview retranscriptions, etc.) and a great number of these concepts might have to be clusterized or classified for the KA process. In this article, we give our approach to use the WordNet ontology (Miller & al., 1990) (which now organizes more than 91,000 concepts) in order to ease the building of the concept type lattice of an application, and its ulterior extensions or merging with other ontologies. We also sum up some interests of basic relations for KA (in (Martin, 1995) we have proposed guidelines for buiding the relation type hierarchy, and presented the top-level relation types of our richly organized set of about 200 basic relation types gathered from the hypertext and knowledge representation literature). Our work is implemented above CoGITo (Haemmerlé, 1995), a workbench for Conceptual Graphs (Sowa, 1984).

WordNet is a public domain on-line lexical reference system developed at Princeton University. Its design is inspired by current psycholinguistic theories of human lexical memory. In version 1.5, some 120,000 word forms of English nouns, verbs, adjectives and adverbs are organized into approximately 91,600 synonym sets (synsets), each representing an underlying lexical concept, that is, a word meaning (each synset is unique, all synsets are disjoint). Thus, given a lexical entry (a word or an expression), WordNet can extract its root (its word form) and give back its various meanings, that is, a list of synsets. The WordNet synsets are connected by semantic relations (e.g. IsA, Part-of, Cause-of, Attribute) and by lexical relations (e.g. Antonym).

We have found a way to automatically build a distinct concept type name with the names in a synset. Then, since the WordNet database is now accessible by a C functional interface, CGKAT (Martin, 1995), our KA tool, can exploit the WordNet database for helping its users to build a concept type lattice: it can search a concept type in WordNet with the exact name of this concept or with any word or expression which refers to it, and it can follow the semantic and lexical relations from a retrieved WordNet concept type.

A problem for KA, knowledge representation (KR), and knowledge inferencing, is that the WordNet concepts are not organized under a "genuine" ontology model (like for example the Situation Data Model of Tepfenhart (1992)): the concepts which nouns refer to are structured under ten exclusive concept categories, the concepts which verb refer to are structured under a very long (and ungiven) list of activities, and the concepts which adjectives and adverbs refer to are not structured by an IsA relation. Therefore, we have manually subordinated and merged the WordNet top-level concepts, and other ontological distinctions from the Situation Data Model, the PENMAN Upper Model (Bateman 1990), CYC (Lenat & Guha, 1990), Esch (1992) and Pfeiffer & Hartley (1992), into an extension of the situational ontology model proposed by Sowa (1992). (Knight & Luk (1994) have done a similar work on WordNet but with the PENMAN and ONTOS (Carlson & Nirenburg, 1990) top-level ontologies).

Since the WordNet top-level concept types are in the lattice, the whole WordNet ontology has not to be included: when the user *browses* on links between types or when he *searches* types with a lexical entry, CGKAT can dynamically *add* in the lattice the retrieved types (with all their WordNet supertypes which are not yet included in the lattice). The user may decide to keep in the lattice the types he really needs for his application (the other types are removed from the lattice when they are no more needed for display). He may specialize or restructure any part of the lattice without loosing access to the WordNet ontology, since the above *search&add* mechanism is independent of the lattice structuration. To sum up, let us say that for guiding the user in the building of its ontology, the WordNet ontology need not to be wholly included in the lattice but just dynamically included.

Using WordNet and our top-level ontology, the knowledge engineer does not have to worry about a *coherent* ontology model, nor how to *place* and *organize* in a coherent way its natural language concept types under this model. WordNet provides an organization for most of the natural language concepts, and we have done the hard work which is to merge and extend the Sowa situational ontology model with all the WordNet top-level concept types. Afterwards, the knowledge engineer has just to find WordNet concept types for the meanings he wants to express (e.g. with a lexical entry) and specialize these types for introducing application specific concept types or for expressing restrictions. Thus, he will build a safer, less brittle, more standard and more easily extensible ontology than without any guide (and above all, he will do it easily). His ontology will also be more comprehensible given that WordNet concept types are very structured, and have precise and detailed names and comments.

We will develop the interests of using WordNet for knowledge reuse later. First, we present the CGKAT interfaces which enables to view and browse large ontologies, including the WordNet one, using what we called a «dynamic» inclusion. Then, we give the rationales behind the top-level structuration proposed by CGKAT for concept types. Lastly, we will quickly present some interests of basic relations for KA and KR.

2 Handling ontologies in CGKAT

Before explaining the dynamic inclusion of WordNet in the concept type lattice, and the top-level ontology we propose for this lattice, we have to explain our conventions in the display of ontologies.

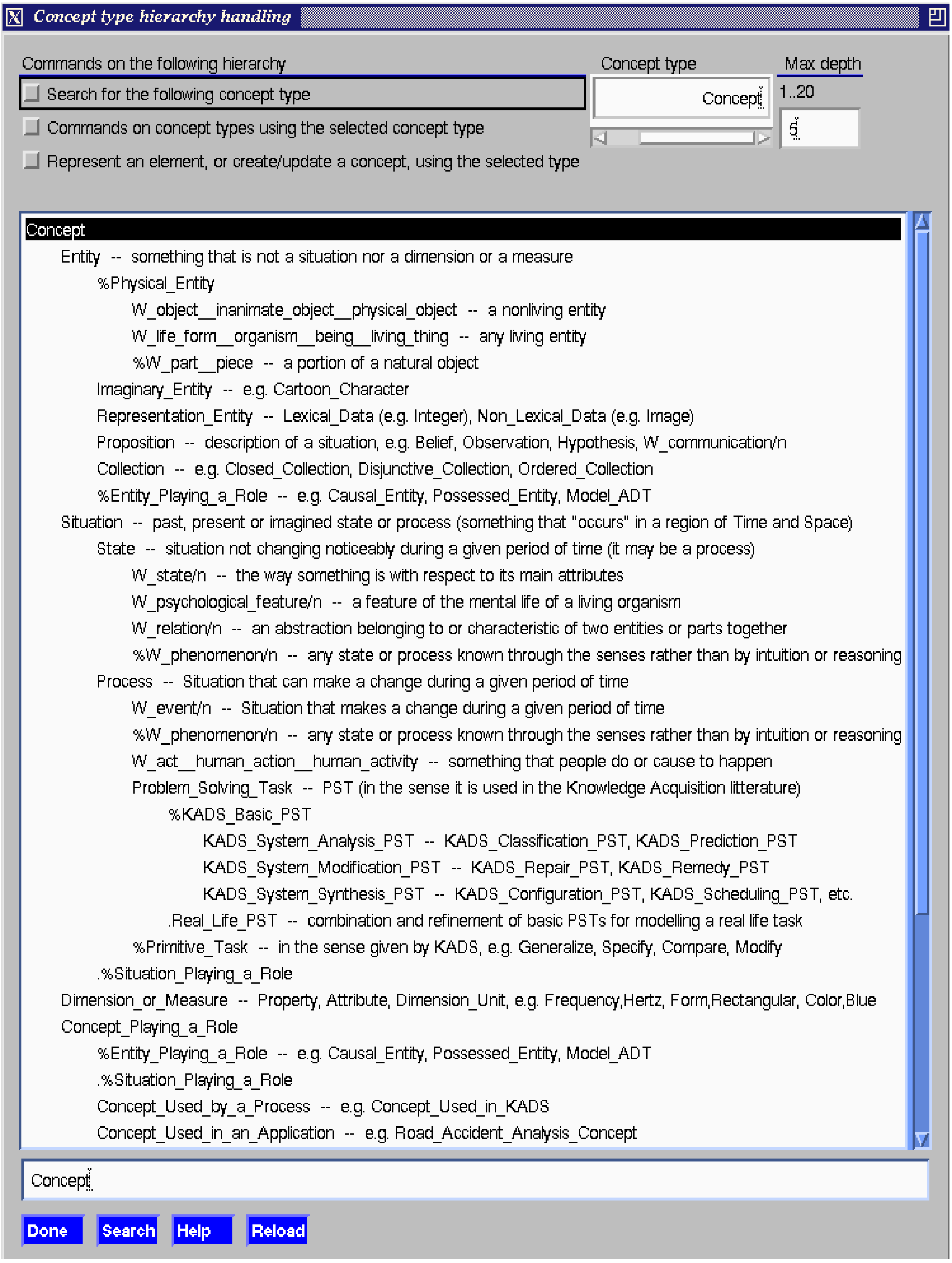

Figure 1: *The concept type handling menu showing some of the top-level types proposed by CGKAT.*

Figure 1 shows the top of the default concept type lattice proposed by CGKAT. Since the CGKAT browsers must display a lot of types (with their associated comment) in order to give the user a synthetic view of an ontology part and therefore to ease its searchs, we have prefered an hierarchically indented list instead of a graph layout. When a concept type is selected, its supertypes and its subtypes are displayed (more precisely, five supertypes are displayed whereas the depth for the subtypes is controlled by content of the «Max depth» number entry widget and the presence of the keyword «e.g.» in the comments of the subtypes). In Figure 1, «Concept», our name for the supertype of all types, is selected, hence no supertype is displayed. In order to highlight the types which have many supertypes, their type names are displayed preceded by a '%'. For more clarity, when these supertypes must be displayed (e.g. when any of their subtypes are manually selected or retrieved with a lexical entry) their type names are displayed preceded by a '^' and the supertypes of these supertypes are not displayed (see Figure 2). Lastly, for saving the user browsing time, the leaves (i.e. the types which only have «Absurd» for subtype) are preceded by '.'.

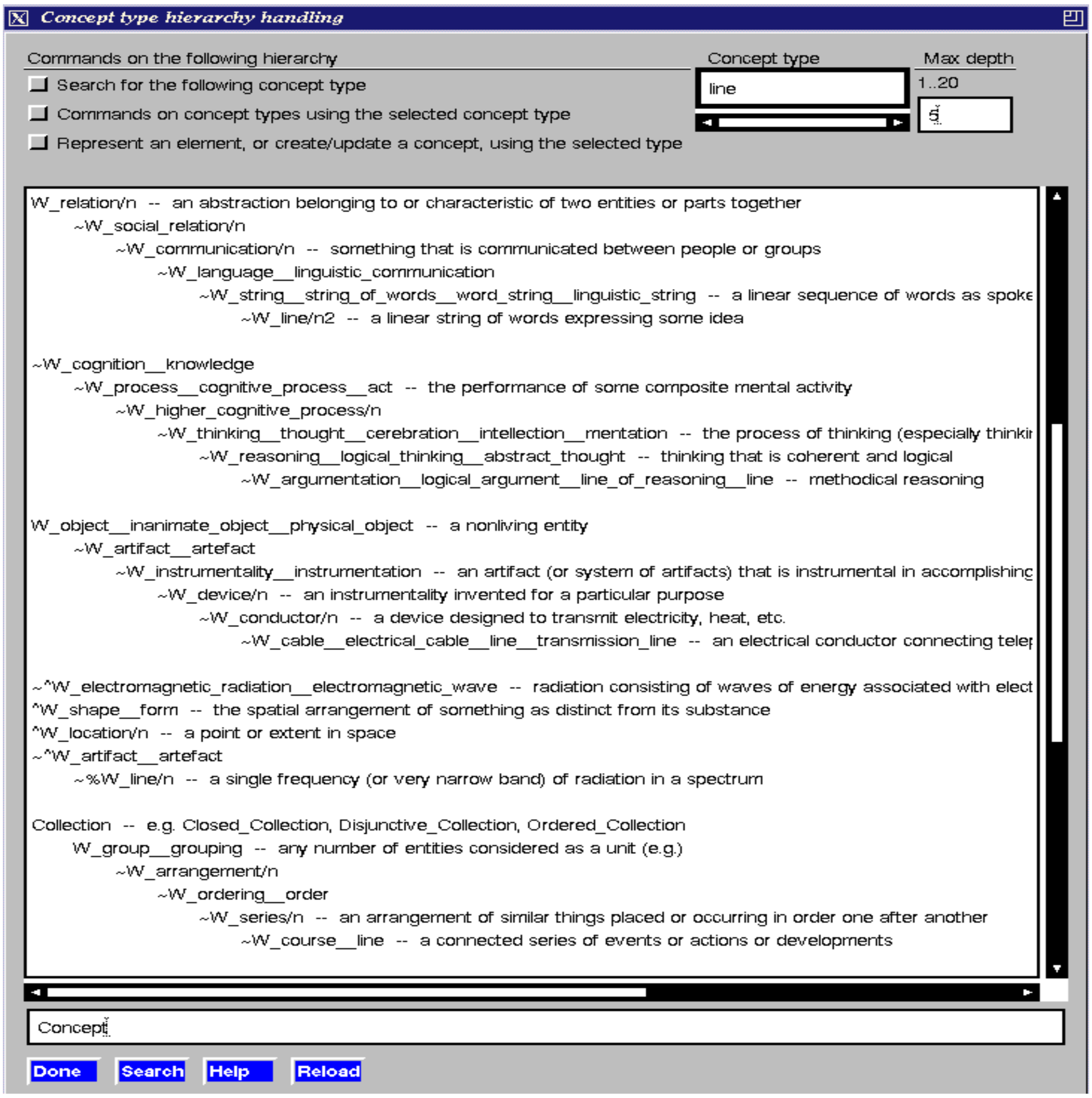

Figure 2: *The concept types (with their supertypes) proposed by Wordnet for the lexical entry «line».*

A type may be retrieved by browsing (i.e. by successive selections) but also with a lexical entry. If only the current lattice must be searched, the exact type name must be given. If the WordNet ontology is searched, any lexical entry (word or expression) known by WordNet may be used. WordNet will extract the word form of the lexical entry and gives back all the word meanings it knows for this word form. We will detail in the next section how CGKAT constructs concept type names with these word meanings and includes them in the lattice. When a type is selected, the user may apply some commands to it: adding of subtype or supertype, type removal, removal of subtypes (with warnings if some loaded GCs use them), etc.

The relation type hierarchy can be browsed and handled in the same way except that WordNet cannot be dynamically included into it. For the dynamic inclusion of WordNet in the concept type lattice, CGKAT only exploits the IsA relation between word meanings (synsets), assuming it is a Kind-Of relation. WordNet connects these synsets with other relations, e.g. Part-of, Cause-of and Attribute. Although we have not implemented it, we think that from the user point of view, these relations could be browsed and handled with the same interface as for the IsA relation. However, in the Conceptual Graph formalism (Sowa, 1984) (Sowa, 1993), apart from the Kind-of relation, relations between concept types are represented via type definitions, schemas, or metalevel graphs using concepts with second-order types and second-order relations like «Kind» and «Subtype». Therefore, even if other relations between types are implemented like the Kind-Of relation, they must be given a clear semantic interpretation in the conceptual graph theory, and a mechanism should probably be implemented for translating these relations into definitions, schemas or metalevel graphs when necessary.

3 Dynamic inclusion of WordNet in the concept type lattice of CGKAT

A synset represents a word meaning, is unique, and is connected to other synsets by an IsA relation[1]. Since these synsets very rarely represent individuals, the IsA relation may be assumed to be a Kind-Of relation and therefore be used to build a concept type lattice.

We found a way to automatically build a unique concept type name with the names in a synset. If there are at least two names in the synset, a simple concatenation of these names is sufficient. If there is only one name, a unique type may be built by adding to this name the initial of the grammatical category of the synonyms (e.g. 'n' for a noun, 'v' for a verb) and the sense number of this synset in this grammatical category (if this sense number is 0, then the sense number is ommited because there is no corresponding synset for the same category[2]). (These information can be accessed via the WordNet functional interface). Here are two examples which can be found in Figure 1. With the synset {object, inanimate object, physical object} CGKAT builds the concept type name W_object__inanimate_object__physical_object (the word forms «object», «inanimate object» and «physical object» are in the same synset because in some contexts they are synonyms and refer to the same concept). With the synset

1. The wordNet authors call the IsA relation between word meanings, an «Hyponym» relation, but agree that IsA is a term that is commonly used for such a relation. The following definition is used to build the Hyponym hierarchy: «a concept represented by the synonyms set {x, x', ...} is said to be a hyponym of the concept represented by the synonyms set {y, y', ...} if native speakers of English accept sentences constructed from such frames as *An x is a (kind of) y*». Hence, we think that parts of the WorNet ontology may be included in various lattice without ontological loss.

2. In this last case, for the encoding of synsets including only one word form, we have tried to not add systematically the initial of its grammatical category to the name, but then the time taken to check if the concept type name was unique (check in the WordNet databases of the other grammatical categories), was too long to be tolerable.

{state} CGKAT builds the concept type name W_state/n. Thus, all concept type names which come from WordNet begin by «W_» (Figure 1 and 2 give many examples). This sometimes makes very long concept type names, especially when the synonyms are verbs, but it makes them rather unambiguous. For its application, the knowledge engineer may specialize these concept types in order to use shorter names and to express semantic restrictions.

The WordNet concept types which are visible in Figure 1 correspond to top-level WordNet synsets. Figure 2 shows some of the various concept types (with their supertypes) which can be retrieved in WordNet with the lexical entry «line» (the comments are also retrieved with the types). In order to highlight the WordNet types which have just been dynamically inserted in the lattice, their concept type names are displayed preceded by a '~', which means that they are temporarily inserted: The user may decide to keep in the lattice the types he really needs for his application, the other ones are removed when they are no more needed for display.

The WordNet supertypes for a retrieved WordNet concept type are dynamically inserted in the lattice until one of the supertypes is already known in the lattice. If none of the supertypes is known, the upper one is placed under «Concept». This mechanism enables the user to specialize, to restructure or to delete any part of the lattice (our top-level or any part of the organization proposed by WordNet for concept types) without loosing access to the WordNet ontology with lexical searchs: the WordNet supertype of a WordNet type is dynamically inserted in the lattice only if this type did not belong to the lattice before. Then only the parts of WordNet which are not ovewritten by the user are retrieved. And since WordNet has many inadequacies and since it cannot be wholly adequate for any application, it is a necessity to overwrite or to complement some parts of its organization.

Similarly, when the user clicks on a WordNet concept type (temporary or not), CGKAT retrieves its first subtypes according to WordNet and inserts them in the lattice (temporarily if they are not yet known), except if the user has specified that it should not do that with this type (this information is saved in a hidden part of the comment of the type). If the user has defined other subtypes for this type, they are shown before the WordNet subtypes. Hence, as for the display of supertypes, the display of subtypes takes into account the user restructuration or completion of the WordNet ontology. To sum up, the knowledge engineer can always be guided by the highly structured WordNet ontology, even when he corrects or completes it.

We have said that some of the WordNet synsets represent individuals. Therefore CGKAT can build concept type names for these individuals and insert them temporarily in the lattice, e.g. W_Johann_Sebastian_Bach will be proposed as a subtype for an organist and a composer. The user should not keep such proposal in the lattice.

Another problem is: does the inclusion of concept types proposed by WordNet may change a lattice into a structure which is not a lattice ? Although the WordNet ontology is mainly a tree, the answer may be positive. Hence, a verification procedure should be run after each inclusion, or more realistically only when the user desires it since the duration of this check is proportional to the cube of the number of types in the lattice. If the WordNet ontology is not dynamically included but wholly included, each checking after some modifications in the lattice would take a very long time. Such checks and other helps to build the lattice will be introduced in CGKAT when it will be connected to the «cooperative program for the construction of a concept type lattice» of Chein & Leclère (1993).

4 Advantages of using the WordNet ontology for knowledge reuse

A KB that is built using concept types coming from WordNet, or specializations of these types, could be rather easily compared with another KB built in the same way since a lot of concept type names used in the two KB would be common and their meanings too[1]. If those concept types are organized a bit differently in the two KB, automatic procedures could detect the differences and help to resolve them[2]. As WordNet is very detailed, the knowledge engineer should rarely have to add intermediate types but rather specialize precise types of WordNet in order to express the shades of meanings needed for his application.

Therefore, in order to ease the use and reuse of the KB knowledge, we suggest to the knowledge engineer to *specialize WordNet concept types*. The knowlege engineer may also define his application types as subtypes of a type like Concept_used_in_an_application (see Figure 1). Hence, he may build and work on the minimal hierarchy necessary for its application, without being bothered by the high structuring offered by WordNet and our high level concept types, but without loosing them. *This structuring may be not useful for the final KBS but it is necessary for a good modelling, for powerful searchs and inferences, and for easing validation, extension and reuse*. Filters could always be applied when only a part of the ontology is needed.

5 The top-level structuration proposed by CGKAT for concept types

We have seen the WordNet ontology and our top-level ontology are just proposals: any part can be modified by the user. Let us present this top-level ontology and explain its rationales.

5.1 Structuring with the notions of situation, process, state, event, proposition and dimension

The first idea was to *merge* the top-level concept types of WordNet into the situational ontological model proposed by Sowa (1992) because it makes ontological distinctions which are not in WordNet but which are important for KR, especially in the Conceptual Graph context. Then, all the WordNet concept types are automatically classed according to these ontological distinctions. These distinctions are the notions of Situation, Time, Space, State, Process and Proposition. Here is an extract of Sowa (1992) which introduces them.

> «Situation semantics (Barwise & Perry, 1983) has been widely adopted as one of the most flexible ways of defining the semantics of language. Each situation is a finite configuration of some aspect of the world in a limited region of space and time. It may be a static configuration that remains unchanged for a period of time, or it may include processes and events that are causing changes. ...
> In conceptual graphs, a situation is represented by a context, which is a concept that contains one or more propositions that describe the situation.. The propositions in a context could be expressed by a paragraph of English sentences or by a collection of conceptual graphs.

The notion of Situation enables to group concepts to which temporal relations like «Point_in_Time» or «Duration» may be attached. Therefore, these relations may be signed on concepts of type Situation and concepts of type Time (the interests of signed relations for KR

1. The concept types coming from WordNet are precise and have detailled names and comments; therefore we think that they do not induce distant interpretations. This is also an important point for interpretation and validation of the KB. Apart from these advantages, if any other large general ontology would exist, it could also be used in the same way by a knowledge engineer for building a detailed and reusable ontology for its application.

2. Of course, a lot of problems will have to be solved during the merging of two lattices, for example, what to be done if different definitions or schemas are associated to the same concept types ? But this merging, or more generally the reuse of other works, is eased if the works rely on a same basis, e.g. WordNet.

and KA will be developed in section 6). The notion of Situation induces its complement, that is the notion of Entity which groups concepts to which attaching relations like Point_in_time or Duration is a non-sense. Figure 1 shows the definitions for Entity, Situation, State, Process[1] and Proposition, and the WordNet top-level subtypes we found for them[2]. We had to group several WordNet top-level concept types for the notions of time and space under the two concept types Time and Space (these two concept types are needed to give signatures to temporal and spatial relation types). The following subtypes of Time and Space, are also needed to sign more accurately some temporal and spatial relations: Point_in_Time, Time_Period, Point_in_Space and Space_Region. But the notions relative to these types are scattered in the WordNet ontology, then we could not group them and we have to let the knowledge engineer specialize these types with the temporal or spatial WordNet types he uses.

Similarly, in WordNet the notions of dimension, dimension unit, measure, property (in the sense of a mesurable characteristic of an entity or a situation) and attribute (in the sense of a measure of a property) are completely mixed. Then, we grouped all the WordNet top-level concept types relative to these notions under the concept type Dimension_or_Measure, and we also offered specific concept types for each one of them. The user had to specialize these specific concept types with the WordNet types he uses, if he wants to use relations which are signed on these specific concept types (for example, the signature of the relation CHRC is Concept->Property, whereas the signature of the relation ATTR is Concept->Attribute). The concept types Time and Space are also subtypes of Dimension_or_Measure. We have subtyped Space by Physical_Entity in order to let spatial relations use a physical entity as a spatial region (this simplify a lot the building and display of CGs). But then, if we had define Dimension_or_Measure as a subtype of Entity, Physical_Entity would not have been a direct subtype of Entity, and it would not have been visible in Figure 1. For ergomic reasons we wanted to avoid that, thereby we have defined Dimension_or_Measure as a direct subtype of Concept.

Like Sowa (1992), we have subtyped Entity by Proposition and Representation_Entity (under this type, we have synthesized or structured the main abstract data types used in computer sciences). We have also added the concept type Collection and specialized it with 1) the top-level WordNet concept types about groups or sets, 2) the set type hierarchy proposed by Pfeiffer and Hartley (1992), 3) the abstract data types which collections (these types are also subtypes of Representation_Entity). Besides, Linear_ADT (a type which groups linear abstract data types like Character or Number) is a subtype of Linear_Dimension_or_ADT which is also a supertype for Time and Space; therefore, mathematical relations signed on Linear_Dimension_or_ADT (e.g. Equal and LessThanOrEqual) may be used between numbers as well as between concepts of time and space.

---

1. Any process (that is a potential causator of changes) may be viewed as a state (that is, according to Sowa, a situation that remains unchanged during a given period of time) if the period of time is sufficiently short. For example, «a rock is rolling» may be represented by: [State]->(Descr)->[Proposition: [Rock]<-(Object)<-[Roll]].

2. In order to understand why we classed such WordNet types as subtypes of State and Process, think that they occur in some region of Time and Space, see in Section 6 the list of the relations that can olny be attached to processes, and see in Figure 2 some examples of subtypes of W_relation/n (the types related to the first meaning for «line» shown in Figure 2) and of W_psychological_feature (the types related to the second meaning for «line» shown in Figure 2). W_communication/n, subtype of W_relation/n in WordNet, is now a subtype of Proposition. W_communication/n is a supertype of W_proposition (and hence of W_theorem/n, W_conclusion/n1, etc.), W_message/n, W_hypothesis/n, etc.

Also like Sowa (1992), we have subtyped Process by Event (more exactly W_event[1]). Sowa doesn't define what an event is, but from his explanations, it may be infered that a process is considered to be an event for a given period of time when it makes a change during this period. If it doesn't make a change during this period, it may be considered as state. Hence an event cannot be a state. But in his type hierarchy, Sowa (1992) defines Action as a subtype of Event, thereby forbidding that an action may be viewed as a state. In order to avoid this, we have defined W_act__human_action__human_activity as a direct subtype of Process. Then, according to the application expertise, the knowledge engineer may use a process as an event or as a state, and if he needs to, he may class some types of process as subtypes of W-event or subtypes of State. Since CGKAT is aimed to be a KA tool, we have also subtyped Process by some types of problem solving tasks we have collected in the KA literature concerning KADS, e.g. (Wielinga & al., 1992).

These ontological distinctions may appears obvious but we have often noted that even when these distinctions are clearly stated and used, knowledge engineers make semantic errors when they represent knowledge. For example, they try to connect «relations for processes» to concepts of type State (e.g. the relation Agent instead of using the relation Consequence or Succ), they define common subtypes to exclusive concept types (e.g. Observation as a subtype of Process and Proposition (hence they confuse the action for its results)), and use types instead of others (e.g. Redness which is a subtype of Color, instead of Red which is a subtype of IsColored). In CGKAT, these problems are much reduced because: 1) all relations types have a signature which is checked when a relation is added; 2) we have implemented exclusion relations between types, and we have defined some in our top-level ontology (the most important or helpful is the exclusion between entities (e.g. propositions) and situations (e.g. processes)).

5.2 Structuring with the notion of role

Finally, we introduced in our top-level ontology, a distinction which is orthogonal to the previous ones, that is the notion of «role» (see in Figure 1 the type Concept_Playing_a_Role and its subtypes). A role type expresses the roles that an individual can play. For example, an entity may be the cause, the agent or the result of a process (examples of agent roles are «taxi driver» or «musician»). We give below an extract of our typology for the roles of entities (some of its types, e.g. Causal_Entity, are necessary for giving a signature to some relation types, e.g. Agent). WordNet does not make distinctions between «natural types» and «role types», thereby we have to let the user make the distinctions when he needs it (at least our top-level ontology offers the framework for that). Some direct subtypes of Concept_Playing_a_Role are Property[2], Attribute, Concept_Used_by_a_Process,

1. The subtypes of this WordNet top-level concept type correspond to the notion of what is an event for us. We have just changed the comment given by WordNet to this type.

2. Being a property or an attribute of another concept is clearly a role type. In Sowa(1992), property types are second order types and attribute types are instances of property types. We could not follow Sowa in that direction since: 1) properties and attributes (in the sense defined above) are not distinguished in WordNet; 2) properties are (first order) role types; 3) second-order concepts cannot be connected to the same relations as the first order concept since a relation type has only one signature. All other ontological model we have seen use the Kind-Of relation between property types and attribute type, e.g. the Situation Data Model of Tepfenhart (1992). Our choice implies that an attribute type (e.g. Red) cannot be in the referent part of a concept with a property type (e.g. Color) except if this may be interpreted as a shortcut for a relation «Value» between this concept and a generic concept with this attribute type (e.g. [Color: Red] would be expansed in [Color]->(Value)->[Red]).

Concept_Used_in_an_Application and Concept_Known_by_Someone. This last type is useful for example in KA from multiple experts or with many knowledge engineers. The type Concept_Used_in_an_Application is especially useful when the concept types from WordNet are used, since it enables the knowledge engineer to focus on the types of its application (even if each of these types are also subtypes of WordNet concept types).

```
Entity_Playing_a_Role
   Causal_Entity -- any entity that can cause a process
      Goal-directed_Entity -- Problem Solver or interactional agent Entity
         Conscious_Goal-directed_Entity -- e.g. a person
         Non_Conscious_Goal-directed_Entity -- e.g. an AI agent
      Perhaps_Goal-directed_Entity -- e.g. supernatural forces
      Without_Goal_Entity -- non conscious Entity and not an AI_Agent
   Input_Entity -- input of a process
   Output_Entity -- output of a process
   Recipient_Entity -- recipient of a process
   Patient_Entity -- the object of a process, e.g. W_subject__content__depicted object
   W_necessity__essential__requirement__requisite__need -- anything needed
   W_inessential -- anything that is not essential
   Possessed_Entity -- e.g. a Pet, W_possesion/n
   Part_Entity
      W_part__portion -- something determined in relation to something that includes it
      W_part__piece -- a portion of a natural object
      W_unit__building_block -- an undivided entity occurring in the composition of something
   Whole_Entity
      W_whole -- all of something including all its component elements or parts
      W_whole__whole_thing__unit -- a single undivided entity
      W_unit/n -- an organization regarded as part of a larger social group
   Representation_Container -- e.g. a text or audio file
   Model_ADT -- a representation entity which is a model, e.g. KADS_Model
```

We said in the introduction that WordNet organizes the concepts which nouns refer to in ten exclusive concept categories. The concept types for these categories are: W_entity/n, W_abstraction/n, W_group__grouping, W_state/n, W_psychological_feature/n, W_event/n, W_phenomenon/n, W_act__human_action__human_activity, W_location/n, W_possession/n. For building our top-level ontology, we had to dispatch the 13 subtypes of W_entity (under Entity and Entity_Playing_a_Role) and the 7 first subtypes of W_abstraction (under Dimension_or_Measure and State). Hence, we have merged/subordinated 28 top-level WordNet concept types into our top-level ontology. We have also subtyped Entity_Playing_a_Role with some deeper WordNet concept types in order to give pointers to some role types hidden in WordNet. Presently, our top-level ontology includes about 200 concept types.

WordNet has a very long list of top-level synsets where the synonyms are verbs. Therefore, we have not merged/subordinated the concept types corresponding to these synsets into our top-level ontology. The CGKAT user will have to place these types himself if he wants to use them instead or in complement of the types which come from the synsets composed of nouns for actions. Similarly, we have not classed concept types relative to notions expressed by adjectives or adverbs since WordNet does not offer typologies for them, but the user may include them in the concept type lattice (by default, they will be placed under Concept).

6 The relation type hierarchy

We call a relation *a basic relation* when it cannot be defined using one or more concepts and other basic relations which are different from the primitive relation LINK. For example, case relations (also called thematic relations) like Agent, Object and Recipient, are basic relations. We call a relation which is not basic, a complex relation. We have shown in (Martin 1995) that relations should remain basic for many reasons among which: 1) the concept type hierarchy is duplicated in the relation type hierarchy when complex relation types are introduced in it; 2) if a complex relation has no definition (with concepts), a conceptual graph (CG) using this relation cannot be expansed and then a lot of graph-matching possibilities are lost; 3) a complex relation which has a definition enables to hide some more basic relations but cannot be used when other basic relations must be added; 4) hidding more basic relations leads to ambiguities, and induces problems for graph-matching. Afterwards, in (Martin, 1995), we have presented a top-level ontology for 200 basic relation types gathered from the hypertext and knowledge representation literature.

WordNet doesn't provide basic relation types, e.g. Part, Purpose and Method, it includes concept types which express roles, e.g. W_part__portion, W_intention__purpose, W_wise__method. And these roles are not grouped (we have done it partly), hence WordNet is of no use for building a relation type hierarchy.

A hierarchy of basic relation types is a great help in KA and KR: 1) it provides cues for representing knowledge or searching it in a CG base; 2) the signatures of the relation types enforce a minimal coherence in knowledge representation (during the buiding of CGs but also during the building of the concept type lattice: our relation type hierarchy was a great guide for enhancing the coherence and completeness of our top-level concept types ontology !); 3) these signatures may also be used for elicitation or knowledge collect: from a concept in a CG, all the types of the relations which may be connected to this concept, may be listed (CGKAT presents this list in hierarchical order, as for any part of the relation type hierarchy; however, since our relation types are signed on top-level concept types, the choice is often large).

7 Conclusion

In order to help the knowledge engineer to build its concept type lattice and represent knowledge with CGs, CGKAT offers him a top-level concept type ontology which includes many conceptual distinctions necessary for KR and KA and under which the WordNet ontology may be dynamically included. This top-level ontology is not obligatory for accessing the WordNet concept types but it gives them a structuration which is useful for KR. We especially think about concept types on which basic relation types are signed, and to the structuration of role types. However, in many cases we could not gather and group all the WordNet concept types relative to a notion, and include them into our top-level ontology (it would not have included about 200 concept types but much more than 10.000 (we have tried !)). Therefore, in those cases the user has to subtype himself the ontological distinctions he find interesting, with the types he uses in his application. CGKAT also offers a top-level ontology for relations for guiding and saving the knowledge engineer work in KR and KA.

In these top-level ontologies, we included the ontological distinctions made by WordNet, Sowa (1992), Sowa (1984), Pfeiffer & Hartley (1992) and also most of the distinctions made by Tepfenhart (1992), Esch (1992), the PENMAN Upper Model (Bateman 1990) and CYC

(Lenat & Guha, 1990). An interesting work would be to pursue this work with other top-level ontologies, e.g. ONTOS (Carlson & Nirenburg, 1990). Another one would be to exploit the other relations provided by WordNet, e.g. «Part-of» and «Cause-of».

8 Acknowledgements

The author thanks its referees and Dr Jurgen Mueller for their comments on previous versions of this article.

9 References

Bateman J. (1990). *Upper modeling: Organizing knowledge for natural language processing*. In Proc. of Fifth International Workshop on Natural Language Generation, Pittsburgh, PA.

Barwise J. & Perry J. (1983). *Situation and Attitudes*. MIT Press, Cambridge, MA.

Carlson L, & Nirenburg S. (1990). World modeling for NLP. Technical Report CMU-CMT-90-121, Center for Machine Translation, Carnegie Mellon University.

Chein M. & Leclère M. (1993). *A cooperative program for the construction of a concept type lattice*. Research report No 93075 of LIRMM (Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier - France - fax : (33) 67 41 85 00), 1993.

Esch J.W. (1992). *Temporal Intervals*. In Conceptual Structures: current research and practice (Eds: Nagle T.E., Nagle J.A., Gerholz L.L. & Eklund P.W.), England , Ellis Horwood Workshops, 1992.

Knight K. & Luk S. (1994). *Building a Large-Scale Knowledge Base for Machine Translation*. In Proc. of AAAI'94, twelfth national conference on artificial intelligence, July 1994.

Haemmerlé O. (1995). *CoGITo: une plate-forme de développement de logiciels sur les graphes conceptuels*. Ph.D thesis, Université Montpellier II, France, January 1995.

Lenat D.B. & Lenat R.V.G. (1990). *Building large knowledge-based systems: representation and inference in the Cyc project*. Reading, MA; Sydney; Tokyo:Addison-Wesley, 1990.

Martin Ph. (1995). *Knowledge Acquisition Using Documents, Conceptual Graphs and a Semantically Structured Dictionary*. Proc. of KAW'95, ninth Knowledge Acquisition for Knowledge-Based Systems Workshop, Gaines, B.R. Eds, University of Calgary, Banff, Canada, 1995.

Miller G.A., Beckwith R., Fellbaum C., Gross D. & Miller K. (1990). *Five Papers on WordNet*. CSL Report 43, Cognitive Science Laboratory, Princetown University, July 1990. (These papers and the system are available by anonymous ftp at clarity.princeton.edu, subdirectory 'pub').

Pfeiffer H.D. & Hartley R.T. (1992). *The Conceptual Programming Environment, CP*. In Conceptual Structures: current research and practice (Eds: Nagle T.E., Nagle J.A., Gerholz L.L. & Eklund P.W.), England , Ellis Horwood Workshops, 1992.

Sowa J.F. (1984). *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley, Reading, MA.

Sowa J.F. (1992). *Conceptual Graphs Summary*. In Conceptual Structures: current research and practice (editors: Nagle, T.E., Nagle, J.A., Gerholz, L.L., and Eklund, P.W.), England , Ellis Horwood Workshops, 1992.

Sowa J.F. (1993). *Relating Diagrams to Logic*. In Proc of ICCS'93, first international conference on conceptual structures (Eds: G.W. Mineau, B. Moulin, J.F. Sowa), Quebec City, Canada, August 1993.

Tepfenhart W.M. (1992). *Using the Situation Data Model to Construct a Conceptual Basis Set*. In Conceptual Structures: current research and practice (Eds: Nagle T.E., Nagle J.A., Gerholz L.L. & Eklund P.W.), England , Ellis Horwood Workshops, 1992.

Wielinga B., Schreiber G. & Breuker J. (1992). *KADS: a modelling approach to knowledge engineering*. In Knowledge Acquisition (1992) 4, pp 136-145.