A Lexico-semantic Approach to the Structuring of Terminology

Marie-Claude L'HOMME
OLST – Université de Montréal
C.P. 6128, succ. Centre-ville
Montréal (Québec), Canada H3C 3J7
Marie-Claude.L'Homme@umontreal.ca
http://www.olst.umontreal.ca

Abstract

This paper discusses a number of implications of using either a conceptual approach or a lexico-semantic approach to terminology structuring, especially for interpreting data supplied by corpora for the purpose of building specialized dictionaries. A simple example, i.e., *program*, will serve as a basis for showing how relationships between terms are captured in both approaches. My aim is to demonstrate that truly conceptual approaches do not allow a flexible integration of terms and relationships between terms and that lexico-semantic approaches are more compatible with data gathered from corpora. I will also discuss some of the implications these approaches have for computational terminology and other corpus-based terminological endeavours.

1 Introduction

Recent literature in terminology circles constantly reminds us that methods and practices have changed drastically due mostly to the extensive use of electronic corpora and computer applications. What might appear as normal and standard in computational circles has had profound consequences for terminologists; this has led many to criticize traditional theoretical principles and some to propose new approaches (Bourigault and Slodzian 1999; Cabré, 2003, among others; see L'Homme et al., 2003 for a review).

One of the issues at the centre of this debate is that of diverging views on the relationship between the term and the abstract entity it is supposed to represent (a "concept" or a "meaning"). Differing views will inevitably lead to very different ways of envisaging terms and methods of structuring them. Some might be compatible with a given application, while others are much more difficult to accommodate.

In this paper, I will try to demonstrate some of the methodological consequences of adopting a conceptual approach or a lexico-semantic approach to terminology structuring. These observations are drawn from my experience in compiling specialized dictionaries using corpora as primary sources and computer applications to exploit them.

Even though the application I am familiar with is very specific and obviously influences my view on the structuring of terms, I believe this topic is also relevant for other terminology-related applications. For example, in computational terminology, there is an increasing interest for structuring extracted terms (articles in Daille et al., 2004 and in Nazarenko and Hamon, 2002, among others). Automatic term structuration is carried out by considering morphological variants (Daille, 2001; Grabar and Zweigenbaum, 2004), performing distributional analysis to build classes of semantically related terms (Nazarenko et al., 2001, among others), or acquiring other types of linguistic units, such as collocations or verbal phrases, from specialized corpora.

These questions will be addressed from a linguistic point of view, but many have been dealt with directly or indirectly by computational terminologists and, in fact, are often raised by their work on specialized corpora. I will also try to demonstrate that the problems dealt with in this paper are by no means a reflection of a tendency often attributed to linguists to make things more complicated than they actually are. I would like to show that they are a reflection of the functioning of terms in running text.

2 Two different approaches to terminology

The conceptual approach I describe is the one advocated by the Vienna School of terminology that has been and is still applied to work carried out by terminologists. The results of its analyses is encoded in term records in term banks or in articles in terminological dictionaries.

The lexico-semantic approach on which my discussion is based is the Explanatory and Combinatorial Lexicology (ECL) (Mel'èuk et al., 1995; Mel'èuk et al. 1984-1999) which is the lexicological component of the Meaning-text Theory (MTT). As will be seen further, ECL provides an apparatus, namely lexical functions (LFs), that can capture a wide variety of semantic relations between lexical units. ECL descriptions are encoded in an Explanatory and Combinatorial Dictionary (ECD) (Mel'èuk et al. 1984-1999).

In order to illustrate the methodological consequences of the two approaches under consideration, I will use a basic term in the field of computing, i.e., *program*. This term was chosen because no one will question its status in computing no matter what his or her view is on terms and terminology.

In addition, like many basic terms, *program* is polysemic, ambiguous in some contexts, and semantically related to several other terms. It will be very useful to show the variety of semantic relationships in which terminological units participate. Finally, *program* does not refer to a concrete object. Hence, its analysis will pose problems different from those raised by terms like *printer* or *computer*.

I will also frequently refer to a corpus from which my observations are derived. This corpus contains over 53 different texts and amounts to 600,000 words. It was compiled by the terminology team within the group *Observatoire de linguistique Sens-Texte* (OLST) in Montreal. Since I am not an expert in computer science, I must rely – like other terminologists – on information provided in a corpus and not on previous knowledge to analyze the meaning of *program* and the other terms to which it is related.

2.1 A conceptual approach to the processing of the term *program*

When considering a unit such a *program*, terminologists who adhere to a conceptual approach will define its place within a conceptual structure. This is done by considering its characteristics (in fact, often by deciding which ones are relevant), and by analyzing classical relationships, such as hyperonymy (or, rather, generic-specific) and meronymy (or whole-part). In order to achieve this, terminologists usually gather information from reliable corpora.

The corpus first informs us that "program" can be subdivided into in one of the following categories; 1. "operating system"; 2. "application software", i.e., "word processor", "spreadsheet", "desktop publishing software", "browser", etc.; and 3. "utility program". It also tells us that there are different types of "programs": 1. "shareware programs", "freeware programs"; "educational programs"; and "commercial programs"; 2. "command-driven programs" and "menu-driven programs".

One possible representation of these relationships has been reproduced in Figure 1. Of course, my interpretation of the data listed above is simplified, since it does not take into account all the relationships that can be inferred from it (e.g., the fact that software programs or educational programs can be menu-driven). Also, part-whole relationships for some of these subdivisions can be identifed (e.g., the fact that programs – classified according to the interface – have parts such as menus, windows, buttons, options, etc.).

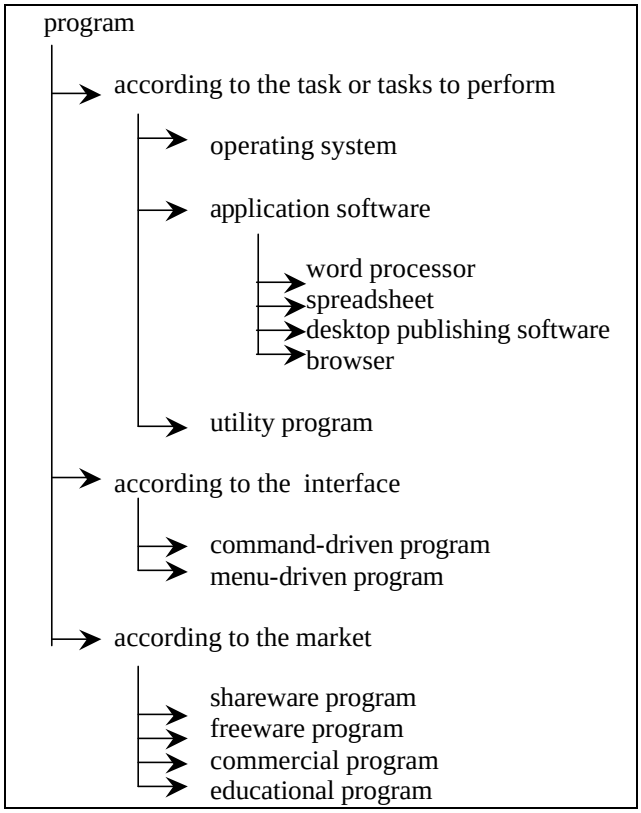

Figure 1: Representation of the relationships between "program" and related concepts

For the time being, I will assume that I have solved the problems related to the relations between "program" and other relevant concepts (which, in fact, is not the case, as we will see below).

The corpus also allows me to observe that the concept I am currently dealing with, has different names: *program* and *software program*. This will normally be dealt with in conceptual

representations by taking for granted that all these different linguistic forms refer to the same concept, and thus are true synonyms. In my representation, they will be attached to the same node as "program" (see Figure 2).[1]

Furthermore, since concepts and conceptual representations are considered to be language-independent, their description and representation should be valid for all languages. Hence, my representation system should apply to French (and to true synonyms in French) and other languages (see Figure 2).

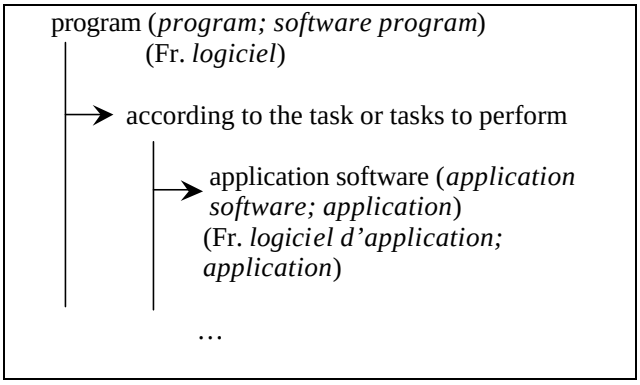

Figure 2: Synonyms in conceptual representations

Regarding this last issue, a choice must often be made between several potential synonyms in order to select a single identifier for a concept. This choice can simply be functional (allowing the labelling of a node in a representation such as that in Figure 1) or result from standardizing efforts. The choice of a unique identifier is central in conceptual analyses, since relationships are defined first and foremost between concepts and are considered to be valid for the linguistic forms that label them.

2.2 Other issues related to the analysis of *program*

In my discussion on the processing of *program*, I deliberately avoided other important issues revealed by the data contained in the corpus. We will look at some of these issues in this section.

First, "programs" can be further classified according to the language used create them ("C programs", "C++ programs", "Java programs"), or according to the hardware device they manage ("BIOS program", "boot program"). Incidentally, in French, the first subdivision (the one represented in section 2.1) corresponds to *logiciel*. The ones we just introduced are named *programme*.

This obviously has consequences for the representation of *program* produced above. The problem can be solved in conceptual approaches by:

a. Considering that *program* refers to a single concept, and trying to account for the different ways of organizing its relationships with other concepts with new conceptual subdivisions. This will produce a very complex, yet possible, graphical representation;

b. Focussing on a single organization of the concept "program" (for example, the one chosen in section 2.1.) and defining the others as being related to vague or improper uses of *program*; or, finally,

c. Saying that *program* is associated with two or three different concepts, and possibly classifying them into three different subfields of computing, i.e., concept1 = micro-computing; concept2 = programming; concept3 = hardware. If the description is carried out in a multilingual context, the subdivision will be necessary to account for the fact that, in French, for instance, *program* can be translated by *logiciel* or *programme*. This latter choice is the one that is closest to the distinctions made with the lexico-semantic approach dealt with in the following section.

Secondly, *program* shares with other lexical units many other different semantic relationships other than the taxonomic and meronymic relations previously considered. All the relationships listed below have been found in the corpus.[2]

- Relationships that involve activities and that are expressed linguistically mostly by collocates of *program*:
  Function: *a program performs tasks*
  Creation: *development, creation of a program, programming*
  Actions that can be carried out on programs: *configuration, installation, running, aborting*, etc.

[1]Large-scale ontologies represent concepts and lexical forms using a similar strategy. For example, the Unified Medical Language System (UMLS) (National Library of Medicine, 2004) makes a clear separation between a Semantic Network and a Lexicon.

[2]Some of these have been listed in Sager (1990) who argued that a large variety of conceptual relationships could be found in specialized subject fields.

- Relationships that involve properties and that are also expressed linguistically by collocates of *program*:
  powerful program, user-friendly program; feature of a program
- Argument or circumstantial relationships:
  Agent: *user of a program*; *programmer*
  Instrument: *create a program with a language*
  Location: *install the program on the hard disk, on the computer*
- Other relationships expressed by morphological derivatives terms that include the meaning of *program*;
  programming, programmable, reprogrammable

Most relationships listed above are non-hierarchical and may be expressed by parts of speech other than nouns. Consider, for example, actions that can be performed on a program (*configuration, configure; install; installation*, etc.). [3] Some will be very difficult to account for in terms of conceptual representations. Of course, conceptual-approach advocates might argue that these relationships are not relevant for terminology.

Thirdly, in my discussion of the fact that concepts could have different names, I mentioned only a synonym, but concepts are expressed in a variety of forms in corpora. Many of these will not take the form of nouns.

2.3 A lexico-semantic approach

In this section, I repeat my analysis of *program* this time using a lexico-semantic approach. This approach is also based on data gathered from corpora. The discussion presented in this section is summarized in Table 1.

First, the analysis of *program* in the corpus reveals that it has three different meanings. *Program* can be defined as: 1) a set of instructions written by a programmer in a given programming language in order to solve a problem (this meaning is also conveyed by *computer program*); 2) a set of programs (in sense 1) a user installs and runs on his computer to perform a number of tasks (this meaning being also conveyed by *software program*); and 3) a small set of instructions designed to run a specific piece of hardware.

[3] Another non-hierarchical relationship has received a lot of attention recently, that of cause-effect.

This sense distinction is validated by the fact that *program* can be related to different series of lexical units.

For example, a *program*$_1$ is something that someone, called a *programmer*, *writes*, *executes*, *compiles* and *debugs*. It can be *machine-readable* or *human-readable*. It can also *end* or *terminate*. *Program* can be modified by names given to languages, i.e., *C program, C++ program, Java program*. Finally, it can also have parts such as *modules, routines*, and *instructions*.

Program$_1$	
Explanation	Set of instructions written by a programmer in a programming language to solve a specific problem
Collocates	*write ~; compile ~, execute ~; create ~; machine-readable ~; human-readable ~; ~ ends, ~ terminates, debug ~; powerful ~*
Hyponyms	*C ~, C++ ~, Java ~*
Other related terms	*to program; programming, programmer; routine, instruction; module; page; segment; language; line*
Program$_2$	
Explanation	Set of programs$_1$ installed and run on the computer by a user to perform a specific task or a set of related tasks.
Hyponyms	*operating system; application software; word processor, spreadsheet*
Collocates	*active ~, running of ~; download ~; develop ~; run ~, install ~; uninstall ~; add/remove ~; user-friendly; quit ~; exit ~; load ~; launch ~*
Other related terms	*user, hard disk; application software*
Program$_3$	
Explanation	Short set of specific instructions designed to run a hardware device
Other related terms	*boot, BIOS, to program, reprogram, programmable, reprogrammable, programming*

Table 1: Semantic distinctions for *program*

A *program*$_2$ is something a *user installs* on his computer, *loads* into the memory, *runs*, and sometimes *uninstalls*. Different sorts of programs can be identified, such as operating systems, applications, and utilities. Programs can have parts such as windows, menus, options, etc. Finally, a *program*$_2$ can be *user-friendly*.

A *program*$_3$ consists of a few code lines written in order to specify the behaviour of a specific hardware device, such as a memory. The device is then said to be *programmable* and/or *reprogrammable*. It can be *programmed* and *reprogrammed*.

In this lexico-semantic approach, the relationships observed between *program* and other terms are attached to its specific meanings. This distinction allows us to relate other terms to specific senses. For example, *program*$_1$ is related to other senses as follows:

Synonym: *computer* ~
Types of programs: C ~, Java ~
Parts of programs: *instruction, page, segment, line, routine*
Creation of a program: *write ~, create ~, to program, programming*
Agent: *programmer*
Cause a program to function: *execute* ~
The program stops functioning: ~ ends, ~ terminates
etc.

Since most semantic relationships are non-hierarchical, they can be represented in a relational model. In ECL, paradigmatic and syntagmatic semantic relations are represented by means of a single formalism, i.e., lexical functions (LFs). LFs are used to capture abstract and general senses that remain valid for a large number of lexical units. The relationships listed above could be formalized as follows:[4]

synonym: **Syn**(program$_1$) = computer ~

agent of a program: **S$_1$**(program$_1$) = programmer

create a program: **CausFunc**$_0$(program$_1$) = create [DET ~], write [DET ~]

Cause a program to function: **CausFact$_0$**(program1) = execute [DET ~]

The program stops functioning: **FinFact$_0$**(program$_1$) = [DET ~] ends, [DET ~] terminates

[4]Meronymic and hyperonymic relationships can also can also be captured by means of lexical functions. Authors have proposed LFs especially designed to represent these relations (**Spec,** for hyponymy; and **Part**; for meronymy). However, ECL will prefer accounting for these relationships with non-standard lexical functions in order to explain the specific nature of the relationships between a lexical unit and its meronym.

3 General comments on the analyses of terms

These two brief analyses of *program* reveal the following about terms:

- Terms can convey multiple meanings. This is not an accidental property that only affects *program*. Numerous examples can be found in corpora and have been dealt with in recent literature. This, of course, has important consequences for both conceptual and lexico-semantic approaches.
- Terms can enter into a large variety of relationships with other terms, and not only taxonomic or meronymic relationships. The understanding of these relationships is necessary to capture sense distinctions; in addition, relationships are valid for a specific meaning.
- Some of the relationships observed between terms are hierarchical: hyperonymy and meronymy.
- Most semantic relationships are non-hierarchical: e.g., actions carried out by terms, properties, cause-effect.
- Some relationships involve lexical units other than nouns: e.g., actions and creation are often expressed linguistically by means of verbs; properties are expressed by adjectives.
- Most relationships involve terms considered as linguistics units rather than labels for concepts: e.g., morphological derivatives.

In fact, what these observations tend to show is that terms behave like other lexical units and must be dealt with accordingly. Terms will acquire their specificity through a given application with set objectives, but as units occurring in corpora, terms cannot be differentiated from other lexical units.

4 Implications for computational terminology and other corpus-based work

The previous discussion has a number of implications for computational terminology (as well as other corpus-based terminology-related applications). I will examine a few in this section.

First, both approaches will focus on different types of units when selecting terms in corpora. In conceptual approaches, a selection is made among linguistic units that can refer to a concept. The focus is on nouns and noun phrases. Even though concepts can be expressed in a variety of linguistic forms, synonyms considered will invariably be nouns of noun phrases. Work on terminological

variation (Daille, 2003; Jacquemin, 2001) has shown the variety of forms that terms can take in corpora (morphological derivation, insertion, elision, anaphora, etc.), but these are taken into account only if they can be associated with an admitted term.

In a lexico-semantic approach as that presented in section 2.3, units considered will be those that convey a meaning that can be related to the field of computing (the subject field is delimited prior to the selection). Lexical units selected can pertain to different parts of speech as long as their meaning can be related to the field under examination: nouns (*program, byte*); verbs (*debug, to program*), adjectives (*user-friendly, programmable*). Even adverbs can convey a specialized meaning (e.g., *digitally, dynamically*).

Secondly, any terminological work based on corpora will run into polysemy, even though it focuses on a small set of terms. The manner in which the distinctions between senses are made has important consequences on way terms will be processed afterwards.

Polysemy can be dealt with using a conceptual approach, which considers this property to be an accidental problem. Hence, distinctions depend on decisions made during the classification process or the construction of conceptual representations.

In lexico-semantic approaches, polysemy is viewed as a natural property of lexical units. Senses are delimited prior to the representation of semantic relationships and this delimitation is based on the observations of interactions between the term under examination and other lexical units. Sense delimitation and distinction is a necessary step before anything else can be done.

Thirdly, regarding terminology structuring, conceptual methods, such as the one discussed in section 2.1, are useful as far as classification is concerned. Hence, they can be used for describing concepts that correspond to entities (concrete objects, substances, artefacts, animates, etc.). Moreover, the focus is on hierarchical relations (hyperonymy and meronymy) which is again valid for entities, and, as far as part-whole relations are concerned, more specifically concrete objects. Many non-hierarchical relationships, such as those listed in section 2.3 are disregarded, either because they involve units that do not refer to entities, or because they are relationships between lexical units and not concepts.

Also, relationships between synonyms are considered from the point of view of true synonymy. Choosing a unique linguistic identifier for a concept and considering competing linguistic forms as true synonyms has implications for the variety of relationships that can be considered. Some relationships can be valid for one synonym but not for another.

In lexico-semantic approaches, semantic relationships are attached to senses that have been distinguished previously. In addition, a wide variety of semantic relationships can be taken into account. These relationships can apply to terms that designate entities, as well as activities, and properties. Hypernymy and meronymy represent only a small part of the semantic relationships terms can share with other terms. Other relationships, such as argument relations, entity-activity relations, can be expressed by different parts of speech.

Fourthly, conceptual approaches lead to representations that distance themselves from data collected in corpora. Many decisions are made during the construction of the representation. On the one hand, many meanings that would appear to be relevant in other approaches are not considered. On the other hand, things are added in order to build the representation. Consider, for example, Figure 1. Some subdivisions are created but do not correspond to lexical units (e.g., *according to the interface*); this sort of classification of units will result in considering several complex sequences that have a compositional meaning (hence, that are not true lexical units).

Terminology structuring in conceptual approaches is often carried out in order to represent knowledge and not linguistic units. Problems arise when this work is done using corpora as a starting point, since linguistic units (such as terms) do not behave in a way that reflects perfectly a given knowledge structure. When analyzing terms, considerations regarding knowledge structure will constantly interfere with factors related to the behaviour of linguistic units in text.

On the other hand, lexico-semantic approaches are much more compatible with data gathered from corpora. Of course, terminologists will make decisions since they must interpret data and synthesize their findings, but these are based on the observation of interactions between lexical units that appear in corpora.

5 Concluding remarks

The point in my discussion, is not to say that an approach is much better than the other for terminology, regardless of the application at hand. This topic has been dealt with extensively by authors and even placed in a theoretical

perspective. Rather, I wanted to demonstrate that an approach is probably better suited that the other as far as terms considered in corpora are concerned. I also wanted to point out the methodological consequences of choosing an approach over another.

Conceptual approaches will account for consensual representations of knowledge, based on a predefined set of hierarchical relationships. However, in must be kept in mind that resulting representations distance themselves from corpus data and necessitate a lot of hand-crafted changes. Often, the ideal knowledge structure is formulated beforehand entirely or partly, and the difficulty consists in trying to find lexical units that fit into it.

Lexico-semantic approaches will provide terminologists with a framework for interpreting data related to terms and the contexts in which they appear. However, one must accept, when using this kind of approach, that terminological structures are discovered gradually through semantic relations and that some of these relations will even contradict assumed knowledge structures.

6 Acknowledgements

I would like to thank Elizabeth Marshman for her comments on a preliminary version of this article.

References

D. Bourigault and M. Slodzian. 1999. Pour une terminologie textuelle. *Terminologies nouvelles*, 19:29-32.

M.T. Cabré. 2003. Theories of Terminology. Their Description, Prescription and Explanation. *Terminology* 9(2):163-199.

A. Condamines. 1995. "Terminology. New needs, new perspectives." *Terminology* 2(2): 219-238.

D.A. Cruse. 1986. *Lexical Semantics*, Cambridge: Cambridge University Press.

B. Daille. 2002. Qualitative term extraction. In D. Bourigault, C. Jacquemin and M.C. L'Homme (eds.), *Recent Advances in Computational Terminology*, 149-166, Amsterdam / Philadelphia: John Benjamins.

B. Daille. 2003. Terminology Mining. In M.T. Pazienza (ed.), *Information Extraction in the Web Era*, Lectures Notes in Artificial Intelligence. 29-44. Springer.

B. Daille, K. Kageura, H. Nakagawa and L.-F. Chien (eds.). 2004. *Recent Trends in Computational Terminology*. *Special Issue of Terminology*, 10(1).

N. Grabar and P. Zweigenbaum. 2004, forthcoming. Lexically-based terminology structuring. *Terminology*, 10(1).

Jacquemin. C. 2001. *Spotting and Discovering Terms through Natural Language Processing Techniques*, Cambridge: MIT Press.

M.C. L'Homme, U. Heid and J.C. Sager. 2003. Terminology during the past decade (1994-2004). An Editorial Statement. *Terminology*, 9(2):151-161.

I. Mel'èuk, A. Clas and A. Polguère. 1995. *Introduction à la lexicologie explicative et combinatoire*, Louvain-la-Neuve (Belgique): Duculot.

I. Mel'èuk et al. 1984-1999. *Dictionnaire explicatif et combinatoire du français contemporain. Recherches lexico-sémantiques I-IV*, Montréal : Les Presses de l'Université de Montréal.

National Library of Medicine. 2004. UMLS Knowledge Sources (http://www.nlm.nih.gov/research/umls/)

A. Nazarenko and T. Hamon. (eds.). 2002. *Structuration de terminologie. Special issue of Traitement automatique des langues. TAL*, 43(1).

A. Nazarenko, P. Zweigenbaum, B. Habert and J. Bouaud. 2001. Corpus-based extension of a terminological semantic lexicon. In D. Bourigault, C. Jacquemin and M.C. L'Homme (eds.), *Recent Advances in Computational Terminology*, 327-351, Amsterdam / Philadelphia: John Benjamins.

J.C. Sager. 1990. *A Practical Course in Terminology Processing*. Amsterdam / Philadelphia: John Benjamins.

E. Wüster. 2004, forthcoming. The structure of the linguistic world of concepts and its representation in dictionaries [translated by J.C. Sager]. *Terminology*, 10(2).