Ontology Development for Machine Translation: Ideology and Methodology

Kavi Mahesh

MCCS-96-292

Computing Research Laboratory
New Mexico State University

Contents

List of Figures

Ontology Development for MT: Ideology and Methodology

Kavi Mahesh

Computing Research Laboratory
New Mexico State University
Las Cruces, NM 88003-8001
Ph: (505) 646-5466 Fax: (505) 646-6218
mahesh@crl.nmsu.edu

Abstract

In the Mikrokosmos approach to knowledge-based machine translation, lexical representation of word meanings as well as text meaning representation is grounded in a broad-coverage ontology of the world. We have developed such a language-neutral ontology for the purpose of machine translation in the situation of the Mikrokosmos project. In order to acquire a fairly large ontology with limited time and manpower resources, a number of problems in knowledge representation, natural language semantics, and software technology were solved by taking a practical, situated approach which does not always produce formally clean or provably correct solutions. The ontology thus developed has already proven to be of great value in almost every stage of natural language processing and linguistic knowledge acquisition and has potential applications in many other fields as well. In this report, we describe the Mikrokosmos ontology and the ideology behind it and present some of the practical solutions adopted to solve a range of problems. We also present concrete details of the methodology for concept acquisition that evolved from our exercises in machine translation.

1 Introduction to the Mikrokosmos Ontology

This report documents the methodology we followed in developing a broad-coverage ontology for use primarily in natural language processing (NLP) and machine translation (MT). The report describes our product, the Mikrokosmos (μK) ontology, and addresses a number of issues that arose during its acquisition. A common thread running through our solutions to the many problems in ontological engineering is a strong preference for submitting to practical considerations as dictated by the entire situation of our MT project. The technology we use for ontology development as well as a detailed set of guidelines and sample scenarios for concept acquisition are documented in a separate report (Mahesh and Wilson, forthcoming). Extensive documentation on the μK ontology is also available on the World Wide Web at the URL http://crl.nmsu.edu/users/mahesh/onto-intro-page.html[1]

This report explains our solutions to problems in ontological engineering in a somewhat informal manner and with the help of concrete illustrations taken from our experience. The reader is encouraged to refer to the BNF specification in Appendix A and the axiomatic specification in Appendix B at the end of the report for precise formulations of the structure and semantics of the μK ontology.

[1]The μK ontology is available to interested researchers in a variety of forms including C++ objects, Lisp-like lists, or plain ASCII text. Interfaces for browsing the ontology and searching for concepts are also available in several forms. Further information may be obtained from the author or on the World Wide Web at the above address.

1.1 The MT Situation

Knowledge-Based Machine Translation (KBMT) views the task of translating a text in one natural language to a different natural language as a problem in mapping meanings. Meanings in the source language, rather than just its syntax, must be mapped to the target language. A basic requirement for doing this is a representation of text meaning which is not in the source or target languages. This interlingual meaning representation must be grounded in a language-independent ontology that supplies not only the primitive symbols that constitute the representation but also a set of well-defined composition operations for combining different primitive symbols to represent bigger chunks of meaning.

Mikrokosmos (μK) is a KBMT system under development at New Mexico State University jointly with the US Department of Defense (Beale, Nirenburg, and Mahesh, 1995; Mahesh and Nirenburg, 1995a, 1995b; Onyshkevych and Nirenburg, 1994). Unlike many previous projects in interlingual MT, μK is a large-scale, practical MT system focusing currently on translating Spanish news articles to English. A lexicon of approximately 7,000 Spanish words supported by an ontology of about 4500 concepts is already in place. High quality semantic analyses of 5 article-length Spanish texts in the domain of company mergers and acquisitions has already been produced.

The role of the ontology in μK is shown in Figure 1. The ontology sits centrally serving at least three different purposes. It provides concepts for representing word meanings in the lexicon for a source or target language. The figure shows an analysis lexicon where words are mapped to concepts in the ontology along with any modifiers that augment the selectional constraints represented by the conceptual relationships in the ontology. Such modification enables the system to capture the many nuances of meanings in different languages. Every lexicon in the μK system, irrespective of its language and whether it is constructed for analysis or for generation, maps to the same set of concepts in the ontology.

The network of concepts in the ontology also serves the role of a semantic network by providing the search space for a powerful search mechanism (called Onto-Search in Figure 1) that finds the shortest path in the ontology between a pair of concepts. Measures of such ''distances'' between concepts are very useful for checking selectional constraints in service of basic disambiguation processes in the semantic analyzer. Moreover, the search also supports far more powerful ''semantic affinity'' tests for the purpose of interpreting various non-literal expressions (such as metaphor and metonymy) and anomalous inputs using special purpose microtheories.

Concepts in the ontology are the building blocks used by the lexicon (statically) and the analyzer (dynamically) to construct Text Meaning Representations (TMR). A TMR can be viewed as an instantiation of a subgraph of the ontology (typically much smaller than the entire ontology) with certain linguistic augmentations (such as aspect, attitudes, modalities, and so on). TMRs serve as the interlingual representation of meanings that are fed to target language generators. The role of the ontology in text generation is not shown in Figure 1 or discussed in this report.

Figure 2 illustrates the μK architecture for analyzing input texts. The μK ontology can be seen situated in this architecture which also includes components where the ontology does not play a direct role. However, the ontology also plays a significant role in the acquisition of lexicons and in the building and testing of analyzers and generators. We return to this issue in the discussion of acquisition methodology later in this report.

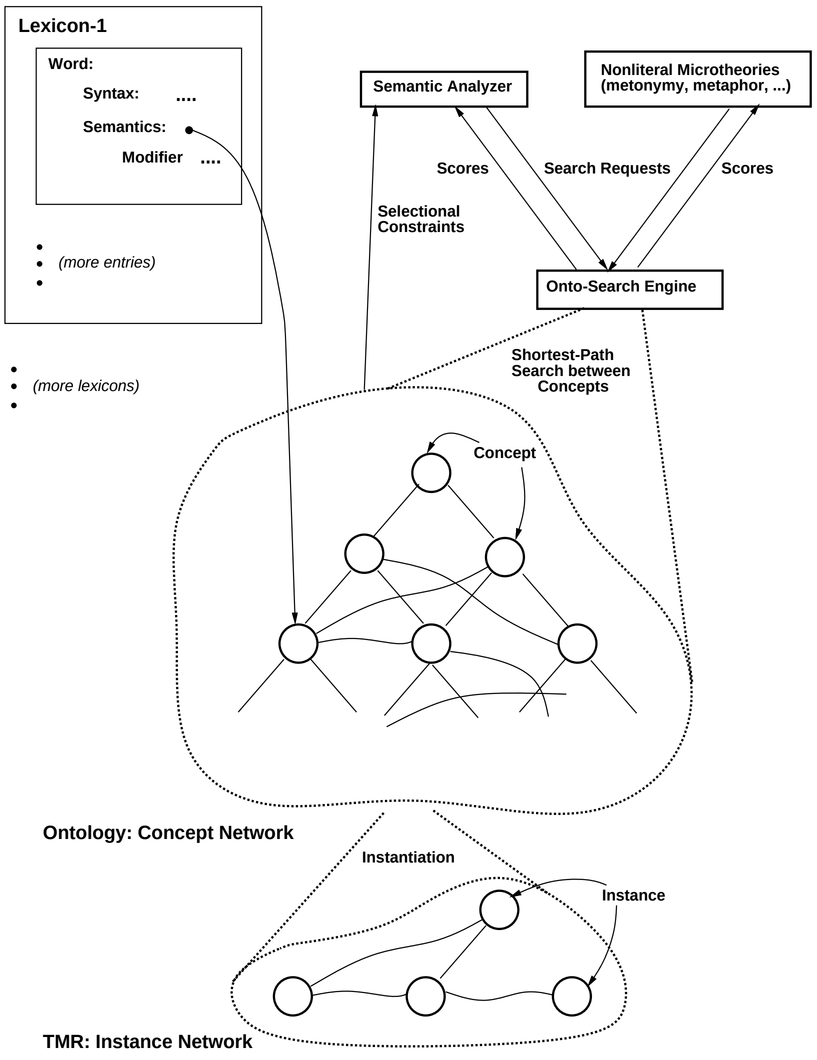


Figure 1: The Role of the Ontology in Mikrokosmos

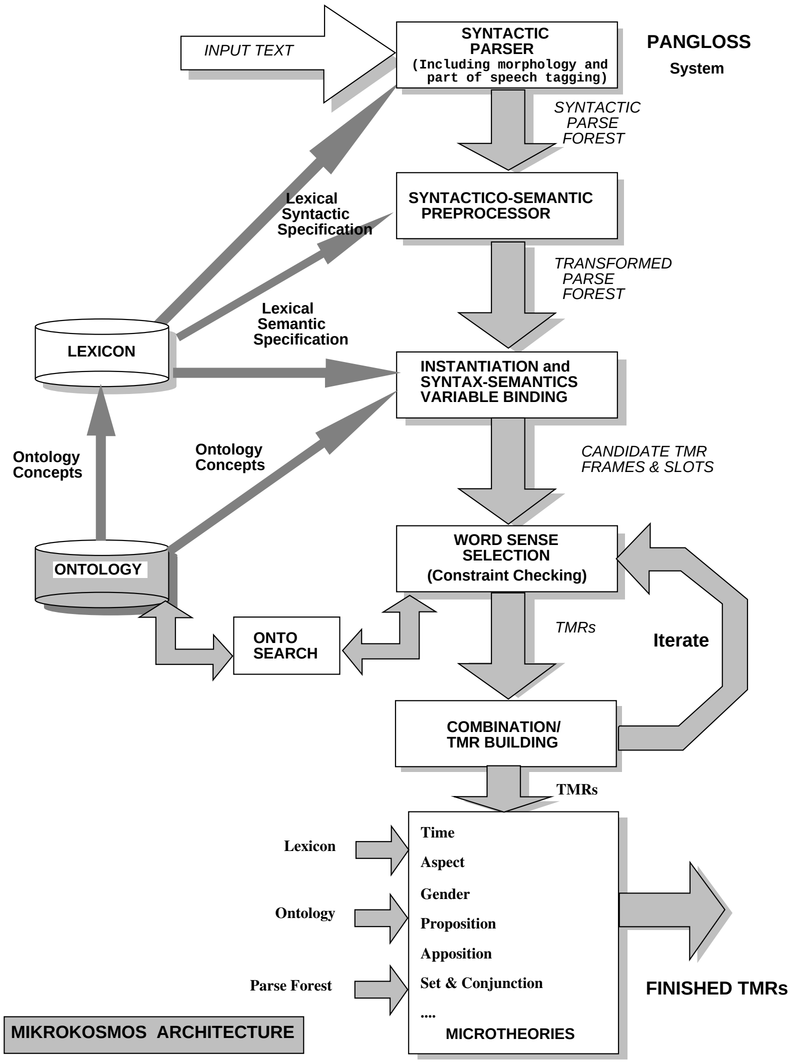


Figure 2: The Ontology Situated in the Mikrokosmos NLP Architecture. It Supplies Conceptual Knowledge both for Lexical Representation and for Constraining Semantic Interpretation.

1.2 What Is an Ontology?

In the knowledge-based approach to machine translation, meanings of source language (e.g., Spanish) texts are represented internally in a language-neutral interlingua (e.g., Nirenburg, 1989).[2] The interlingual meaning representation (that we call a TMR) is derived from representations of word meanings in computational lexicons and from representations of world knowledge in ontologies (and possibly episodic knowledge bases). An interlingual meaning representation once derived is input to a language generator that produces the translation in the target language (e.g., English). A key issue in the design and development of KBMT systems is the set of symbols used to represent interlingual meaning as well as the structure of the meaning representation. In our methodology, the set of symbols and possible relationships among them are grounded in a language-independent knowledge source called the *ontology.* The symbols are defined as *concepts* in the ontology. The same set of concepts are used to represent word meanings in lexicons. The internal structure of these concepts is reflected in both lexical and text meaning representations. TMRs are essentially instantiations of ontological concepts connected together according to constraints derived both from the ontology and elsewhere.

A typical dictionary definition of an ontology is ‘‘The branch of metaphysics that studies the nature of existence.’’ For us, an ontology is a computational entity, a resource containing knowledge about what ‘‘concepts’’ exist in the world and how they relate to one another. A concept is a primitive symbol for meaning representation with well defined attributes and relationships with other concepts. An ontology is a network of such concepts forming a symbol system where there are no uninterpreted symbols (except for numbers and a small number of known literals).

An ontology for NLP purposes is a body of knowledge about the world (or a domain) that a) is a repository of primitive symbols used in meaning representation; b) organizes these symbols in a tangled subsumption hierarchy; and c) further interconnects these symbols using a rich system of semantic relations defined among the concepts. In order for such an ontology to become a computational resource for solving problems such as ambiguity and reference resolution, it must be actually constructed, not merely defined formally. The ontology must be put into well-defined relationships with other knowledge sources in the system. In an NLP application, the ontology supplies world knowledge to lexical, syntactic, semantic, and pragmatic processes, and other microtheories.

An ontology is a database with information about

- what categories (or **concepts**) exist in the world/domain,
- what properties they have,
- and how they relate to one another.

In interlingual machine translation, the principal reasons for using an ontology are (Figure 1):

- to provide a grounding for representing text meaning in an interlingua;
- to enable lexicons for different languages to share knowledge;
- to enable source language analyzers and target language generators to share knowledge.

[2]This and the following subsections in Section 1 reuse parts of the introductory text in Mahesh and Nirenburg (1995a; 1995b).

In addition, ontologies are also used in KBMT

- to store selectional restrictions and other pieces of world knowledge;
- to ''fill gaps'' in text meaning by making inferences based on the content of conceptual knowledge in the ontology;
- to resolve semantic ambiguities and interpret non-literal language by making inferences using the topology of the ontology to measure the semantic affinity between meanings;
- as a classification of people, places, social roles, and organizations which forms the basis for organizing an onomasticon, a gazetteer, or other such databases.

In addition, the same ontology can be of great value in a variety of other tasks including database merging (Dowell, Stephens, and Bonnell, 1995; Van Baalen and Looby; 1995), integration of software or business enterprise models (Fillion, Menzel, Mayer, and Blinn, 1995), and so on. Essentially, an ontology such as the μK ontology is invaluable wherever a ''semantic wall'' is to be scaled, be it to translate between a pair of natural languages, a pair of database schemas, or to integrate different models of the same domain or similar phenomena in the world. We provide specific illustrations of the uses of the ontology in NLP and MT below, but refer the reader to other literature for examples of more broader uses of ontologies (IJCAI Ontology Workshop, 1995).

In the Mikrokosmos project, we have developed an ontology covering a wide range of categories in the world. Several illustrative concepts from the μK ontology will be shown below. The above uses of the ontology in machine translation will also be illustrated through examples. See Mahesh and Nirenburg (1995b) and Beale, Nirenburg, and Mahesh (1995) for more detailed examples.

1.3 A Situated Ontology

A situated ontology is a world model used as a computational resource for solving a particular set of problems (Mahesh and Nirenburg, 1995a). It is treated as neither a ''natural'' entity waiting to be discovered nor a purely theoretical construct. World models (ontologies) in computational applications are artificially constructed entities. They are created, not discovered. Many ontologies are developed for purely theoretical purposes and never really constructed to become a computational resource. Even those that are constructed, Cyc (Lenat and Guha, 1990) being the best example, are often developed without the context of a practical situation (e.g., Smith, 1993). Many practical knowledge-based systems, on the other hand, employ world or domain models without recognizing them as a separate knowledge source (e.g., Farwell, et al. 1993). In the field of NLP, there is now a consensus that all NLP systems that seek to represent and manipulate meanings of texts need an ontology (e.g., Bateman, 1993; Nirenburg, Raskin, and Onyshkevych, 1995). In our continued efforts to build a multilingual KBMT system using an interlingual meaning representation (e.g., Onyshkevych and Nirenburg, 1994), we have developed an ontology to facilitate natural language interpretation and generation. The central goal of the Mikrokosmos project is to develop a system that produces a comprehensive Text Meaning Representation (TMR) for an input text in any of a set of source languages.[3] Knowledge that supports this process is stored both in language-specific knowledge sources and in an independently motivated, language-neutral ontology (Carlson and Nirenburg, 1990; Mahesh, and Nirenburg, 1995a).

[3] The current system prototype is an analyzer of Spanish.

Not only is the μK ontology situated in our machine translation architecture, its development is also very much situated in the processes of lexical knowledge acquisition, development of analysis programs, and system testing and evaluation. The primary source of meanings to be encoded as new concepts in the ontology is the continual stream of requests for concepts from lexicographers trying to represent meanings of words using concepts in the ontology. Concepts acquired per such requests are in turn tested in the semantic analyzer almost immediately and changes and corrections sent to the ontology acquirers within a few days. Moreover, since the number of people browsing the ontology (lexicographers, system builders, and testing and evaluation experts, over 10 in our situation) is many times the number of ontology developers (at most two in our case), it is highly likely that any error will be noticed during browsing, especially by those who requested the concept in error, and corrected through the feedback process. In fact, because of this imbalance in the numbers of ontology developers and customers, we had to resort to computer support in the form of interfaces and bookkeeping programs to assist in sending and keeping track of requests and complaints to ontology developers. Ontology developers at present interact regularly with lexicographers building lexicons in Spanish, Japanese, and Russian.[4]

1.4 What Does the Ontology Do for NLP?

As already noted, an ontology has several different uses in KBMT. Brief examples of how the ontology aids NLP and MT are shown below:

- It is the main repository of selectional preferences on meaning composition. Knowledge of such constraints is invaluable for resolving ambiguities by means of the constraint satisfaction process shown in the form of the ‘‘Onto Search’’ (Onyshkevych, 1995) box in Figures 1 and 2. For example, consider the Spanish sentence ‘‘El grupo Roche adquirio Docteur Andreu.’’ Did Roche ACQUIRE[5] or LEARN Docteur Andreu? The verb ‘‘adquirir’’ can mean either of these. However, selectional constraints on the ACQUIRE and LEARN concepts in the ontology tell us that if the theme is not an ABSTRACT-OBJECT, then the meaning of the verb is likely to be ACQUIRE and not LEARN (see Figure 7). Since Docteur Andreu is known to be the name of a CORPORATION (which is a SOCIAL-OBJECT, not an ABSTRACT-OBJECT), the correct meaning of ‘‘adquirir’’ in this sentence is ACQUIRE. The ontology is also used for checking selectional constraints. For example, it is used to determine if Docteur Andreu is an ABSTRACT-OBJECT.

- It enables inferences to be made from the input text using knowledge contained in the concepts. This can help resolve ambiguities as well as fill gaps in the text meaning. A default value from an ontological concept can be filled in a slot, for instance, when a text does not provide a specific value. For example, if the text says ‘‘John went swimming although the water was cold,’’ using the default value of the SUBSTRATE slot of SWIM, namely WATER, the analyzer can infer that the relationship between the MOTION-EVENT SWIM and WATER is one of SUBSTRATE (and its inverse SUBSTRATE-OF) rather than other possibilities such as a MATERIAL that is INGESTed.

- It enables inferences to be made using the topology of the network, as in searching for the shortest path between two concepts. Such search-based inferences are used all the time (in the ‘‘onto search’’ box in Figures 1 and 2) to check how well a selectional constraint is satisfied by a piece of text. It is often the case in NLP that none of the alternatives satisfies the constraints completely. The topology

[4]Construction of Japanese and Russian lexicons has begun recently. Although the ontology is already being used for representing lexical meanings in all three languages, the μK analyzer has so far only been tested on Spanish texts.

[5]Names of concepts in the ontology are shown in small capitals in this report.

of the ontology helps the analyzer pick the meaning that is ''closest'' to meeting the constraints. Such inferences can also support metonymy and metaphor processing, figuring out the meaning of a complex nominal, or be used in other types of constraint relaxation when the input cannot be treated with the available knowledge.

The ontology parallels μK lexicons in developing the MT system. Word meanings are represented partly in the lexicon and partly in the ontology (see Figure 7 for an example). *In principle, the separation between ontology and lexicon is as follows: language-neutral meanings are stored in the former; language-specific information in the latter.* In a multilingual situation, it is not easy, however, to determine this boundary. As a result, ontology and lexicon acquisition involves a process of daily negotiations between the two teams of acquirers. The easiest solutions to many difficult problems in lexical semantic representation require the addition of a new concept to the ontology under certain ''catch all'' frames (this is the only solution in the ''word sense approach'' to ontology development where word senses are mapped almost one to one to concept names (e.g., Bateman, et al. 1990; Knight and Luk, 1994)). In Mikrokosmos, a set of guidelines was developed for suggesting ways of finding solutions to lexical problems without adding ''catch all'' concepts. (A sample of these guidelines is presented in Section 7.6 of this report.)

The ontology also aids meaning representation and, in particular, lexical representation as follows:

- It forms a substrate upon which word meanings in any language are grounded and constructed in the lexicon (see, e.g., Nirenburg and Levin, 1992). It guarantees that every symbol used in representing lexical semantics is defined as a concept, is well-formed, and has known relations to all other symbols. Moreover, it provides the basic internal structure of meaning elements (i.e., concepts) so that lexicographers only need to add in the lexicon any necessary modifiers to the ontological meaning.

- The ontology guides lexicographers along particular ways of decomposing and representing word meanings. Lexicographers constantly face the choice between making a word sense a primitive in the meaning representation and decomposing it in various ways to arrive at other, existing primitives. Such a choice is almost always underconstrained given only the meaning of the word in that language. The ontology provides a basis for making a good decision by bringing in a range of ontological and representational constraints that cover the classification of all related meanings, irrespective of the language the word belongs to.

- It helps partition multilingual MT work. This is a big methodological advantage since it allows us to partition the task of developing multilingual MT systems into the independent development of analysis and generation lexicons for the different languages. The ontology serves as a common ground both between an analyzer and a generator and between different languages. It provides a way for a clean separation of language independent world knowledge from linguistic knowledge to maximize the sharing of knowledge between the different representations (Mahesh and Nirenburg, 1996).

- An ontology-based approach to lexical semantics makes lexical representations highly parsimonious by reducing the number of different entries needed in the lexicon. Meaning common to many words in a language can be factored out and represented in the ontology in a concept to which the words map. Moreover, this approach provides rich expressiveness for modifying and augmenting base meanings in a variety of ways, instead of simply enumerating a multitude of senses. Concepts in the ontology can be modified both through the relations and attributes in their slots and using the representational apparatus external to the ontology for encoding aspect, attitude, modality, and other elements of linguistic semantics. As a result, the ontology allows the lexicon to capture word senses through far fewer entries than found in a typical dictionary. For example, Nirenburg, Raskin, and Onyshkevych (1995) have shown that the 54 meanings for the Spanish verb ‘‘dejar’’ listed in the Collins Spanish-English dictionary can be collapsed into just 7 different lexical mappings using the μK approach. Much of this power comes from the expressiveness of the representation that allows complex combinations and modifications of ontological concepts in meaning representations. See Onyshkevych and Nirenburg (1994) for detailed examples of lexical representations.

- It allows generic and incomplete lexical representations which are nevertheless well-formed. For example, in mapping certain adjectives and adverbs, the lexicon needs to refer to the AGENT slot (say) of an EVENT without knowing which particular event it is. This can be done in Mikrokosmos by directly referring to the AGENT slot of the EVENT concept (even though not every EVENT has an AGENT) and using a variable-binding or coreference mechanism to ultimately map the event to a more specific event in the TMR.
- It allows variable-depth semantics where lexical semantics can range from a direct mapping to a concept without any modifications all the way to a complex combination of several concepts with various additional constraints, relaxations, and additional information such as time, aspect, quantities, and their relationships. It helps avoid unnecessarily deep decomposition of complex meanings by resorting to a controlled proliferation of primitive concepts (see Section 4.1).
- It also enables sophisticated ways of reducing manual effort in lexicon acquisition. For example, it allows a derivational morphology engine to automatically suggest concepts for the lexical mappings of the derived words which can then be refined by the acquirer instead of acquiring from scratch.

1.5 Computational Ontologies for NLP

Just as no one so far has found ‘‘the right’’ grammar for a natural language (such as English), it is reasonable to argue that any ontology we build will not be ‘‘the right’’ or the only ontology for a domain or a task. The μK ontology is one possible classification of concepts in its domain constructed manually according to a well developed set of guidelines. Its utility in NLP can only be evaluated by the quality of the meaning representations or translations produced by the overall system or through some other evaluation of the overall NLP system (such as in an information extraction or retrieval test). This is not to say that the ontology is randomly constructed. It is not. Its construction has been constrained throughout by guidelines (see Section 7.6 below), by constant scrutiny by lexicographers, testers, and other experts, as well as by the requirements of meaning representation and lexicon acquisition.

In NLP work, the term ‘‘ontology’’ is sometimes also used to refer to a different kind of knowledge base which is essentially a hierarchical organization of a set of symbols with little or no internal structure to each node in the hierarchy (e.g., Farwell, et al. 1993; Knight and Luk, 1994). Frames in the μK ontology, however, have a rich internal structure through which are represented various types of relationships between concepts and the constraints, defaults, and values upon these relationships. It is from this rich structure and connectivity that one can derive most of the power of the ontology in producing a TMR from an input text. Mere subsumption relations between nearly atomic symbols do not afford the variety of ways listed above in which the μK ontology aids lexicon acquisition and semantic processes such as disambiguation and non-literal interpretation.

The above distinction between highly structured concepts and nearly atomic concepts can be traced to a difference in the grain size of decomposing meanings. A highly decompositional (or compositional) meaning representation relies on a very limited set of primitives (i.e., concept names). If we do this, the representation of many basic concepts becomes too complex and convoluted. The other extreme is to map each word sense in a language to an atomic concept. As a result, the nature of interconnection among these concepts becomes unclear, to say nothing about the explanatory power of the system (cf. the argument about the size of the set of conceptual primitives in Hayes, 1979). In Mikrokosmos, we take a hybrid approach and strive to contain the proliferation of concepts for a variety of methodological reasons, such as tradeoffs between the parsimony of ontological representation and that of lexical representation and the need for language independent meaning representations. Control over proliferation of concepts is achieved by a set of guidelines that tell the ontology acquirer when not to introduce a new concept (see Section 7.6 below).

We do not make any distinction between an ontology and a world knowledge base. The μK ontology is a taxonomic classification of concepts as well as a repository of world knowledge expressed in the form of conceptual relationships. Instead of separating the classification (often called ''ontology'') from the rest of the knowledge about the concepts being classified (often called ''knowledge base''), we keep all knowledge about concepts in the ontology. This way all knowledge that is not specific to any particular natural language will be in the ontology while language-specific knowledge is confined to individual lexicons for the different languages.

The μK ontology however makes a clear distinction between conceptual and episodic knowledge and includes only conceptual knowledge. Instances and episodes are acquired in a separate knowledge base called the *onomasticon.* The methodology for acquiring the onomasticon includes a significant amount of automation and is very different from ontology acquisition which is done manually via continual interactions with lexicographers.

1.6 Our Product: The Mikrokosmos Ontology

The μK system currently processes Spanish texts about mergers and acquisitions of companies. However, since the input language is unrestricted, the ontology must, in fact, cover a wide range of concepts outside this particular domain. In addition, the analyzer encounters a variety of well-known problems in NLP which require the application of a significant amount of world knowledge.

We are currently in the process of a massive acquisition of concepts (OBJECTs, EVENTs as well as PROPERTYs) related to the domain of company mergers and acquisitions (Mahesh and Nirenburg, 1995a, 1995b).[6] However, since our input texts are unedited, real-world texts, the ontology must support a wide range of word meanings many of which are outside the domain of mergers and acquisitions by any stretch of imagination. Moreover, we also need to support alternative meanings of words in the texts which must be represented in the lexical entries for the word. As a result, the μK ontology is best described as a broad-coverage ontology with more complete coverage in the domain of mergers and acquisitions than in many other domains.

Over a period of several months, the μK ontology was developed by starting with an ontology that had less than 2000 concepts and acquiring nearly 3000 new concepts organized in a tangled hierarchy with ample interconnectivity across the branches. Continued efforts have been made to monitor some of the

[6]In parallel, a Spanish lexicon that maps lexemes to concepts in this ontology is also being acquired on a massive scale.

qualities of the ontology and maintain certain overall characteristics. For example, the ontology emphasizes depth in organizing concepts and reaches depth 10 or more along a number of paths. The branching factor is kept much less than 10 at most points (the overall average being less than 5). Each concept has, on average, 10 to 15 slots linking it to other concepts or literal constants. The top levels of the hierarchy (Figure 3 below) have proved very stable as we are continuing to acquire new concepts at the lower levels.

Unlike many other ontologies with a narrow focus (e.g., Casati and Varzi, 1993; Hayes, 1985; Mars, 1993), our ontology must cover a wide variety of concepts in the world. In particular, our ontology cannot stop at organizing terminological nouns into a taxonomy of objects and their properties; it must also represent a taxonomy of (possibly, complex) events and include many interconnections between objects and events to support a variety of disambiguation tasks. At present, the μK ontology has about 4600 concepts in its hierarchies. Each concept, on an average, is connected to at least 14 other concepts through the relationships represented in its slots.

All entities in the μK ontology are classified into *free-standing entities*[7] and *properties.* Free-standing entities are in turn classified into OBJECTs and EVENTs. Figure 3 shows the top-level hierarchy in the ontology. OBJECTs, EVENTs, and PROPERTYs constitute the *concepts* in the ontology which are represented as *frames.* Figures 4, 5, and 6 show snapshots of parts of the OBJECT, EVENT, and PROPERTY hierarchies in the μK ontology. These figures show only the names of concepts and the taxonomic relations among them. Each node shown in these figures in fact has an elaborate internal structure of slots and fillers. An example of the internals of a concept can be seen in Figure 7. A more detailed tour of the μK ontology is presented in ''A Guided Tour of the Mikrokosmos Ontology'' (see Mahesh and Wilson, forthcoming).

Each frame is a collection of slots with one or more facets and fillers. The slots (including inherited ones) collectively define the concept by specifying how the concept is related to other concepts in the ontology (through *relations*) or to literal or numerical constants (through *attributes*). Lexicon entries represent word or phrase meanings by mapping them to concepts in the ontology.

A TMR is a result of instantiating concepts from the ontology that are referred to in a text and linking them together according to the constraints in the concepts as well as those listed in lexicon entries and special TMR building rules. A number of concepts in the domain of mergers and acquisitions are located under the ORGANIZATION subtree under SOCIAL-OBJECTs and the BUSINESS-ACTIVITY subtree under SOCIAL-EVENTs (see Figures 3, 4, and 5). Figure 7 shows a sample frame in the ontology along with a lexical mapping to that concept.

2 The Structure of the Ontology

The ontology is a directed graph where nodes are concepts. Links between nodes are represented as slots and fillers. Slot names themselves are the class of concepts known as PROPERTYs except for the names of some ''bookkeeping'' slots. Links are in fact multi-dimensional since each link can have several facets. Each facet can take one or more fillers. No importance is attached to the order of multiple fillers in a slot.[8] A complete BNF description of the ontology is provided in Appendix A. A corresponding set of axioms defining the structure of the ontology is shown in Appendix B at the end of this report.

[7] *Free-standing entity* is not an actual concept in the ontology; we use the term when describing the top-level organization of the ontology in order to distinguish between OBJECTs and EVENTs on the one hand and PROPERTYs on the other.

[8] See Section 6.4 for an exception in the case of Range specifications of LITERAL-ATTRIBUTEs.

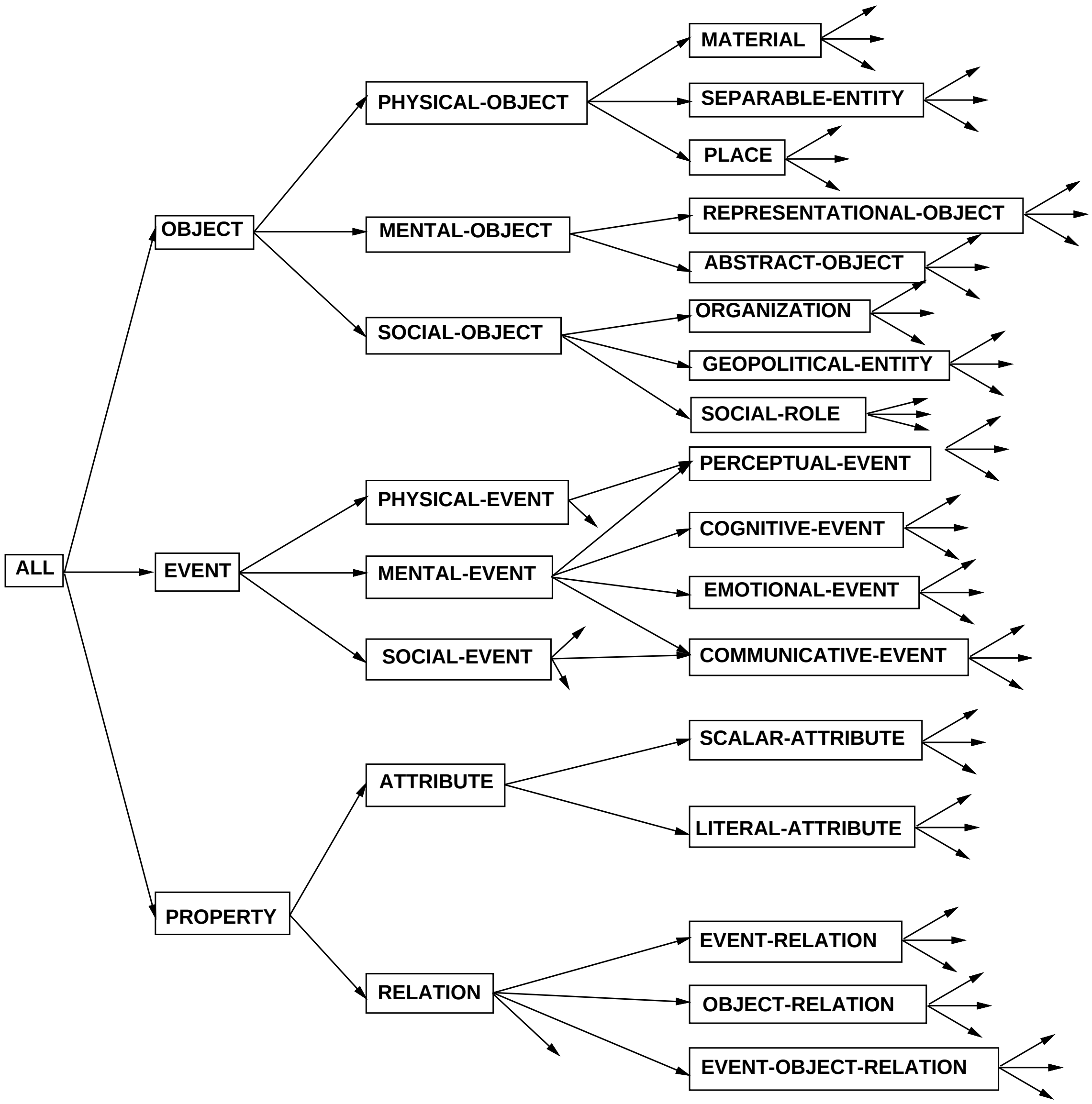


Figure 3: Top-Level Hierarchy of the Mikrokosmos Ontology Showing the First Three Levels of the OBJECT, EVENT, and PROPERTY Taxonomies.

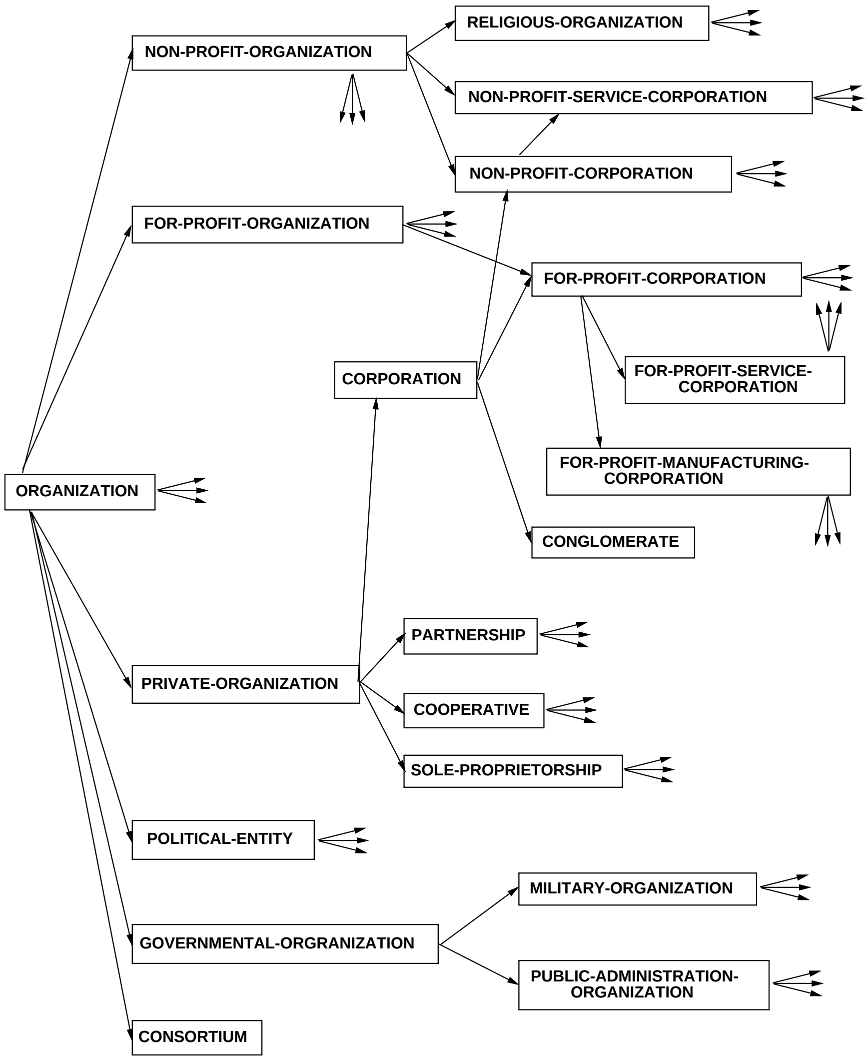


Figure 4: A Snapshot of the ORGANIZATION Hierarchy Under OBJECT in the μK Ontology.

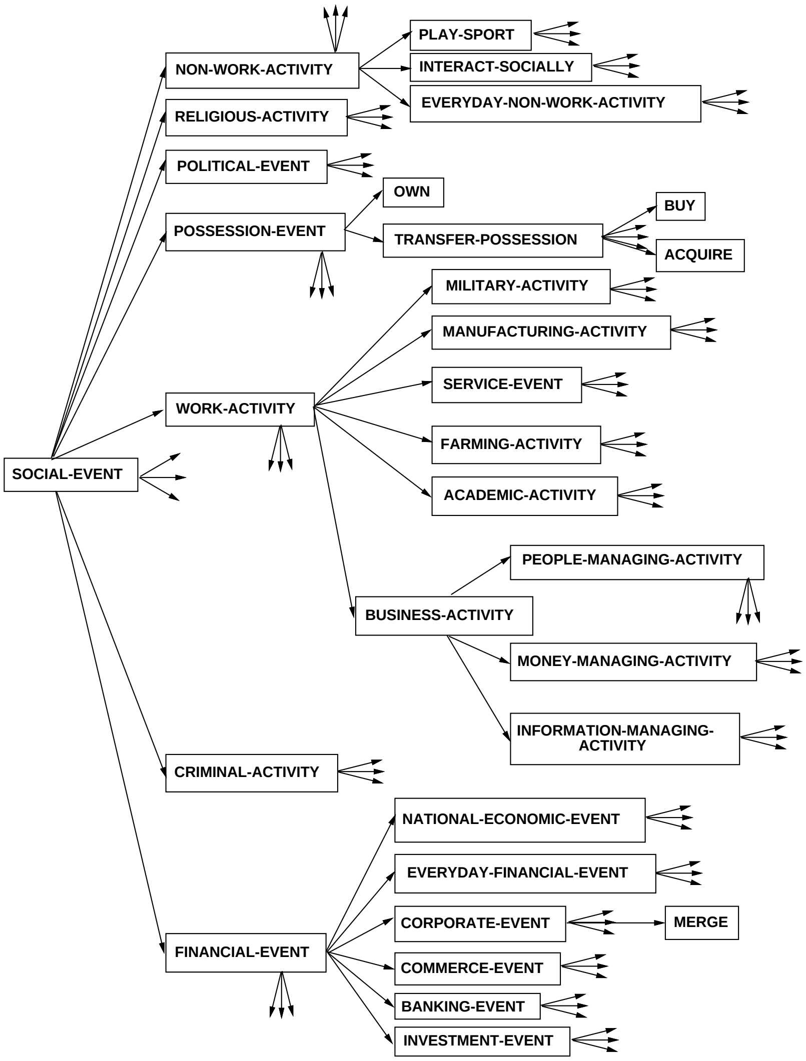


Figure 5: A Snapshot of the SOCIAL-EVENT Hierarchy in the μK Ontology.

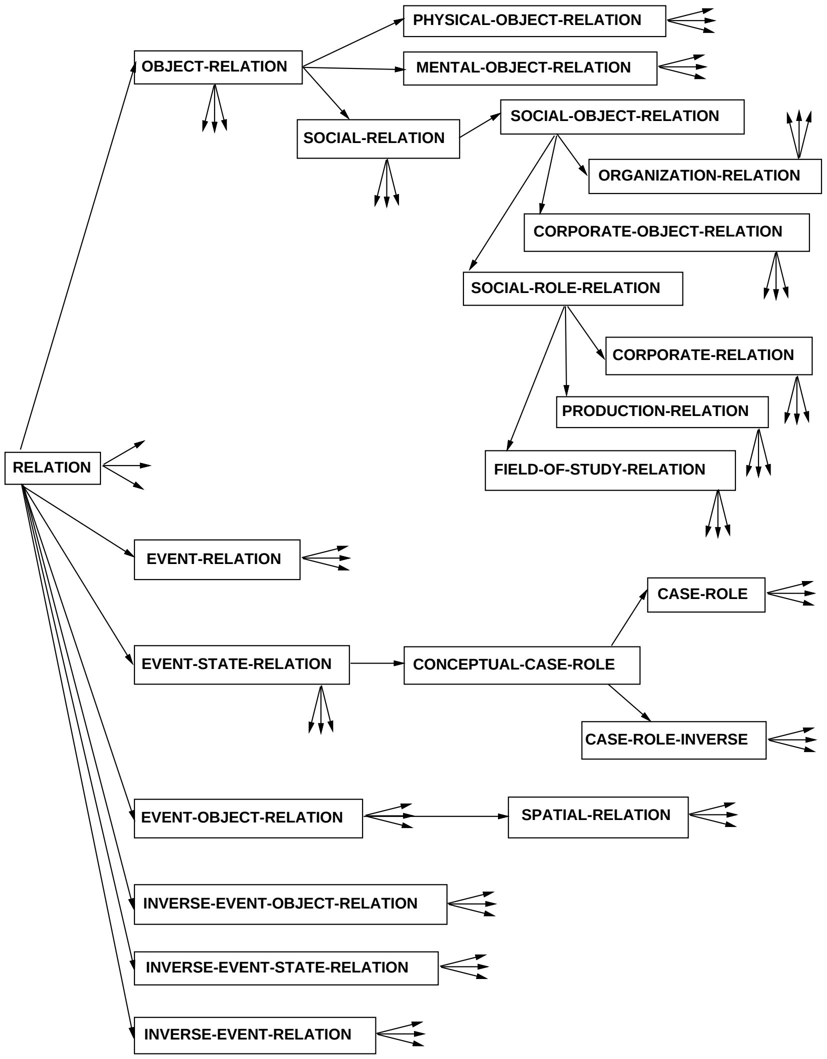


Figure 6: A Snapshot of the RELATION Hierarchy Under PROPERTY in the μK Ontology.

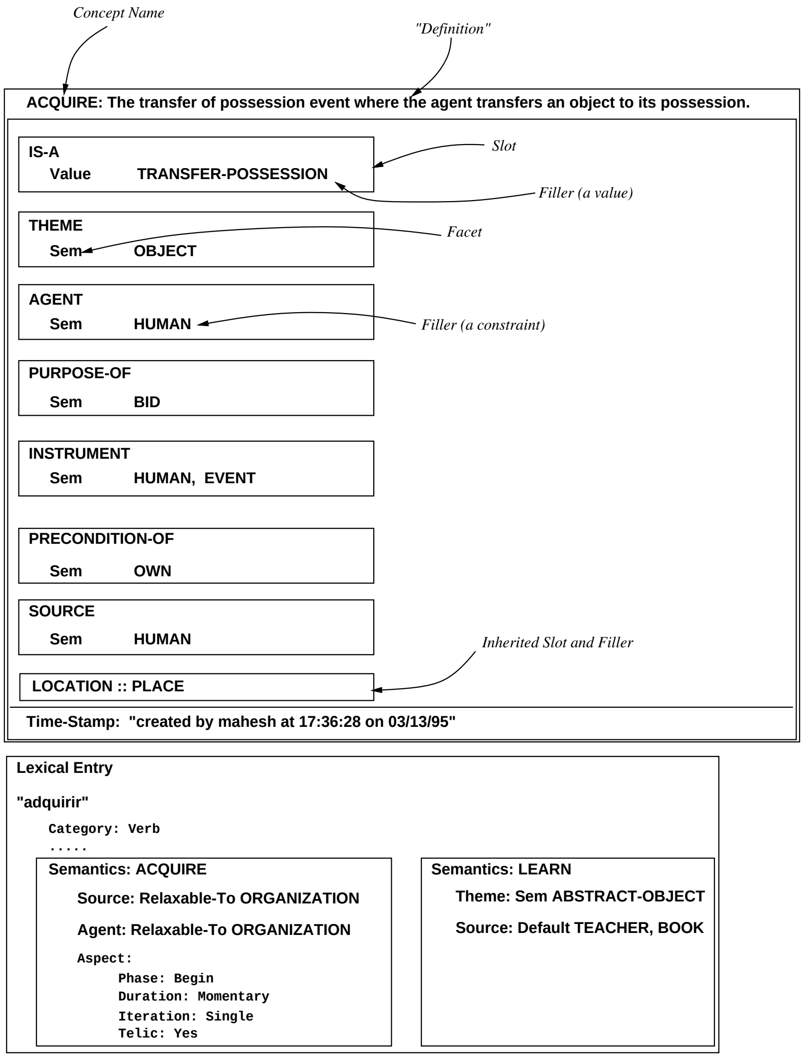


Figure 7: Frame Representation for Concept ACQUIRE. Also Shown is a Part of the Lexical Entry for the Spanish Verb ‘‘adquirir’’ with Semantic Mappings to ACQUIRE and LEARN Events. The Mappings Modify the Constraints in the Ontology and Add New Information such as Aspect.

Formally, the ontology is a graph with *only two kinds of patterns in its subgraphs.* The structure of these patterns will be elaborated further below. All concepts in the ontology are classified into one of OBJECTs, EVENTs, or PROPERTYs. OBJECTs and EVENTs are stand-alone concepts; they are instantiated in the TMR. PROPERTYs, on the other hand, are not normally individually instantiated;[9] they become slots in OBJECTs and EVENTs. As such, PROPERTYs cannot modify other PROPERTYs. They are specified only in terms of their Domains and Ranges.

PROPERTYs are of two types: RELATIONs and ATTRIBUTEs. Corresponding to these two are the two subgraph patterns that constitute the ontology (shown in Section 2.2 below). RELATIONs differ from ATTRIBUTEs in that RELATIONs map an OBJECT or EVENT to another OBJECT or EVENT while ATTRIBUTEs map an OBJECT or an EVENT to a scalar (i.e., number) or a literal symbol. In other words, a filler of an ATTRIBUTE slot is a number, a literal symbol, an open or closed range of numbers, or, in unfortunate cases, an undefined symbol not in the known set of literals.

2.1 Slots and Facets

A slot is the basic mechanism for representing relationships between concepts. In fact, *slot* is the fundamental meta-ontological predicate based on which the entire ontology can be described axiomatically (see Appendix B). Most ''content'' slots are PROPERTYs which are themselves defined as concepts in the ontology. There is a closed class of special slots. These slots are described below.

All slots have all permissible facets except as mentioned for the special slots below. An additional constraint from the distinction between concepts and instances is that *slots in concepts* (other than the special slots listed below) *have only Sem facets and slots in instances have only Value facets.* Permissible facets are:

- Value: the filler of this facet is an actual value; it may be an instance of a concept, a literal symbol, a number, or another concept (in the case of the special slots listed below).
- Sem: the filler of a Sem facet is either another concept or a literal, number, or scalar range. In any case, the filler serves as a selectional restriction on the fillers of the Value and Default facets of the slot. It is through these selectional restrictions that concepts in the ontology are related (or linked) to other concepts in the ontology (in addition to taxonomic links).
- Default: the filler of a Default facet is the value of the slot in the absence of an explicit Value facet. For the sake of simplicity, Default facets are not inherited in the μK ontology (and hence are not in much use either).
- Measuring-Unit: this facet is used to add a measuring unit for the number that fills the Value, Default, or Sem facets of the same slot. MEASURING-UNITs are also defined as concepts in the ontology.[10]
- Salience: the Salience facet can be used to indicate how important or central a slot is to the entire concept. This may be used in lexicons or actual TMRs to indicate the focus of a part of the text, but is not used in the ontology itself.

[9] They may, however, be instantiated in a TMR by means of a reification operation (e.g., Russell and Norvig, 1995), thereby making them stand-alone instances in the TMR. Such reification can also be triggered from the lexicon.

[10] ''Measuring unit'' has been used as the name of the facet as well as the concept. The facet should perhaps be renamed as ''Unit.'' However, this facet is not used much in the ontology.

- Relaxable-to: this facet is used only in the lexicon and indicates to what extent a language permits a selectional constraint to be violated in nonliteral usage such as a metaphor or metonymy. The filler of this facet is a concept that indicates the maximal set of possible fillers beyond which the text should be considered anomalous.

In addition to slots and facets, additional structure is available through the notion of a *View.* Each facet can have multiple views. In the current representation, we use only one view called Common. All facets in all slots of concepts have the Common view. In fact, the Common view is transparent in the Mikrokarat knowledge acquisition tool used for building the μK ontology.

2.1.1 Special Slots

- **Definition:** This slot is mandatory in all concepts and instances. It has only a Value facet whose filler is a definition of the concept in English intended only for human consumption.
- **Time-Stamp:** This is used to encode a signature showing who created or updated this concept and when. It may be noted that not all concepts have an up to date Time-Stamp since the tools we have been using have not provided this feature at various times.
- **Referenced-By-Token:** This slot was used in the past to encode a sentence from a source text which led to a need for this concept. Many English sentences were thus inserted into concept frames in the past. In the current multi-lingual setup of this project, we have done away with this slot (mainly because of inconsistent ways of encoding the example sentences in the ontology we started with). This slot perhaps belongs only in a lexicon and will not be used in the ontology.
- **Is-A:** This slot is mandatory for all concepts except ALL which is the root of the hierarchy. Instances do not have an Is-A slot. This slot has only a Value facet which is filled by the names of the immediate parents of the concept. A concept missing an Is-A slot is called an *orphan.* Ideally, only ALL should be an orphan.
- **Subclasses:** This slot is mandatory for all concepts except leaves (which do not have other concepts as their immediate children). Concepts that only have instances as children do not have this slot. This slot also has only a Value facet which is filled by the names of the immediate children of the concept.
- **Instances:** This slot is present in any concept that has instances as children. A concept may have both Subclasses and Instances. There is no requirement that only leaf concepts have instances. This slot also has only a Value facet filled by the names of the instances of this concept.
- **Instance-Of:** This slot is mandatory for all instances and is present only in instances. It has only a Value facet which is filled by the name of the parent concept.
- **Inverse:** This slot is present in all RELATIONs and only in RELATIONs. It has only a Value facet which is filled by the name of the RELATION which is the Inverse of this RELATION.
- **Domain:** This slot is present in all PROPERTYs and only in PROPERTYs. It has only a Sem facet which is filled by the names of concepts which can be in the domain of this RELATION or ATTRIBUTE. A Domain slot never has a Value facet since there is never an instance of a PROPERTY in the ontology. A PROPERTY enters a text meaning representation (TMR) only as a slot in an instance of an OBJECT or EVENT unless the slot has been reified in which case the reified slot has Domain and Range slots in the TMR.

- **Range:** This slot is also present in all PROPERTYs and only in PROPERTYs. It too has only a Sem facet. In RELATIONs, the filler of the Sem facet is the names of concepts that are in the range of this RELATION. In an ATTRIBUTE, the Sem facet is filled by all the possible literal or numerical values permissible for that ATTRIBUTE. The filler can also be a numerical range specified using appropriate mathematical comparison operators (such as $>$, $<$, ...). Again, the Range slot never has a Value facet since there is never an instance of a PROPERTY in the ontology.

All other slots have Sem facets in concepts and Value facets in instances. Any slot either in a concept or in an instance may have in addition a Default, a Salience, or a Measuring-Unit facet. These facets have only been used sparingly in the current ontology. In fact, the Salience facet has never been used. It may also be noted that no facet is mandatory in these non-special slots. For example, a slot may have just a Default facet specified (with the implication that there is no constraint on other possible values for the filler).

2.2 Subgraph Patterns in the Ontology

The entire ontology is built of only two different subgraph patterns. These two patterns correspond to a slot that is a RELATION and a slot that is an ATTRIBUTE. In this analysis, we ignore certain special slots. Some of them like Definition and Time-Stamp are like ATTRIBUTES but have fillers that are not in the set of literals. Others are like RELATIONs but are not defined as RELATIONs. Examples are the subsumption links of Is-A and Subclasses and the instance links Instances and Instance-Of. There are also two complexities associated with the RELATION pattern that we illustrate briefly.

2.2.1 The RELATION Pattern

When a slot in an EVENT or OBJECT is a RELATION, the filler of the slot will be another EVENT or OBJECT. This filler should have a corresponding link to the first concept through a different slot. This slot will also be a RELATION that is an Inverse of the first RELATION. This pattern is shown in a simplified form in Figure 8. For simplicity, the first two slots are simply labeled PROPERTY. Only the third slot which is a RELATION is expanded. It may be noted that the RELATION itself is defined as a concept in the ontology. This RELATION, shown in the bottom half of the figure, has only Domain, Range, and Inverse slots.

Additional complexities arise from the facts that the Inverse link may be

- inherited by the second concept from one of its ancestors rather than defined as a local slot in that concept, or
- to a concept that is an ancestor of the first concept, or
- implicit in the domain and range specifications of the RELATIONs involved.

The resulting complex patterns are illustrated in Figure 9 which shows patterns developing from both inheritance and implicit links. Part (a) of the Figure shows an inherited link that does not have a direct inverse. There is a pair of links, HEAD-OF and HEADED-BY, between CORPORATION and PRESIDENT-CORPORATION. The two RELATIONs HEAD-OF and HEADED-BY are inverses of each other. The concept BANK is a descendant of CORPORATION and inherits the HEADED-BY link to PRESIDENT-CORPORATION.

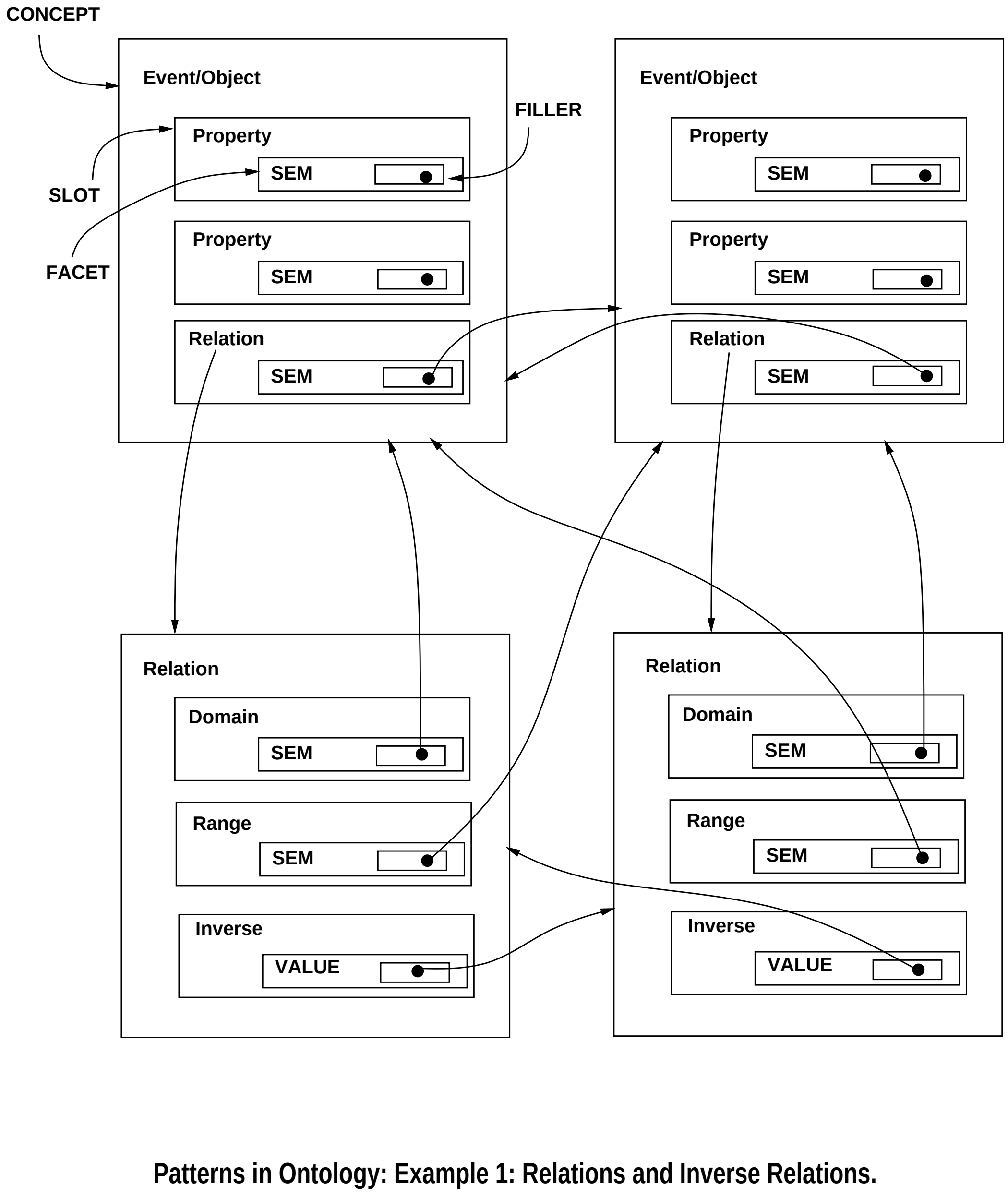


Figure 8: The RELATION Pattern.

Now, if we just look at BANK and PRESIDENT-CORPORATION, we find that BANK has a HEADED-BY link to PRESIDENT-CORPORATION but the inverse link is not to BANK but to its ancestor CORPORATION. Intuitively, we refer to a pattern such as the one in Part (a) of the Figure as an *inheritance triangle.*

Inheritance triangles can occur on both sides of a pair of reciprocal links. For example, Part (b) of Figure 9 shows a pair of relations CORPORATE-ASSET and CORPORATE-ASSET-OF linking the concepts CORPORATION and FINANCIAL-OBJECT. When these slots are inherited down to MONEY (via ASSET) and DRUGSTORE as shown, the resulting pattern has the shape of an X: MONEY has a link to CORPORATION but not vice versa; DRUGSTORE has a link to FINANCIAL-OBJECT but not vice versa. Nor is there any direct link between MONEY and DRUGSTORE. However, these links are implicitly present and a constraint that is looking for them will be satisfied through inheritance. For example, MONEY can fill the CORPORATE-ASSET slot of a DRUGSTORE.

A complex pattern resulting partly from inheritance and partly from implicit inverse links is shown in Part (c) of Figure 9. Here, HUMAN has a slot OWNER-OF which is constrained to OBJECT. This link gets inherited, for example, to TERRORIST. However, there is no explicitly encoded inverse link from OBJECT to HUMAN. Instead, when we look at the PROPERTY concept OWNED-BY (or its inverse OWNER-OF) and examine its Domain and Range slots, we find that they implicitly say that HUMAN can be OWNED-BY of an OBJECT. This information in PROPERTY concepts is in fact extracted and used by the μK analyzer to check constraints. Thus, an OBJECT can be OWNED-BY a TERRORIST although there is no such link from OBJECT to TERRORIST or to any of its ancestors.

Finally, Part (d) of Figure 9 shows another pattern similar to Part (c) but with redundant links. In this case, there is no direct link in either direction between EVENT and OBJECT as far as the pair of RELATIONs THEME/THEME-OF is concerned. These links are implicit in the Domain and Range slots of THEME and THEME-OF. However, there is an explicit THEME-OF slot in ANIMATE, a descendant of OBJECT. This link is redundant, but does not hurt either the consistency of the ontology or the analysis processes in any way. Similarly, there is a redundant THEME link from PHYSICAL-EVENT to OBJECT.

Because the tools we use for concept acquisition are limited in their ability to help us visualize these complex patterns, our ontology does contain several such redundant links which are harmless for our purposes. It may also be noted that, when there is an explicit link, it takes precedence over the information in the corresponding Domain and Range slots. Typically, Domains and Ranges of PROPERTYs contain the most general constraints; we add explicit links between concepts to encode a more specific constraint on a conceptual relation.[11]

2.2.2 The ATTRIBUTE Pattern

In the second subgraph pattern shown in Figure 10, the slot in the concept is an ATTRIBUTE. The filler therefore is a number or a literal. The ATTRIBUTE is also defined as a concept in the ontology. It has only Domain and Range slots.

[11] There are other reasons for adding redundant links. A slot must have a filler. In fact, our acquisition tool does not retain a slot that has no fillers. As such, we are sometimes forced to introduce a redundant filler just to retain the slot. We would want to retain a slot to fulfill lexical requirements for indicating to the analyzer that it should look for a filler for this slot whenever the concept is instantiated.

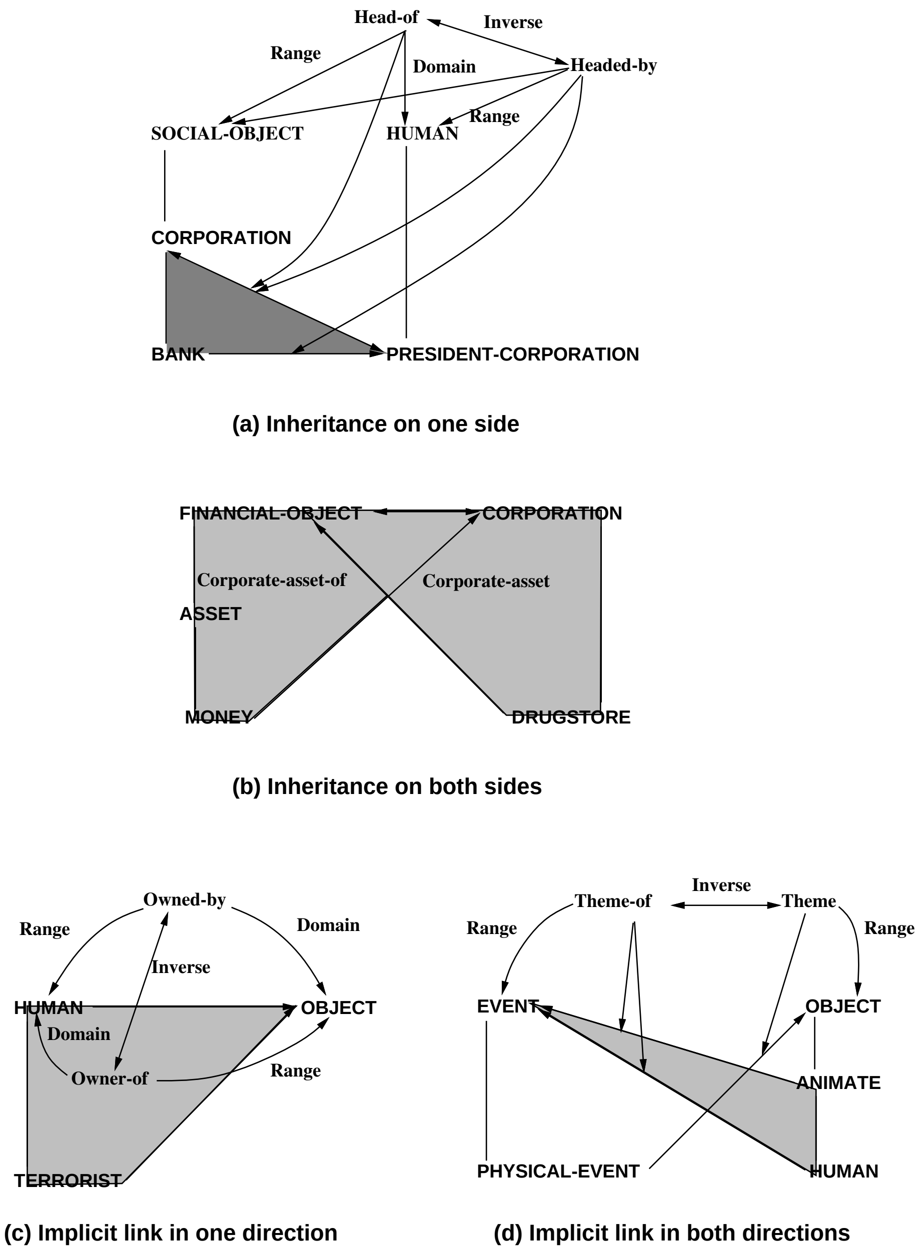
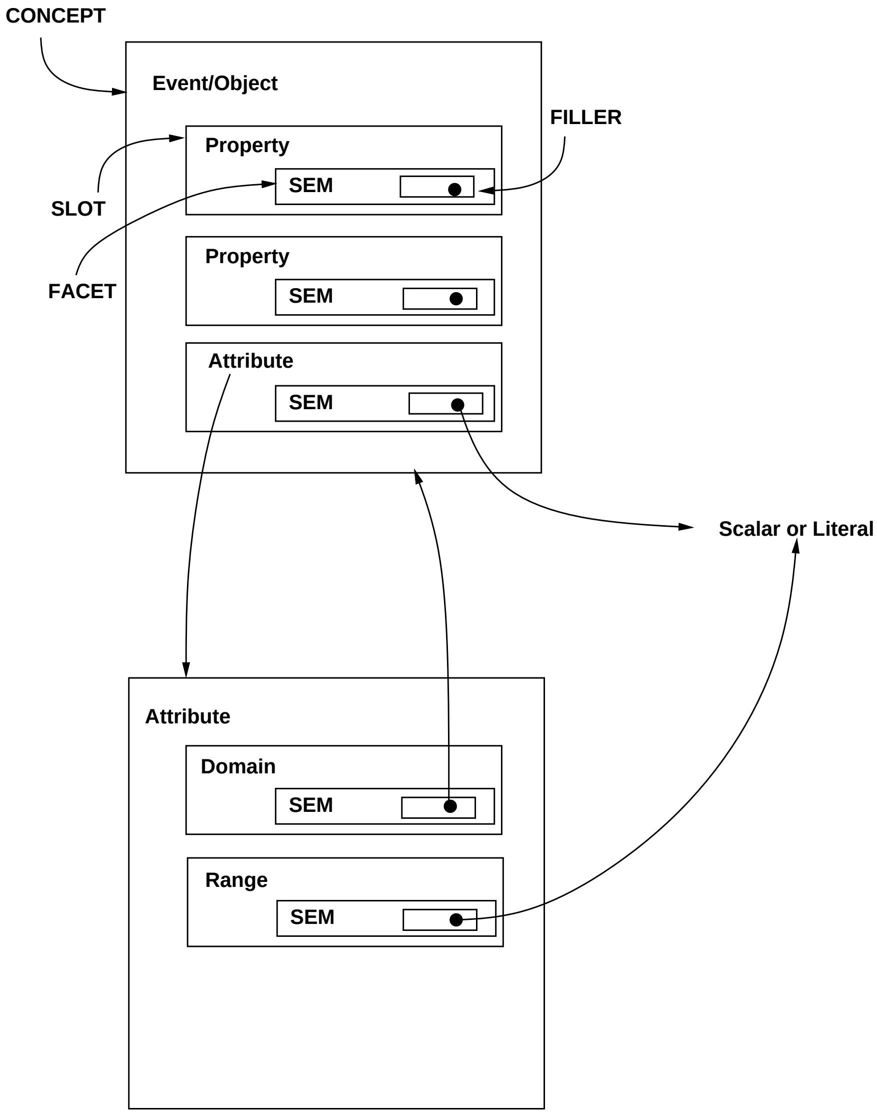


Figure 9: Complex RELATION Patterns



Patterns in Ontology: Example 2: Attributes.

Figure 10: The ATTRIBUTE Pattern.

2.3 A Taxonomy of Symbols

All symbols in the ontology are alphanumeric strings with the addition of only the hyphen character. No accents are permitted on any of the characters. Such enhancements are permitted only in lexicons. As far as ontology development is concerned, all symbols that we encounter can be classified into one of the following types:

- concept names: typically English words with at most four words in a name separated by hyphens.
- instance names: following the standard practice in AI, an instance is given a name by appending its concept name with a hyphen followed by an arbitrary (but unique) integer.
- special slot names: Is-A, Subclasses, Definition, Instances, Instance-Of, Time-Stamp, Domain, Range, and Inverse. These are not defined as PROPERTYs in the ontology.
- literal constants: these are also usually English words, in fact single words most of the time.
- the special symbol *nothing*: used to block inheritance as described below.
- other miscellaneous symbols used in the μK system including:
  - TMR symbols
  - lexicon symbols
  - numbers and mathematical symbols
  - other extraneous symbols (which are detected and eliminated periodically using a set of quality control programs)

Brief listings/glossaries of TMR and lexicon symbols are provided in the accompanying report by Mahesh and Wilson (forthcoming). Ideally, numbers are not present either in the lexicon or the ontology (though for current purposes we may have added some to the lexicon). They are recognized as numbers by a morpho-syntactic analyzer (such as the Panglyzer system for Spanish from the Pangloss (1994) project) and directly incorporated into the TMR. Mathematical symbols such as the multiplication and division operators, * and /, are used in certain slot fillers in the ontology, such as, for example, to represent conversion factors between different MEASURING-UNITs. All other symbols are considered extraneous and should not be present.

2.3.1 Characteristics of Literal Symbols

Literal symbols in the ontology are used to stop unending decompositions of meanings. These symbols are used to fill certain slots (namely LITERAL-ATTRIBUTEs) but are not defined in any way in the ontology. Some characteristics of literal symbols worth noting include:

- Literals symbols are used in our representations in much the same way as the qualitative values used in qualitative physics and other areas of artificial intelligence (AI) that deal with modeling and design of physical artifacts and systems (de Kleer and Brown, 1984; Goel, 1992).

- Literal symbols are either binary or constitute approximate positions along an implied scale. For binary values, it is often preferable to use attribute-specific literal symbols rather than a generic pair (such as ''yes'' and ''no'' or ''on'' and ''off''). The primary benefit of doing this is being able to map the lexical entries of corresponding words in a language to these literal symbols.

- Literal symbols are often used when there is no numerical scale in common use in physical or social models of the concerned part of the world. For example, OFFICIAL-ATTRIBUTE has in its range the literals ''official'' and ''unofficial.'' Although one can talk about an EVENT or a DOCUMENT being more official than another, there is no obvious scale that is in use in the world for this attribute. Thus the two literals seem to serve well in the range of this ATTRIBUTE.

- It is not always true that scalar and literal ATTRIBUTEs correspond to the existence or not of a numerical scale in the physical or social world. A classical example of this is COLOR. Although several well-defined numerical scales exist in models of physics (such as the frequency spectrum, hue and intensity scales, etc.) such a scale does not serve our purposes well at all. First of all it would make our TMRs more or less unreadable by humans if it has a frequency in MHz in place of a literal such as ''red'' or ''green.'' Moreover, it makes lexicon acquisition more expensive; lexicographers will have to consult a physics reference to find out the semantic mapping for the word ''red'' instead of quickly using their own intuitive understanding of its meaning.

- Often, values of LITERAL-ATTRIBUTEs nevertheless will need to be quantified to express the meanings of phrases such as ''light red.'' It is not practical to introduce more and more literals (such as ''light-red'') to solve this problem. As a solution to this, we have introduced two general RELATIONs in the ontology called GREATER-THAN and LESS-THAN. Using this along with a reification operation called *attribute raising*, in a TMR, we can say LESS-THAN ''red'' to indicate a COLOR that is somewhere between ''red'' and the previous literal in the list of literals in the Range of COLOR.

2.4 The Need for Nonmonotonic Inheritance

The special symbol `*nothing*` has been introduced to block inheritance. If a parent has a slot with particular facets and fillers and if one of its children has a `*nothing*` in the Sem facet for that same slot, then the slot won't be inherited from the parent. Since the local slot in the child has `*nothing*` as a filler of the Sem facet, no instance of any OBJECT or EVENT or any number or literal will match this symbol. As such, no filler is acceptable to this slot and the slot will never be present in any instance of the child concept. This has the same effect as removing the slot from the child concept by blocking its inheritance from parent concepts.

There are two reasons why we need to block inheritance using `*nothing*`:

- In a subtree in the ontology, all but a few concepts might have a slot. It is much easier to put the slot at the root of the subtree and block it in those few concepts (or subtrees) which don't have that slot rather than put the slot explicitly in each of the concepts that do accept the slot.

  For example, all EVENTs take the AGENT slot except PASSIVE-COGNITIVE-EVENTs and INVOLUNTARY-PERCEPTUAL-EVENTs. We can put the AGENT slot (with a Sem ANIMAL) in the root EVENT and put a Sem `*nothing*` in PASSIVE-COGNITIVE-EVENT and INVOLUNTARY-PERCEPTUAL-EVENT. This will effectively block the AGENT slot in the subtrees rooted under these two EVENTs while all other EVENTs automatically have the AGENT slot.

- A stronger reason for introducing this mechanism comes from the needs of lexical semantics. Sometimes, lexical mappings of certain words will need to refer to a slot of an entire class of concepts even though a few Subclasses in that class do not have the slot. For example, in the lexical entry for one of the senses of the Spanish word ''activo,'' we must refer to the AGENT of EVENT without knowing which EVENT it is. This requires us to add an AGENT slot to EVENT even though there are two Subclasses of EVENT that do not have AGENT slots. The alternative would be to list every type of EVENT other than the above two in the entry for this word which is not practical at all.

  Another example where we have used `*nothing*` is RUN. All CHANGE-LOCATIONs have a THEME Sem PHYSICAL-OBJECT except RUN because you don't run somebody else. Hence the THEME slot of RUN has been blocked using a `*nothing*`.

In a sense, this mechanism introduces the power of having default slots just like we have a Default facet in a slot. A slot specified for a class of concepts acts like a default slot: it is present in every concept unless there is an explicit Sem `*nothing*` filler in the concept.

2.5 Other Structural Problems

In reality, an ontology should be an *And-Or graph.*[12] This, however, makes inheritance very complex. Our ontology is an ''And graph'' where children with multiple parents must be subsumed by each of the parents. Multiple parents mean a conjunction of each of those parents. There is no way to represent a disjunction of parent links. This can be a problem when a child can have any of several parents but not all instances of the child are subsumed by each of the parents.

Sometimes it is possible to overcome this problem by introducing an ATTRIBUTE as a slot in the concept, instead of classifying further with disjunctive multiple inheritance. For example, we have classified all SERVICE-EVENTs into MEDICAL-SERVE, TRANSPORTATION-SERVE, COMMUNICATION-SERVE, and so on. Some of each of these types of service involve direct interactions with customers and others do not. Rather than classifying each of these concepts into ''medical-serve-customer-end,'' ''medical-serve-back-end,''[13] etc., we introduce a CUSTOMER-CONTACT-ATTRIBUTE as a slot in SERVICE-EVENTs and specify ''yes'' and ''no'' as possible literal fillers. This now allows us to classify the various types of service-events without unnecessarily proliferating the ontology with a large number of concepts. It may be noted here (as illustrated in Carlson and Nirenburg, 1990) that another choice would have been to classify SERVICE-EVENTs first into ''service-customer-end'' and ''service-back-end.'' However, it should be clear that this choice would lead to an even greater number of concepts under this tree. We would now need medical service EVENTs under both ''service-customer-end'' and ''service-back-end,'' and so on.

The above solution of introducing an ATTRIBUTE works when there is no need to encode any other knowledge about that slot or ATTRIBUTE. For example, we assumed in the above example that we do not need to elaborate on what a CUSTOMER-CONTACT-ATTRIBUTE is or how it is related to customers and contacts. If we need to elaborate on such information we need a different solution.

For example, we have classified ORGANIZATIONs into FOR-PROFIT- and NON-PROFIT-ORGANIZATIONs. There are some ORGANIZATIONs such as a LABORATORY which can be of either type. We could not introduce a ''Laboratory-Attribute'' and say ''yes'' or ''no'' for each ORGANIZATION. This is not only awkward, but

[12] See any textbook on artificial intelligence for an explanation of And-Or graphs. For example, see Rich and Knight (1991).
[13] It may also be noted here that such long names may violate the kind of naming guidelines outlined later in this document.

it also does not allow us to represent all the knowledge about laboratories that we want to represent. We do not have the expressive power to make a slot in a concept conditional on the value of a filler in another slot. For example, we cannot associate a ‘‘research-area’’ slot to ORGANIZATIONs and say this slot can be filled only when the ‘‘Laboratory-Attribute’’ slot is filled with ‘‘yes’’. In our representation, all slots represented in a concept are valid and can be filled any time. Moreover, no matter what dimensions (such as profit/non-profit) we choose for classifying a concept into its children concepts, there will often be some children or descendants which do not belong exclusively under any one branch. Since there is no disjunction in subsumption links (Is-A and Subclasses) in the ontology, we cannot make that descendant a child of more than one concept. For example, we cannot put LABORATORY directly under both FOR-PROFIT-ORGANIZATION and NON-PROFIT-ORGANIZATION. Such a thing would be self-contradictory since it would mean that *every* LABORATORY is both a FOR-PROFIT-ORGANIZATION and a NON-PROFIT-ORGANIZATION. Our solution to this problem is shown in Figure 11. This solution guarantees that there is still a place to put all the knowledge that is common to both types of laboratories. However, classifications along multiple dimensions are collapsed into the same level of the hierarchy which could be unintuitive. For example, ORGANIZATIONs are now classified as LABORATORYs, NON-PROFIT-ORGANIZATIONs, and FOR-PROFIT-ORGANIZATIONs which could be read as ‘‘laboratories are neither for-profit nor non-profit’’ though this is not the intended meaning of the classification.

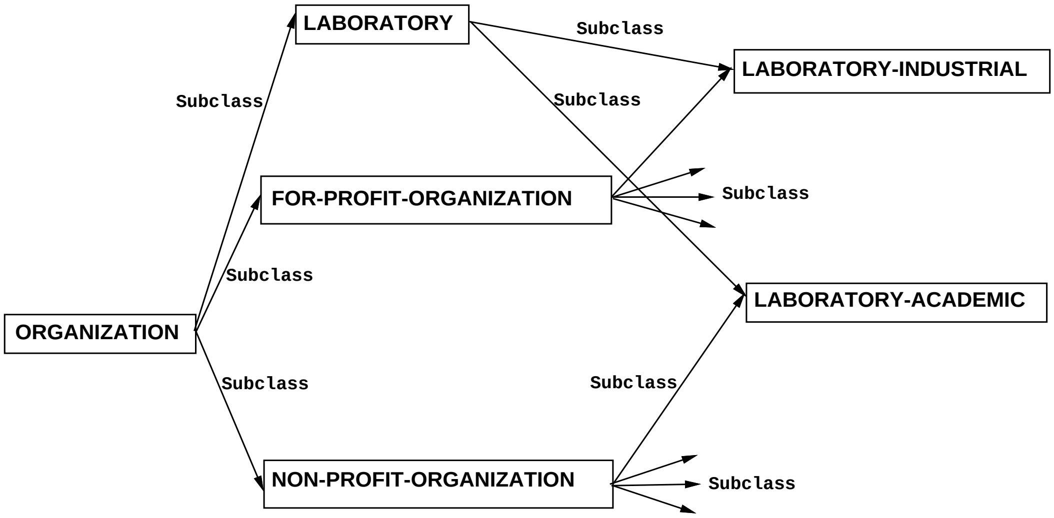


Figure 11: Pattern to Produce Overlapping Subclasses.

In effect, we are saying that when a concept has more than one child, the children do not represent disjoint subsets that cover the parent concept. Rather, they represent overlapping subsets that together cover the parent concept. The relationships between these sets and subsets is shown in Figure 12 for our example of the LABORATORY. While Figure 12 shows the intended meaning of this representation, Figure 13 shows a possible misinterpretation of it. If the ontology developers themselves misinterpret such a representation in this manner (which is possible and in fact likely), they could add a third child to LABORATORY thereby making the interpretation in Figure 13 the only possible interpretation. There is nothing in our current representation or our tools to prevent such an eventuality. In addition, in some other case of a parent and a set of children, if the intended meaning is disjoint classification (as shown in Figure 14), we do not have a representational mechanism to distinguish this from the overlapping cover of a parent shown in Figure 12.

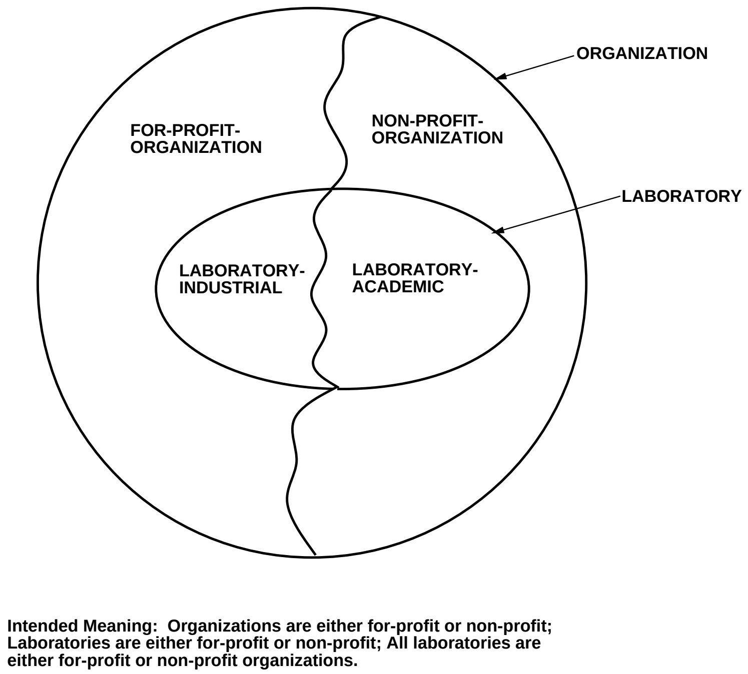


Figure 12: Overlapping Subclasses: Intended Meaning.

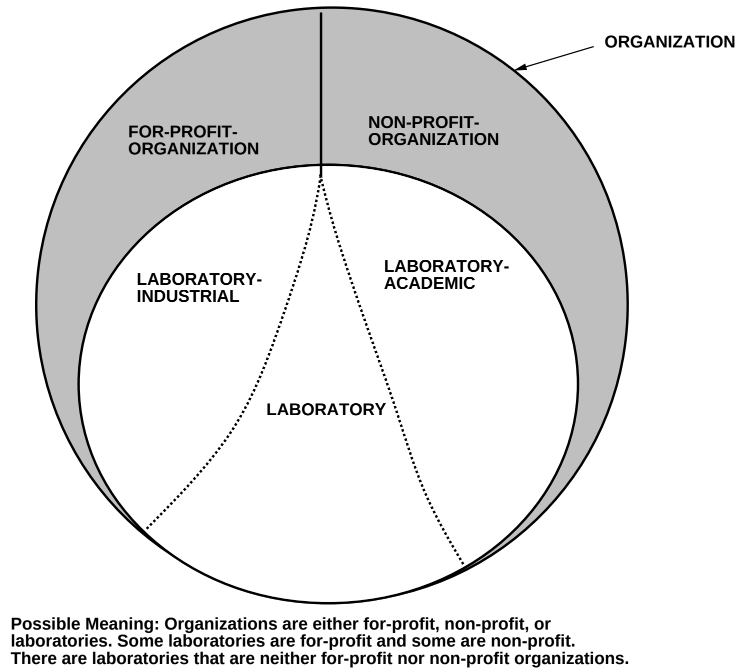


Figure 13: Overlapping Subclasses: Possible Misinterpretation.

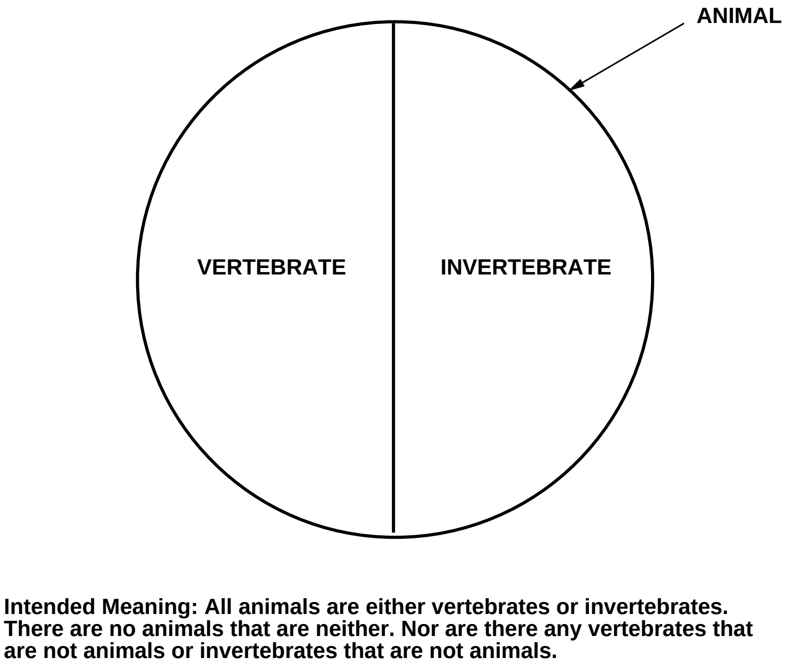


Figure 14: Disjoint Subclasses: Intended Meaning.

One solution to the above problems is to explicitly represent multiple orthogonal classifications in the ontology. For example, we could allow multiple sets of fillers for the Subclasses of each concept where each individual set is a partition (i.e., an exclusive-or, or disjoint classification) of the parent concept into children concepts. The cost of doing this is a significant decrease in the usability and ease of comprehension of the ontology. It will no longer be possible to look at an individual frame in the ontology and understand the full meaning of the concept it represents. (See Section 3.2 for a discussion of limited expressiveness and its virtues.)

2.6 Complex Events: A Proposal and its Rejection

A *complex concept*[14] is one that cannot be represented in a single frame in the ontology. Each concept in the current ontology is represented in a single frame. A complex concept may also involve the need for ontological instances (see below) and variable binding within the complex concept. It is desirable to be able to introduce complex concepts into the ontology for the following reasons:

- Cleanliness of Knowledge Partitioning: A good criteria for partitioning knowledge between lexicons and the ontology is whether the piece of knowledge is specific to a language (and hence belongs in the lexicon for that language) or is general (and hence belongs in the ontology). However, in order to adhere to this rule, we must be able to represent complex concepts in the ontology. For example, in the current ontology, we cannot represent the fact that there must be at least two ORGANIZATIONs for

[14]We use the term complex concept because we might have a need for complex objects at some point in addition to complex events (and be able to use the same representation for both). However, only the representation of complex events is discussed here.

a MERGE to take place.[15] The only alternative at present is to violate the above rule and duplicate this knowledge in the lexicon for each language.

- Inference and Context: In order to take semantic analysis beyond disambiguation using selectional restrictions, we need to be able to model context and make inferences from nascent TMRs. This requires having more detailed representations of situations and therefore complex concepts in the ontology.

A complex event can be represented as follows:

- a head or main frame which bears the name of the complex concept will be attached to the hierarchies just like any other concept (through Is-A links).
- this main frame has a Subevent slot that is filled by one or more fillers that are "ontological instances."
- an ontological instance is an "orphan" in the ontology with no Is-A or Instance-Of link; it hangs off of a Subevent-Of slot from the complex event. However, we need to introduce an Onto-Instance-Of link to attach these subevents to the parent event of their types. This will serve to distinguish them from onomasticon instances.
- the fillers of a Subevent slot are implicitly ordered in sequence. Any other ordering should be handled by temporal relations.
- the head frame will have an Onto-Part slot where we put all additional relations such as temporal or any other reified relation frames that are part of the description of the complex event.

This way, we can represent entire TMRs as complex concepts in the ontology. For example, the concept of a conversation can be represented as follows:

```
Example: CONVERSATION

 (make-frame CONVERSATION
   (Definition (Value "a complex event that involves more than one
                communicative-event between 2 or more humans where
                more than one person does the communication."))
   (Time-Stamp (Value "Not yet created!"))
   (Is-A (Value COMMUNICATIVE-EVENT))
   (Subevent (Value CONVERSATION-COMMUNICATE-1 CONVERSATION-COMMUNICATE-2))
  )

 (make-frame CONVERSATION-COMMUNICATE-1
   (Definition (Value "a subevent of the complex event CONVERSATION."))
   (Time-Stamp (Value "Not yet created!"))
   (Subevent-Of (Value CONVERSATION))
   (Onto-Instance-Of (Value COMMUNICATIVE-EVENT))
   (Agent (Value CONVERSATION-COMMUNICATE-2.Destination))
```

[15] Of course, it is possible to introduce a ‘‘number-of-organizations’’ ATTRIBUTE and fill (> 2) in it, but that is a hack and is redundant with the way we represent sets. Since we employ set notations in our meaning representations, we should be able to use cardinalities of sets to represent the minimum-2 requirement for MERGE. Using sets implies using more than one frame to represent the MERGE concept.

```
    (Destination (Value CONVERSATION-COMMUNICATE-2.Agent))
    (Aspect (Value (duration prolonged) (iteration multiple)))
   )

  (make-frame CONVERSATION-COMMUNICATE-2
    (Definition (Value "a subevent of the complex event CONVERSATION."))
    (Time-Stamp (Value "Not yet created!"))
    (Subevent-Of (Value CONVERSATION))
    (Onto-Instance-Of (Value COMMUNICATIVE-EVENT))
    (Agent (Value CONVERSATION-COMMUNICATE-1.Destination))
    (Destination (Value CONVERSATION-COMMUNICATE-1.Agent))
    (Aspect (Value (duration prolonged) (iteration multiple)))
   )

;; Add temporal relations here to indicate that both subevents occur
;; "during" the other.
```

We can also make full use of the set notation in describing complex concepts in the ontology. For example, for MERGE, we will now be able to say that there must be at least two ORGANIZATIONs that merge. In summary, new machinery needed for the representation of complex events proposed above includes:

- the Subevent and Subevent-Of relations
- the Onto-Instance-Of and Onto-Instances relations
- the Onto-Part and Onto-Part-Of relations.

Subevents are also Onto-Part-Ofs. However, we introduce Subevents to distinguish sub-events from other components of a complex concept and to order them implicitly. We could also introduce any relationships between the subevents (other than a sequence) as needed and add them to the Onto-Part slot.

It may be noted in addition that the dot notation above such as in ''conversation-communicate-2.Destination'' refers to other subevents without any direct link from them. That is, there is no link between ''conversation-communicate-1'' and ''conversation-communicate-2'' above. The above representation assumes that we can get the necessary information by going up the Subevent-Of link and then down the Subevent link. Alternatively, we could introduce an Other-Subevent relation to fill in this information directly in each subevent.

2.6.1 Implications of Introducing Complex Concepts

Introducing complex concepts into the ontology will have many implications for lexicon acquisition and semantic analysis, some of which make it an expensive proposition. Some of the costs of introducing complex concepts include:

- A new semantic analyzer. Significant non-local changes need to be made to the μK semantic analyzer to deal with variable binding, instantiation, and constraint checking with complex concepts.
- Better tools. We will need at least some improvements to our acquisition, browsing, and display tools to deal with complex concepts with new types of links between them.

- New training: Additional training will be needed not only for ontology acquirers but also all the lexicographers and testing personnel.

For example, in the analyzer, instantiation and variable binding become very complex. We get syntax-semantics variable binding information from the complex concept in addition to the lexicon. How do we combine all of this information so that we can use the selectional constraints buried inside the subconcepts of the complex event? Should the lexicon duplicate this information from the ontology and map the variables to syntactic vars or can it provide only partial mappings? For example, the lexicon can provide the mappings for the head concept in the complex event and the analyzer can somehow identify the fillers for the other concepts in the complex concept.

Software tools for acquiring and browsing lexical and ontological knowledge bases also need to be made more sophisticated to handle complex concepts. This will complicate their design, make them more expensive, and perhaps also slow. For example, in the ontology acquisition editor, we should at least be able to see Subevent and Subevent-Of links just as Is-A and Instance-Of links (so that we can use graphical editing operations, such as the *add-link* operation in the Mikrokarat tool, on them).

Given these costs, we have chosen not to introduce complex concepts at this time. It is a desirable enhancement of our ontological representations but not a practical one at the present time. Another justification for this stance is that complex concepts may not be as necessary for machine translation as they are for other NLP tasks, such as question answering or reasoning of some sort.

2.6.2 Ontological Instances

The introduction of complex concepts also entails the need for variable binding across frames in the ontology. For example, the DESTINATION of one Subevent is the same as the AGENT of the other Subevent in the CONVERSATION complex event above. (See Carlson and Nirenburg, 1990 for a second example, the TEACH complex event.) As noted above, such binding can be done by introducing the notion of ontological instances, that is, instances of concepts that reside in the ontology purely for the purpose of variable binding across frames in a complex concept.[16] These instances add to the complexities of representation, browsing, and software design for ontology development in significant ways and hence we have not attempted to include them in the μK ontology.

3 Principles of Ontology Design

Though there is no set of formal principles that determine precisely the structure or content of the ontology we are developing, we can list several principles that constitute the foundation of our methodology and that suggest the guidelines we follow. The principles can also be viewed as the desiderata for the ontology we are developing.

These principles are task oriented or situated unlike the structural principles for ontologies that have been proposed in the literature (Bouaud, Bachimont, Charlet, and Zweigenbaum, 1995; Zarri, 1995). In fact, we show below that many of the structural principles had to be violated in our ontology in order to

[16]The issue of *generic instances* is a separate one; it deals with instances of concepts, simple or complex, such as ‘‘the Big Mac,’’ that do not refer to any particular entity in the world.

meet the practical objectives of simplicity in representation, software requirements, understandability and ease of browsing by non-experts.

3.1 Ontology Development: Desiderata

The μK ontology is based on the following desiderata:

1. An ontology must be language independent. An ontology is not a transfer dictionary between a pair of languages. Our ontology is language independent in two ways:
   (a) It is not specific to any particular natural language such as English or Spanish.
   (b) The concepts in the ontology do not have a one-to-one mapping to word senses in natural languages. Many concepts may not map to any single word in a language;[17] other concepts may map to more than one word in the same language.

   What counts as a concept should be a decision that is quite independent of what words exist in a particular language. An illustration of our approach for deciding what concepts to add to the ontology is provided in Section 7.5 of this report. In the μK project, ontology developers are not Spanish or Japanese speakers so that we hope to balance any Spanish or Japanese bias of lexicographers against an English (or other) bias on the part of ontology developers. This situation where lexicons for two different languages are being developed simultaneously with the development of the ontology is an ideal one for ensuring language independence.

2. An ontology must be independently motivated, not dictated by the lexicon of a particular language. Ontology development is not subservient to lexicography. The two are sister processes that aid each other and at the same time constrain each other in significant ways.

3. An ontology must be well-formed according to a precise, axiomatic specification (such as the one shown in Appendix B) and internally consistent in structure, naming, and content across concepts.

4. It must be consistent and compatible with other knowledge and processing resources such as the lexicon, the semantic analyzer, the language of the TMR, and any expert microtheories.

5. An ontology must be rich in content, conceptual structure, and degree of connectivity among concepts. For machine translation and other NLP tasks, its structure is necessarily rich; it cannot be just a hierarchy of concept names. A concept cannot be just a label; it must have a rich internal structure resulting in a high degree of connectivity with other concepts in the ontology.

6. Limited expressiveness is a virtue. In particular, we reject full first-order predicate logic and the ability to make arbitrary assertions within the ontology. Limited expressiveness is essential for the acquisition of large-scale lexicons that conform to the ontology.

7. An ontology must be easy to comprehend. It must be simple to search for concepts. It must be easy to browse, easy to train acquirers, presentable, and so on. For example, an And-Or tree with disjunctive inheritance is not suitable for our ontology because it is too hard to comprehend and use for both ontology acquirers and lexicographers. Computational complexity of the inheritance algorithm is not the reason for rejecting complex inheritance methods.

[17] Although, for practical reasons, one might want to introduce phrasal lexical entries such as compound nouns that map directly to many concepts in the ontology.

8. An ontology must have a high *utility* for the task it is meant for. It must ultimately aid language processing in resolving a variety of ambiguities and making necessary inferences. Ontology development is a goal driven process: first there must be one or more tasks, well understood ways of using the ontology for the task(s), and an immediate need for the knowledge to be acquired. For example, although the ontology must be language independent, it could be rather language-processing dependent in the following ways:
   (a) It must have all the necessary knowledge to sufficiently constrain language interpretation and generation.
   (b) It must be a richly connected network of concepts to enable a variety of inferences.
   (c) It must support deep but variable-depth semantics. The semantics must be deeper than what is possible with just labels for word senses.
9. The ontology must be cost effective. Not all knowledge that we can lay our hands on should be added to the ontology. Given that we do not have the resources to build a complete encyclopedic ontology of the entire world, we should be acquiring only those concepts and conceptual relations that are of immediate utility in the chosen task(s). This can be monitored in a simple way, in the μK situation for example, by calculating the number of concepts that are not used in any of the lexical entries (and nor are any of the descendents of those concepts). It turns out that in the current μK ontology, the maximum height of an unused subtree in the ontology is just two, indicating that there is no large portion of the ontology that is of no utility for the μK system. This shows clearly that we have in fact been doing a *situated development* of the μK ontology (see Section 7).
10. It must have a limited scope to make its acquisition tractable. The ontology is not an Encyclopedia Britannica; it has a significantly smaller scope than Cyc (Lenat and Guha, 1990).
11. The ontology has conceptual but not episodic knowledge. Acquisition of episodic knowledge (i.e., an onomasticon) is significantly different in scope, methodology, and cost from ontology development. Our methods for concept acquisition are not cost-effective for acquiring instances in the onomasticon. Onomasticon acquisition must be automated to a far higher degree.
12. The ontology is not limited to a domain but does focus on a domain of choice.
13. Ontology development must be technology aided. The task is made more tractable by the deployment of latest technologies: faster machines, color graphical user interfaces, graphical browsers and editors, on-line lexicons, corpora, other ''ontologies,'' semi-automated interfaces for customer interactions and database maintenance, and programs to detect errors and inconsistencies and enforce a comprehensive set of guidelines.

In the situation of NLP, a good way to operationalize the desire for a *limited proliferation of concepts* is to say that the # concepts must be $<$ the # words in any one lexicon. In μK , at present, more than two Spanish words refer to any concept that is used in the lexicon.[18]

The task oriented and situated nature of ontology development can be characterized as an attempt to maximize the ratio:

$$\frac{\#\ concepts\ accessed\ in\ processing}{\#\ concepts\ acquired}$$

[18] This ratio is expected to go up to about 6 once the lexicon is populated by entries generated automatically by means of a derivational morphology engine currently under development.

In a more fine-grained analysis, one can actually consider each slot of a concept (after accounting appropriately for inheritance) to see how much of the actual knowledge encoded in the ontology is used in processing a set of texts. Such an analysis of the μK ontology will be attempted at a later time.

3.2 Limited Expressiveness and Usability

In our practical, situated approach to ontology building, the usability of the ontology in browsing and searching for the right concepts is a fundamental requirement. To this end we have chosen a representation for concepts that is rather limited in its expressiveness. The ontology is essentially a frame hierarchy with multiple inheritance. In general, it is not possible to represent, within the ontology, negations, ternary and higher-order relations (without creating an OBJECT or EVENT), universal and existential quantification or any form of sets, disjunctions, or other arbitrary assertions. Such expressiveness is available to a certain extent in other components of the μK system such as in lexicons and TMRs. Some of the above features are also available in restricted forms in particular parts of the ontological representation (as can be seen from examples in this report).

Methodologically, a primary use of the ontology is in supporting the representation of word meanings in the ontology. Lexicon acquirers must be able to comprehend the ontology, visualize the ways in which it organizes concepts, and find the right concept(s) to construct the representation of the meaning of a word. It is critical in the μK situation that users be able to browse the ontology and see an entire concept together in one place. Therefore, the entire set of slots and fillers for a concept must be represented in one place. The meaning of a concept cannot depend on which set of parents a particular instance has. Nor should it require inferences to be drawn from assertions which may be distributed all over the ontology. Introducing full first-order assertions makes the ontology impossible to browse. For our purposes, we must ensure that the ontology is an entirely declarative body of knowledge that can be browsed statically. We must be able to compute all inheritance effects a priori and let our users see the resulting static description of each concept. Increasing the expressiveness of the ontology beyond simple frame inheritance makes it a black box that can be used only to ask queries and get answers back. Our customers need to visualize the entire ontology, not just derive answers to specific queries.

3.3 Structural Principles and Why We Violate Them

Several structural principles have been proposed in ontology literature as methodological principles for constructing a taxonomy. Instead of starting with a predetermined structure for the ontology and attempting to carve the world to fit that structure, we look at the world (or domain) and decide what is a reasonable structure given the above desiderata. In this section, we take up a set of four principles suggested by Bouaud, Bachimont, Charlet, and Zweigenbaum (1995)[19] and illustrate why we had to violate each of these principles on several occasions. Our methodology relies on guidelines that we developed to help adhere to a set of more complex axioms (see Appendix B) instead of mandating a simple set of structural principles.

The principles suggested include:

- Similarity Principle: This principle essentially says that a child must share the meaning of a parent. Since, in our case, the meaning of a concept is the combination of the information in all its slots, this

[19] Some of these principles are very well known and, in fact, date back to Aristotle.

principle means that a child must inherit all the slots of a parent. As already seen in Section 2.4, we often need to block inheritance using the special symbol `*nothing*`. When we do that, a child does not share the meaning that is part of the parent concept. For example, many ANIMALs have a MATERIAL-OF RELATION to AGRICULTURAL-PRODUCTs. However, HUMANs do not constitute materials of which agricultural products are made. As such, we must block the MATERIAL-OF slot in HUMAN.

- Specificity Principle: This states that the child must differ from its parent in a distinctive way which is the necessary and sufficient condition for being the child concept. Often, in our ontology, there is no distinction represented between a parent and its child. The distinction may not be useful for our purposes in the MT situation. We often cannot justify the cost of formulating and representing all such distinctions. For example, the distinction between ANIMAL and its child concept INVERTEBRATE is not encoded in our ontology.
- Opposition Principle: This states that a concept must be distinguishable from its siblings and the distinction between each pair of siblings must be represented. As noted below in Section 4.2, and illustrated by the pair of concepts WALK and RUN, such distinctions are often unnecessary in our situation and are not represented.
- Unique Semantic Axis Principle: This states that all children of a given concept must differ from one another along a single dimension. As already illustrated with the example of LABORATORY in Section 2.5, this too is unduly restrictive for our purposes. If we were to adhere to this principle, the tradeoff would be to introduce disjunctions among parents when there are multiple parents, a choice that we believe hurts the comprehensibility of the ontology to a far greater extent than not having the semantic axis principle.[20]

4 Ontology as a Sharable Resource

Only ontologies that are constructed as computational entities can be shared effectively. The informational *content* of a computational ontology is much more important in solving practical problems than the *form* in which it is represented. We do not place much emphasis on the representational formalism used in an ontology database. We are open to converting the μK ontology into another format if there is a standard that emerges. Until such time, we must live with different representation formats in addition to different choices of primitives, conceptual relations and configurations.

Given the diversity in ontological designs, a good way to share them as computational resources may be to share a set of tools which provides a common substrate. Various translators can be built upon this substrate for converting an ontology from one representation to another. Such translations of ontologies can then be shared between modules and subgroups of a project, with other projects, and with the community at large. We already have such a system in operation within the μK project where the ontology is routinely translated between two different representational forms and is shared among the large group of people working for the project. We also have the beginnings of other translators such as one that converts TMRs to a template of the kind employed in the TIPSTER text processing initiative.

[20] Surprisingly, Bouaud et. al (1995) argue that there is no need for multiple inheritance and that ontologies should be simple trees. We find this vastly inadequate for our requirements and those of most knowledge representation systems.

4.1 Sharability and the Ten Commitments

Although we are not committed to the format we use for concept representation, we are committed to the following characteristics of our ontology. If a standard emerged either in knowledge representation or in ontological representations such as, for example, the KIF (Genesereth and Fikes, 1992), these are the characteristics that we would consider in deciding whether and how to port the μK ontology to the standard format.

The 10 Commitments:

1. **Broad coverage:** Since our input texts are real-world, unedited news articles, they contain words which have meanings from practically any domain in the world. In order to represent those word meanings in the lexicons and to process the texts in the μK system, the ontology must contain concepts that cover a broad range of topics in the world. A domain-specific ontology, such as one that may be sufficient for database merging or process model integration in a domain, will not serve our purposes. This does not mean that our ontology must contain every conceivable concept in the world before we start using it for machine translation. On the other hand, it must cover many of the commonly used terms from a wide range of domains, leaving out the more technical terms from domains that are not the focus of our texts. For example, we do not need all the terminology used in neuroscience and brain surgery, but we are very likely to need commonly used terms such as BRAIN and SURGERY in our ontology.

2. **Rich properties and interconnections:** One of the biggest uses of our ontology in processing natural language texts is in checking how well selectional constraints are satisfied. In the majority of cases, constraints are not directly satisfied by natural language texts. They are often partially satisfied by each of the possible meanings of an ambiguous piece of text. In order to compare the various choices against each other and determine the best choice, there must be a rich set of connections between concepts in the ontology in the form of a number of properties of concepts. Given any pair of concepts, we must be able to find the best (as in shortest or least cost) path between the two in the ontology. We want our ontology to act not only as a taxonomic classification of concepts in the world, but also very much like a semantic network. The difference between the two types of knowledge bases is shown in Figures 15 and 16 which show the concept CORPORATION with hierarchical information only and in its actual form in the μK ontology, respectively.

3. **Ease of understanding, searching and browsing:** The majority of our customers (lexicographers and testing and evaluating personnel) are not experts in knowledge representation. They are expert linguists or experts in a particular language or domain. It must be easy for them to search for a concept in the ontology, to browse the hierarchies, and understand the relationships between different concepts in the ontology. They should be able to do all this starting with just a rough sense (or gloss) of the meaning they are trying to represent in the lexicon or find in the ontology. In order to meet this goal, we had to lean towards simplicity rather than aim for better expressive power or theoretical cleanliness in resolving a number of issues as noted throughout this report. Choices made for enhancing ease of understanding include:

   - No complex concepts: each concept is represented in exactly one frame.
   - No disjunction in inheritance: a simple, depth-first, conjunctive inheritance algorithm is used; inheritance is precomputed and inherited slots displayed to the user.
   - No default inheritance: default values are not inherited.

- No ontological instances.
- No multiple views: there are no alternative views at the hierarchy, concept, slot, or facet levels.
- No sets and quantifiers: there are no set or quantifier notations in the ontology; such expressiveness is confined to concepts in lexical and TMR representations.
- All slots are equal: there is no distinction between definitional and factual or necessary and peripheral slots; a concept is the conjunction of all its slots.

Searching for a concept is facilitated by a simple string matching technique in our browsing tools. Users can search for a concept by providing a sub-string of its name or any sub-string of its English definition. An example is shown in Figure 17 where all concepts with the string ''food'' in them are listed on the left while all those with ''food'' mentioned somewhere in their definition strings are shown on the right. In addition, our tools enable users to view the hierarchies graphically as well as jump from one concept to any other concept that it is related to.

4. **NLP-oriented:** The μK ontology is designed and built for machine translation. As illustrated in Section 4.2, the kinds of knowledge that are needed in concept descriptions for machine translation may not be the same as what is needed for a reasoning or inference task. Our task is to extract and represent the ''direct'' meaning of the input text using the concepts in the ontology. We often do not need to make elaborate inferences for this purpose.

5. **Economy/cost-effectiveness/tractability:** As in most projects, we have limited resources to build the ontology. We don't have either the time or the number of people to expend several person-centuries to build the ontology. Instead, we must build a broad-coverage, usable ontology in only a few person months. The ontology must be usable at every stage in its development no matter how partial or inconsistent it is.

6. **Language independence:** Concepts in the ontology must not be based on words in any one natural language. Our goal is to derive an interlingual meaning representation from any of a set of natural languages. Although we use English names for concepts, this is merely for convenience and readability. Concept names do not correspond one to one with English words or words in another language such as Spanish. A good example of a language dependent ontology is WordNet (Miller, 1990).

7. **No unconnected terms:** Every concept in the ontology must be related to other concepts in well-formed ways. There should be no disjoint components in the ontological graph. This is necessary since virtually any two concepts might have to be compared for ''closeness'' in checking selectional constraints during NLP.

8. **Taxonomic organization and Inheritance:** These are inevitable for representing word and text meanings succinctly. If we had just a list of concepts not organized in any hierarchical way, lexical semantic entries would be prohibitively long and tedious to acquire.

9. **Intermediate-level grain size:** As illustrated in the discussion on complex concepts (Section 2.6), we do not want either an elaborate decomposition of meanings into a small set of primitives or little decomposition where each word sense is its own atomic concept. We want word meanings to be decomposed to a significant extant so that relationships and commonalities in meanings between words become clear and so that meaning representations are not language specific. Yet, we do not want to enforce a closed set of primitive concepts into which all meanings should be decomposed. The search for such an ''ideal'' set of primitives becomes a research enterprise of its own, has been attempted by other researchers in the past, and does not suit our practical, situated approach well.

10. **Equal status for all properties:** As already noted, there are no necessary and sufficient, or central, or definitional parts of the description of a concept. All properties have an equal status in determining what instances belong to a concept. We want to be able to process texts from any domain using the ontology. It is therefore impossible to determine what is a necessary property and what is peripheral to a concept.

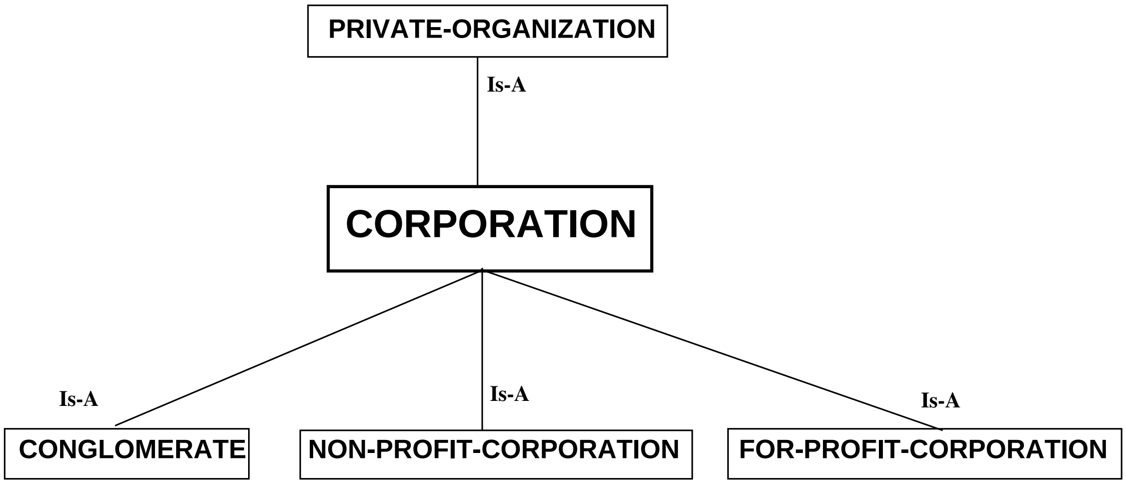


Figure 15: CORPORATION Concept in a Taxonomy Only Ontology.

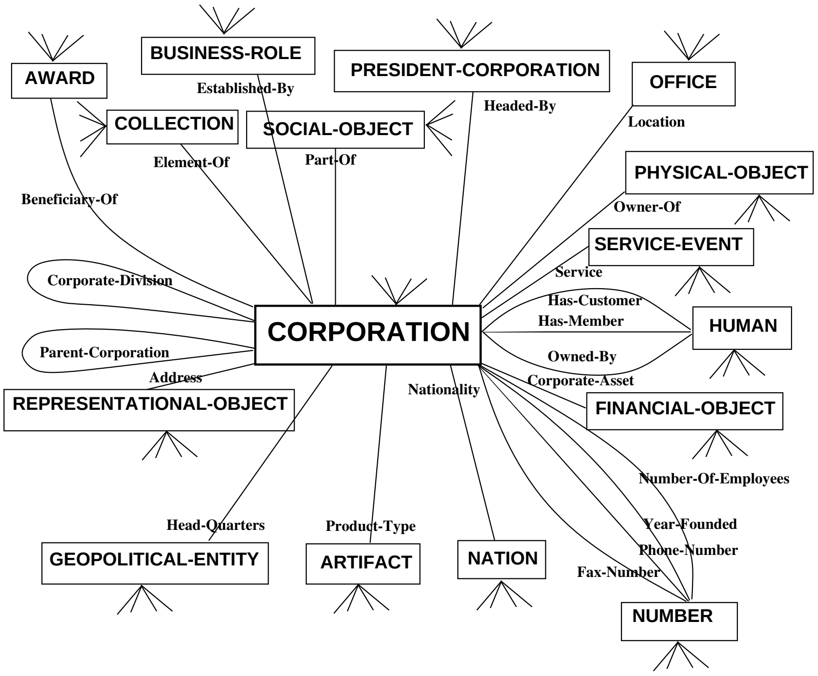


Figure 16: CORPORATION Concept in the μK Ontology Showing Non-Taxonomic Links.

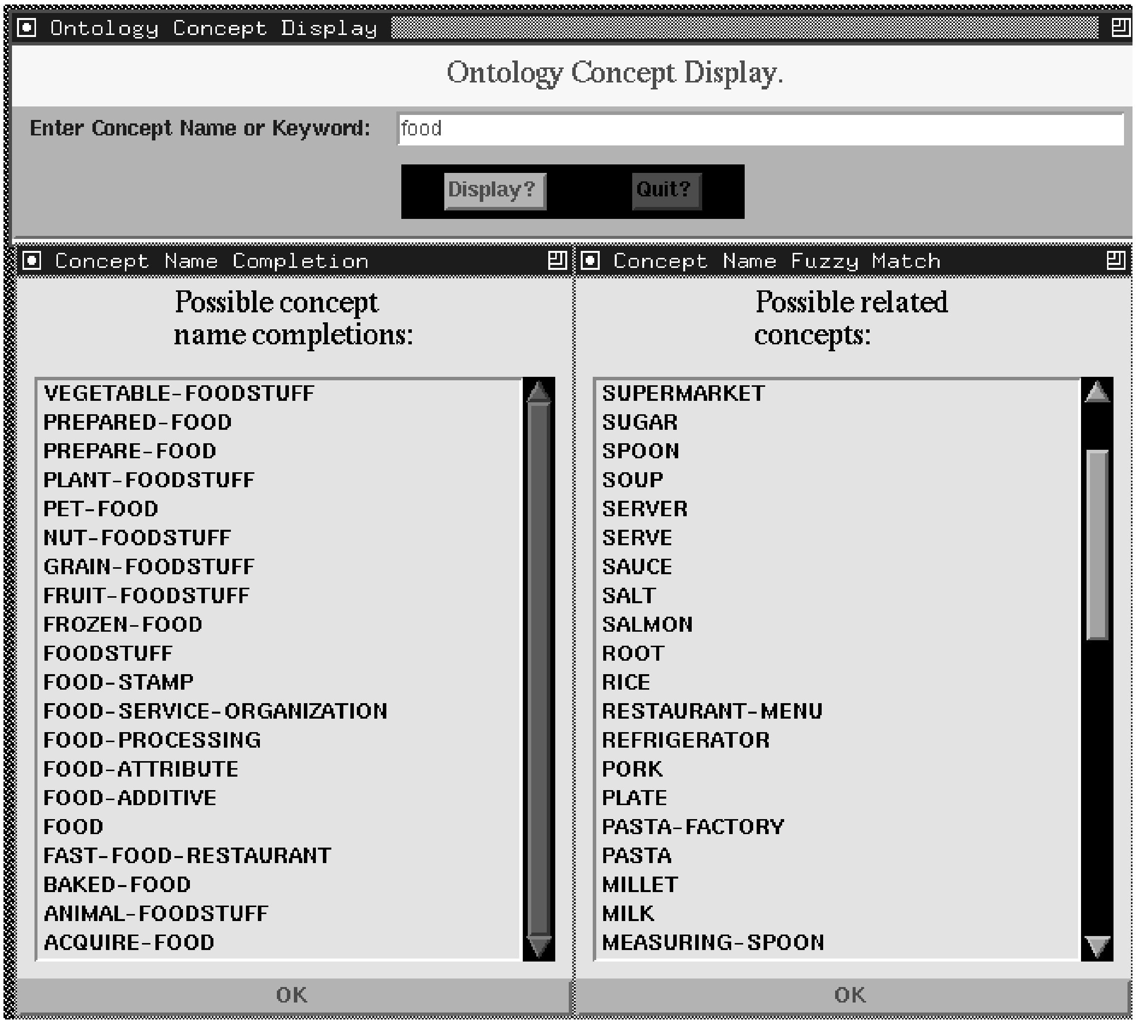


Figure 17: Searching for Concepts Related to FOOD.

4.2 Ontologies for MT $\neq$ Encyclopedia

We believe that ontologies are task dependent to a certain extant. More than the overall task, the types of reasoning that the ontological knowledge is subjected to determines the very content necessary in the ontology. This is true particularly in the case of machine translation since concepts from the ontology are used more to represent linguistic meaning rather than to make elaborate inferences or carry out other nonlinguistic actions. In order to illustrate this point, we show several types of knowledge that we do not include in concepts in the μK ontology but which may in fact be necessary if the task was something other than MT and required strong inference capabilities.

Certain types of knowledge we **do not** need:

- Complex concepts: complex decompositions of concepts may be necessary for other tasks including other NLP tasks. While they may be useful for MT, their cost outweighs their necessity for MT.
- Unnecessary distinctions: we often encode sibling concepts without adding any property to them that distinguishes each concept from its siblings. A concept is often different from its siblings merely by having a different name and being a different node in the hierarchies. This is done to avoid having to encode distinctions in meanings that are not likely to be useful for our task. For example, the real distinction between walking and running is that when humans walk at least one foot is always in contact with the ground whereas running involves moments when both feet are off the ground. This distinction is useless for our purposes since it is unlikely that linguistic forms will differ in a language based on this physical distinction. As such, *Walk* is WALK and *Run* is RUN in the μK ontology.
- Typical, episodic, and procedural knowledge are not included. For example, typical sizes of physical objects are not acquired in our concepts. Although such information may well be useful, it is expensive to determine the right values to put in.
- Scientific knowledge is not included. While we do have concepts for MOTION-EVENTs, FORCE, and ACCELERATION, we have not related them to each other using Newton's second law, for example. It is hard to imagine how Newton's second law would be of sufficient utility for MT to justify the expense of encoding it in our ontological knowledge base.

Given the above differences in the contents of ontologies built for different purposes, we believe that representational differences are not the bigger problems in knowledge sharing efforts (Mahesh and Nirenburg, 1995a; 1996). We could build translators between the different formats that people prefer for representing ontologies; what is more important is to find ways of merging the different types of knowledge that are being encoded in the different ontologies so that they complement each other in useful ways.

5 Ontological Redundancy: A Theoretical Problem

Many guidelines that we are developing for ontology building are derived from common conventions and serve to enforce uniformity and consistency to a certain extent. However, there are certain constraints derivable from the underlying model of the world used to represent concepts in the ontology. In this section, I attempt to elicit some theoretical constraints which may be used to derive concrete guidelines for ontology development.

Our ontology is committed to classifying concepts at the top level into OBJECTs, EVENTs, and PROPERTYs. However, it is by no means an easy task given a word sense to decide if it maps to an OBJECT, an EVENT, or a PROPERTY in the ontology. Our world, and in turn our ontology, seems to support inherent dualities between objects and events, between events and states, between objects and properties, and so on. Let us look into these dualities to see what our choices are and what guidelines we can follow in deciding if we should make a concept an OBJECT, EVENT, or PROPERTY.

5.1 Duality between States and PROPERTYs

Although we do not have states in the ontology, whenever we add a PROPERTY as a slot to another concept, we are in fact representing information about a state. The ontology developer encounters a recurring choice between adding a PROPERTY as a slot to an OBJECT or EVENT and classifying the OBJECT or EVENT further into more specialized concepts. It is important to understand this choice in terms of dualities between OBJECTs and PROPERTYs and between EVENTs and PROPERTYs. We present examples of dualities and discuss the need for properties as well as some constraints on deciding whether to use properties in a particular problem in classification.

We do not plan to introduce states into the ontology. Given that we already have inherent dualities among objects, events, and properties, admitting states (or processes, etc.) into the ontology will only multiply the ‘‘ontological redundancy’’ among the major classes of concepts.

5.1.1 Duality between OBJECTs and PROPERTYs

Consider the meaning of ‘‘the headquarters of a corporation’’ (or any ORGANIZATION for that matter). We have represented this using the HEADQUARTERS RELATION as a slot in CORPORATION. This, for example, is sufficient to represent the meaning of (one of the senses of) the Spanish word ‘‘central.’’

Another choice would have been to further classify ORGANIZATIONs into headquarters and branches. This would be appropriate if the headquarter-branch distinction was a major dimension in classifying ORGANIZATIONs. Given all the other dimensions along which our ontology classifies ORGANIZATIONs (for-profit versus non-profit, service versus manufacturing, etc.), the above dimension is not a prominent one and hence does not merit a top-level position in the ORGANIZATION hierarchy. Choices such as this reflect the underlying duality between OBJECTs and PROPERTYs. Reasonable choices for resolving the dualities can only be made in the context of all the other classifications already (or being) built into the ontology.

5.1.2 Duality between EVENTs and PROPERTYs

There are a number of EVENTs that have a begin phase, a continuous phase, and an end phase. For example, [fall-asleep; sleep; wake-up], [learn; know; stop-knowing], [acquire; own; relinquish], and so on. The continuous phases of these EVENTs are sometimes called states. In our ontology, we do not have a separate class of concepts for states; states are represented as EVENTs in their continuous phase as well as PROPERTYs that modify OBJECTs and EVENTs.

Continuous phases of events can be represented either as EVENTs or using PROPERTYs. For example, we might want to make OWN an event or simply introduce an OWNED-BY property (and its inverse OWNER-OF). To add to the ontologist's freedom, there is also a choice in whether the begin and end phases become EVENTs or are marked using the notation for aspect. For example, we can make OWN a concept and mark acquire as the begin phase of an own. The drawback of doing this, given the limited expressiveness of our ontological representation, is that we cannot represent information about acquire events (which applies only to the begin phase of own, not to its continuous or end phases) in the ontology. Moreover, the begin (or end) phase of the event may be a central concept in the chosen domain. For example, in our domain of company mergers and acquisitions, ACQUIRE is a central concept. We must have ACQUIRE as a separate EVENT so that we can add to it all the domain knowledge about acquisition events. Additional considerations that affect these choices may come from the need to further classify one of the phases of the event. For example, there may be many ways to acquire something and many other different ways to continue to own something that was acquired. The only way to classify particular phases (again given the limited expressiveness of our concept representations) is to make it an EVENT, not a pair of PROPERTYs.

5.1.3 On the Need for Properties in the Ontology

Here is a summary of reasons for introducing a large number of ATTRIBUTEs and RELATIONs into the ontology. From the point of view of NLP, there seem to be at least three reasons why we need PROPERTYs (other than CASE-ROLEs) in the ontology:

1. (Obvious) To modify and enhance EVENTs and OBJECTs in ways other than what CASE-ROLEs (also called thematic roles) permit. CASE-ROLEs capture those aspects of meaning that are typically marked in the surface form of the text linguistically. However, we need to be able to add ATTRIBUTEs such as SIZE and COLOR and RELATIONs such as OWNED-BY and OWNER-OF so that the meanings of different words in a text can be linked together in the meaning representation for the text.[21]
2. To capture conceptual relationships among OBJECTs by providing shorter paths between pairs of OBJECTs such as between the AGENTs and INSTRUMENTs of an EVENT. We need these paths to resolve ambiguities by searching for ''shorter'' paths in the ontology. For example, there must be a closer link between MECHANICs and tools than between all other HUMANs and tools.
3. When a concept can be partitioned into Subclasses in more than one way, we can create Subclasses along only a few of those dimensions. For all other orthogonal dimensions, we use PROPERTYs (often ATTRIBUTE) to mark the concepts. A good example is the HEADQUARTERS RELATION on CORPORATIONS.

While it is true that our ontology is not a simple tree but a plex structure allowing multiple parents, there is a significant constraint on adding multiple parents. A concept with two parents must inherit everything from either parent (and there must not be any contradiction between PROPERTYs inherited from both). In other words, multiple parents stand for a conjunction of those links, not a disjunction. Every instance of the child concept must be an instance of both parents. Going back to the LABORATORY example, we cannot make LABORATORY a child of both ACADEMIC-ORGANIZATION and MANUFACTURING-ORGANIZATION because not every LABORATORY is both though there are some LABORATORYs that are parts of industrial organizations and some that are part of academic ones.

[21] CASE-ROLEs are also PROPERTYs in the ontology; the issue here is the need for PROPERTYs other than CASE-ROLEs.

One way to deal with this is to introduce an ATTRIBUTE to indicate what the classification is instead of introducing Subclasses of concepts. For instance, as already noted, we have classified CORPORATIONs into manufacturing and service corporations. We also want to classify them orthogonally into privately owned and publicly owned (i.e., through shares; let us ignore government controlled ones for the present) CORPORATIONs. The two classifications are orthogonal because both private and public CORPORATIONs can be engaged in manufacturing or service. We can introduce an ATTRIBUTE called PUBLIC-ATTRIBUTE to indicate when a CORPORATION is ''private'' and when it is ''public''.

When should we use ATTRIBUTEs to indicate an orthogonal classification and when should we introduce a plex structure with multiple parents? The decision is based largely on how much information common to the intended classification needs to be placed on the Subclasses below. In the case of public and private CORPORATIONs, if we need to store a lot of information about public CORPORATIONs, then we better have a subclass for that concept. Otherwise, we would end up having to duplicate this information manually for each child or instance that has its PUBLIC-ATTRIBUTE set to ''public''. Given the limited expressive power of our representations, we do not have a mechanism for associating information with all concepts or instances that have a particular (range of) value(s) of an ATTRIBUTE.

5.2 Duality between OBJECTs and EVENTs

We now look at the duality between OBJECTs and EVENTs and show why we must live with it. In order to support word sense disambiguation and the interpretation of non-literal expressions, it is desirable to have direct links among the objects that participate in an event. For example, direct links may be needed between AGENTs and INSTRUMENTs of events. This need is not only legitimate but can actually be taken further.[22] We often need direct links between *any* two semantic roles (e.g., between DOCTORs and HOSPITALs, i.e. between AGENTs and LOCATIONs) and, for that matter, among several roles simultaneously (e.g., between AIRPLANE, landing, RUNWAY, and PILOT). It is not clear whether the EVENT can provide all such links always. For example, are through-the-event links strong enough for shortest-path search (see Figure 1) to serve disambiguation and other semantic microtheories well? (See also Mahesh, Nirenburg, and Beale, submitted.)

One argument in favor of just having the EVENT is that, often, these semantic roles are only connected through an EVENT. Thus, a LABORATORY is all that it is when related to an EVENT such as ''do-science,'' but it is something entirely different with regard to the event ''have-an-office-party.''

A problem with the EVENT only approach is the absence of direct links between the OBJECTs that take part in the EVENT. For example, if OWN is the continuous phase of the same concept of which ACQUIRE and ''relinquish'' are the begin and end phases, respectively, there is no direct link between the owned OBJECT and the owner. Having a property OWNED-BY establishes this link. This leads to a necessary proliferation of properties.

By the same token, however, establishing the PROPERTY OWNED-BY for OBJECTs and removing OWN from the EVENTs will lead to the loss of all links between ACQUIRE and OWNED-BY (in a TMR, for instance). We find this equally undesirable. It is necessary to represent the EVENTs (such as OWN and ACQUIRE), the OBJECTs (such as LABORATORY), and the PROPERTYs that link pairs of OBJECTs and EVENTs (such as OWNED-BY).

[22] This section is based on a summary of a debate on this topic between Victor Raskin, Sergei Nirenburg, and the author. This subsection is adapted from a summary written by Victor Raskin.

This ''ontological redundancy'' results in roughly parallel structures among large classifications of OBJECTs and EVENTs. If we think of ''do-medicine,'' this has many children: ''do-surgery,'' ''do-ophthalmology,'' ''do-gynecology,'' and so on. On the other hand, ''physician'' has the corresponding children, ''surgeon,'' ''ophthalmologist,'' and ''gynecologist.'' Each of the OBJECTs is, of course, the AGENT of a corresponding EVENT. We consider it necessary to have both hierarchies.[23]

6 Constraints on Concept Representation

Several constraints on concept acquisition can be derived from the way we structure the ontology. These ontological constraints not only affect ontology development but they also impose important constraints on how lexical mappings are constructed. Several such constraints are illustrated below. A more detailed discussion will be presented in the accompanying technical report (Mahesh and Wilson, forthcoming).

6.1 PROPERTYs are Not Stand-Alone Concepts

Though we divide all concepts into OBJECTs, EVENTs, and PROPERTYs and though we have frames for PROPERTYs just as for OBJECTs and EVENTs in the ontology, there is an important difference between PROPERTYs and other concepts: PROPERTYs are not instantiated directly. They don't stand alone; their only use is in the form of slots in OBJECTs and EVENTs. PROPERTYs can be reified to get a stand-alone frame in a TMR, but this operation is only done when there is a ''semantic reification'' in the text.

This constraint has several implications on lexical-semantic mappings. For example, no word can map directly to a PROPERTY unless there is a semantically justified reification. Normally, lexicographers must find an EVENT or OBJECT, use the PROPERTY as a slot in the EVENT or OBJECT, and constrain the slot fillers appropriately.

6.2 PROPERTYs Cannot be Fillers in Other Slots of OBJECTs and EVENTs

PROPERTYs cannot be fillers of other slots in EVENTs or OBJECTs in the ontology for the following reason: we will not have instances of PROPERTY concepts under normal conditions to fill the slots. For the purpose of constraining or relaxing possible fillers of a slot, only OBJECTs and EVENTs can be used in the Sem facet (and Default facet, etc.) in the ontology. If we want to allow reified PROPERTYs to be fillers of slots in a lexicon or TMR, we indicate this in the ontology by not having any constraint at all on the slot (i.e., by making the Sem ALL).

For example, to represent the ''specify a need'' sense of the Spanish verb ''precisar,'' we cannot use the EVENT SPECIFY and constrain its THEME to (Sem PRECONDITION) since the filler must then be a PRECONDITION RELATION. Nor can we create a child of SPECIFY in the ontology whose THEME is constrained to be a PRECONDITION. This meaning involves a semantic reification and it must be represented in the lexicon by reifying a PRECONDITION RELATION and using the reified frame to fill the theme slot of SPECIFY.

[23] We may need a formalism to relate one hierarchy to the other. Such a formalism in the form of ''ontological generative rules'' is currently in the works.

6.3 Binary RELATIONs Only

We can only represent binary links. We are forced to decompose ternary and other higher-order relations into binary RELATIONs. This can be a problem. For example, suppose we need to say that a CORPORATION has a certain degree of reputation in a certain AREA-OF-ACTIVITY. We must decompose this into a binary RELATION and an ATTRIBUTE (i.e., a unary ''relation'') between the concepts involved. The binary RELATION maps the CORPORATION to its AREA-OF-ACTIVITY such as ADVERTISEing. The ATTRIBUTE (called REPUTATION-ATTRIBUTE) assigns a scalar (or literal) value to the CORPORATION.

One problem with this decomposition is we cannot say that a CORPORATION has a high reputation in area A and a low reputation for its products in area B, and so on. In other words, the division of PROPERTYs into RELATIONs and ATTRIBUTEs is an approximation. In reality, there are PROPERTYs that are 3-place predicates that relate two concepts as well as quantify the relation. A good example is DISTANCE. A solution to this is to introduce a generic attribute such as AMOUNT that can be added to any RELATION in order to attach a quantity to the RELATION.

Decomposing higher-order relations into sets of binary RELATIONs results in a loss of the ''big picture'' and gives rise to ambiguities when one tries to infer the big picture from the pairwise RELATIONs. This problem is well known in semantic networks. The solution commonly adapted there is to introduce nodes in the network that stand for instances of these higher-order relations. This cannot be done in our ontology since we have a strongly typed network of concepts (i.e., a taxonomic classification plus a network of conceptual relationships) and since we do not reify PROPERTYs within the ontology.

This problem can also be solved by allowing PROPERTYs to have other PROPERTYs as slots. This is done in Cyc (Lenat and Guha, 1990), for example. It results in an order of magnitude increase in expressiveness with consequent tradeoffs in the practical utility of the ontology (see Section 3.2). Another way to look at this is to consider the closed set of facets in our ontology as a limitation. Certain ATTRIBUTEs could be viewed as facets on other slots and thereby solve problems such as the one about reputation above. One can also argue that this has already been done to a certain extent in the μK ontology by including Measuring-Unit among the valid facets. In other words, *there is an inherent duality between* ATTRIBUTE*s and facets.*

6.4 Literal and Scalar ATTRIBUTEs

The classification of all ATTRIBUTEs into literals and scalars is also too rigid to model reality. Often, as already mentioned, a LITERAL-ATTRIBUTE must be quantified in order to meet the expressive power of natural languages. Our solution to this involves (a) treating the set of possible literal values in the range of a LITERAL-ATTRIBUTE to be an ordered sequence, (b) treating each literal as a point on an implicit scale, and (c) using the relations GREATER-THAN and LESS-THAN to refer to positions on this implicit scale that fall in between two literals.[24]

A related problem is one where the range of a SCALAR-ATTRIBUTE is unbounded (often at one end only) but we need to express the value of the ATTRIBUTE in relative terms. For example, in the case of DISTANCE, there is no upper bound on its range. However, to express the meanings of words such as ''near,'' ''far,'' and ''there'' we need to talk about DISTANCEs in relative terms. This would have been easy if the range was bounded (say between 0 and 1). This can be done typically in the case of ATTRIBUTEs that are not

[24] Computational mechanisms for dealing with such modifications of literal values are yet to be worked out in the μK system.

quantified with actual numerical values in a language or in the domain. For example, SIZE itself is not given numerical measure. We only talk about ''big'' and ''small'' although more specific attributes such as AREA and VOLUME are given numerical values. As such, we use a range of (<> 0 1) for SIZE so that we can express the meanings of ''big'' and ''small'' in relative terms ($>$ 0.5 and $<$ 0.5, respectively). This solution does not work for ATTRIBUTEs like HEIGHT since there is no upper bound on its fillers.

Some of these problems suggest a more flexible scheme in which the same concept can have multiple views in an ontology. We could then select a view of DISTANCE that has a bounded range when appropriate and one that has an unbounded one when convenient. Of course, the introduction of multiple views for concepts makes all of our tools much more difficult and expensive to build. It also makes it harder for non-experts to read TMRs and the ontology. As such, although we had included the ability to have multiple views on each facet of a slot, we have only used the ''common'' view on all slots of all concepts and have not pursued this approach. In fact, in recent versions of concept representation, we have even dropped the intervening ''common'' view between a facet name and its filler.

6.5 All Scoping Is over Concepts

All scoping, whether existential or universal, is over entire concepts.[25] We have no ability to scope over the slots in a concept, or the facets of a slot. For instance, we must explicitly specify which slot it is we are referring to. We cannot say OBJECT_1 plays *some* role in EVENT_1. We must explicitly enumerate all possible roles.

This is often a problem even with all the power of the set notation. For example, how do we represent the ''history'' of a CORPORATION? Presumably, it is the (partial) ordering of all (significant?) EVENTs involving the CORPORATION that occurred prior to the current time. We do not know what role the CORPORATION played in all those EVENTs. All we know is that the CORPORATION was somehow involved in those EVENTs. We cannot represent this. Instead, we simply make HISTORY a concept (i.e., an atomic primitive in the representation) and use an appropriate RELATION to link it to CORPORATION.

7 Ontology Acquisition: Situated Development

Ontology acquisition involves deciding what knowledge to acquire, formulating the knowledge according to the principles and guidelines behind the ontology, and then actually representing the knowledge according to the structure and axiomatic semantics of the ontology. In this section, we describe different approaches to ontology acquisition briefly before presenting the situated methodology used in developing the μK ontology.

7.1 Approaches to Ontology Acquisition

The following approaches to ontology acquisition differ from one another mainly in the driving force they use for deciding what concepts to acquire. Concept acquisition can be:

[25]The special symbol `*unknown*` used in lexicons and TMRs provides some power to express the existence of an unknown filler in a slot.

- **Encyclopedia driven:** The best example of this approach is the development of the Cyc knowledge base (Lenat and Guha, 1990). The goal here is to cover an entire encyclopedia rather than the particular conceptual knowledge that may be needed for a domain or a task.
- **Domain analysis driven:** In this approach, a (typically small) domain is chosen and the goal is to *cover* the domain as much as possible. Assumptions and peculiarities of the domain may become an integral part of the ontology. As such, the resulting ontology may be useful only for the chosen domain. Although it is true that most ontologies have a domain of choice, their acquisition may not be entirely domain-driven.
- **Task driven:** In this approach, the focus is on the needs of a task such as database merging or integration of business enterprise models. The resulting ontology may contain only knowledge that was needed for performing a task and hence may not cover a domain fully.
- **Lexicon driven:** In the lexicon-driven approach, the goal is to include all concepts that are necessary to represent the meanings of words in a lexicon. The resulting ontology may incorporate specific nuances in word meanings from a particular language, especially when a lexicon for a single natural language is being developed. In addition, it is often easiest to add a new concept for each new sense of a word so that its lexical semantics can directly map to that concept.
- **Formal:** In this approach, the goal is to adhere to a formalism and to maintain the formal cleanliness of the ontology at all costs (Gruninger, 1995; Guarino, 1995). The resulting ontology may not be large enough to serve any domain or task.

In contrast to the above methods, we have chosen a situated approach where we are driven by the immediate needs of our entire situation, not just one lexicon or one task such as semantic analysis. The concepts we acquire are also immediately put to test in more than one component of the project giving us immediate feedback on their correctness and usefulness. Decisions in lexical semantics and concept acquisition are often made independently by different people with different constraints in mind. In spite of significant differences in constraints, goals, and intuitions, the situated methodology affords a protocol and an environment for fruitful negotiations to achieve total compatibility between lexical and ontological knowledge representations.

7.2 The Situated Methodology

A situated ontology is best developed incrementally, relying on continuous interactions with other knowledge sources. In practice of NLP, this translates to the concurrent development of both the ontology and one or more lexicons through a process of continual negotiation. This negotiation to meet the constraints on both a lexical entry and a concept in the ontology leads to the best choice in each case. It also ensures that every entry in each knowledge base is consistent, compatible with its counterparts, and has a purpose towards the ultimate objective of the task such as producing quality TMRs. The ideal method for situated development of knowledge sources for multilingual NLP is one where an ontology and at least two lexicons for different languages are developed concurrently. This ensures that the ontology is truly language independent and that representational needs of more than one language are taken into account.

Ontology acquisition is a very expensive empirical task. Situated development is a good way to constrain the process and make it attainable. For example, in the NLP situation, the acquirer must focus on concepts in the domain of the input texts (i.e., a corpus) and thereby increase the ratio of the number of concepts (and their PROPERTYs) that are actually used in processing a set of texts to the total number of concepts (and their PROPERTYs) present in the ontology. The best example of a large ontological database acquired with enormous efforts but entirely outside the context of a particular task is Cyc (Lenat and Guha, 1990). While the utility of Cyc in a particular situation such as large-scale NLP is yet to be demonstrated, it is also true that most projects cannot afford to spend as many resources as it has taken to develop Cyc and must strive to constrain acquisition significantly or share existing ontologies. While it is conceivable that a comprehensive store of knowledge such as Cyc will enable a system to make almost any inference possible, it has not yet been demonstrated that the approach of Cyc is useful or even necessary for text analysis. It is not clear, for example, that concepts in Cyc which are built without a task (such as text analysis) in mind constitute useful primitives for representing text meanings. A system based on Cyc will have overtly to incorporate all inference rules necessary for solving a concrete problem, such as, for example, machine translation. In general, the formidable task of encoding all human knowledge and all the conceptual relations between its many parts is a largely intractable endeavor.

In addition to situated development, ontology acquisition can be made more tractable by partial automation and by the use of advanced tools. However, the kind of language-independent ontology described above, where each concept has a rich structure and relationships to other concepts, is almost impossible to acquire automatically from a corpus of texts that has been tagged on mere syntactic or superficial semantic features. It is conceivable, however, that an ontology can be acquired incrementally from a corpus where each text is tagged with an entire meaning representation (TMR) for the text.

Automation of ontology acquisition appears to be feasible only in the ‘‘word sense’’ approach which equates a concept in the ontology with a word sense in a particular language (Knight and Luk, 1994). Such an ontology is language dependent and is inseparable from the word senses in the lexicon of a particular language. Moreover, an ontology of word senses is also much larger than one with language-independent concepts, since multiple word senses with common meanings are not decomposed into a smaller number of concepts in this approach (in SENSUS and WordNet databases, for example; see Knight and Luk, 1994; Miller, 1990; see also Skuce, 1995). Due to the large size and automated acquisition of such ontologies, they contain almost always only taxonomic and perhaps partonomic relationships among concepts. They do not have the richness of interconnectivity through conceptual relations that is a hallmark of the μK ontology.[26] While the adherents of this method claim enhanced automaticity of acquisition, in practice it hasn't been shown that the material acquired automatically is appropriate even as the basis for a bilingual dictionary, let alone an interlingual meaning representation, to support transfer across languages in tasks such as machine translation.

We believe that ontology development must not be controlled by the requirements of the lexical specification of the semantics a language. Nevertheless, our approach *is* application oriented being driven by the needs of a practical task (machine translation in our case). The application is the defining factor in pinpointing the grain size to which a word meaning must be decomposed in the ontology. Our methodology for ontology development strongly discourages direct encoding of word senses as ontological concepts and at the same time constrains the developers from getting bogged down in unending decompositions of word meanings into deep underlying concepts and conceptual relations. In both ways, we have been successful in limiting the size of the ontology as well as the effort required to acquire it and maintain its quality. We have developed a set of guidelines to enforce such constraints on ontology acquisition. An important guideline we have employed is not to decompose complex EVENTs or OBJECTs unless we see a practical need from a problem in analyzing an actual text for doing so. By taking this practical approach we do not have to expand

[26] Parts of this section are based on work presented by the author and Sergei Nirenburg at the 1995 meeting of the Society for Text and Discourse in Albuquerque, NM.

the ontology significantly by decomposing the structure of a complex event such as TEACH or a complex object such as an AUTOMOBILE. While this approach may limit the ability of a system using our ontology in making general inferences (say about the internal workings of an AUTOMOBILE), we contend that such completeness in inference capabilities is not essential for tasks involving text analysis such as machine translation (albeit knowledge-based). Nor do we need our knowledge bases to encode *all* the conceptual relations between TEACHing and other COMMUNICATIVE-EVENTs and social situations. While we certainly admit that such knowledge if available can be useful in making certain inferences or even in resolving certain problem in analyzing a text, we emphasize the point that manual acquisition of such knowledge is inherently intractable and that we can achieve reasonable success in building text analysis systems by limiting ontology development efforts according to our guidelines. Our experience shows that the limited ontology we developed is still invaluable in solving problems such as resolving word sense ambiguities.

Our methodology has two main aspects that help us conform to the ten commitments listed above (Section 4.1): *situated development* and formulating and adhering to a set of *guidelines.* Briefly, the benefits of situated development are:

- Makes acquisition tractable.
- Increases connectivity among concepts since lexicographers suggest a number of properties for concepts.
- Enhances ease of understanding and searching the ontology.
- Increases the ratio: $\frac{\#\ concepts\ accessed\ in\ processing}{\#\ concepts\ acquired}$

Figure 18 depicts the process model of ontology acquisition in the Mikrokosmos situation showing information flow and interactions among the ontology developer, lexicographer, domain expert, analyzer developer, tester, and the various tools and databases. The primary source of information for concept acquisition is the flow of requests for additions and changes from lexicographers, testers, and analyzer developers. Domain experts, other ontologies, domain dictionaries, and published standards (e.g., SICM, 1987) also provide useful information when available.

One can also reverse the methodology and populate the lexicon and the ontology in two other ways:

- Acquire concepts not requested by lexicographers to fill gaps in the ontology. For example, if several concepts for types of FURNITURE were requested, some other types that were not requested may be added. Such gaps are often obvious to the ontology acquirer. In fact, concepts acquired to fill gaps are often requested by lexicographers at a later time.
- The lexicon can be populated by acquiring all words that can be mapped to a concept. For example, when a word is mapped to a concept and added to the lexicon, other words that map to the same concept may also be acquired. This can also be done starting from concepts that were acquired to fill gaps in the ontology.

7.3 Technology for Ontology Development

In order to aid ontology acquisition and maintenance, to check its consistency, to monitor its quality, and to support interactions with lexicographers, a variety of semi-automated tools have been developed and deployed in the μK project. Tools are in use for:

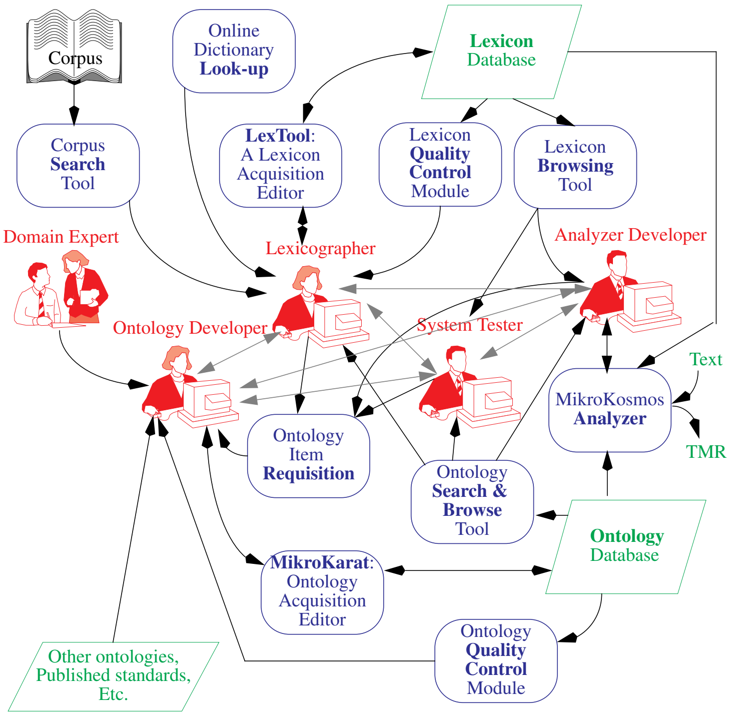


Figure 18: Knowledge Acquisition in the Mikrokosmos Situation.

- browsing the hierarchies and the internals of concepts in the ontology;
- graphical editing support: the ‘‘Mikrokarat’’ tool[27] supports complete functionality for editing the graph structures in an ontology;
- translating between two different representations: the object-oriented one suitable for computational purposes and the plain-text representation that is more suitable for certain other programs and for manual search and maintenance purposes;
- various types of consistency checking both within the ontology and with lexicons, and for conformance with the guidelines;
- supporting interactions with lexicon acquirers through an interface for submitting requests for changes or additions (see below).

This set of tools is being shared across geographical, disciplinary, and project group boundaries on a daily basis. For example, on a typical day, 8-10 people browse the μK ontology and one or two people develop it using the Mikrokarat tool.

7.3.1 The Customer Support Interface

Given the volume of requests the ontology developers get from lexicographers and other customers on a typical day, it is not practical to rely solely on personal or unformatted electronic communication. We have developed a simple form-filling interface that ontology customers can access through the browsing tools and submit a variety of questions or requests to the ontology group. This graphical user interface is being used routinely to submit requests for changes or additions to concepts in the ontology. This tool saves the request in a useful format and immediately prompts the ontology developers by sending an electronic mail to them. It also logs the requests for later reference.

A snapshot of the main window of the customer support interface is shown in Figure 19. This interface can be reached from an ‘‘Onto-Complain’’ button in the ontology browsing tool. A similar interface is also available for submitting questions and requests to the lexicon team.

7.4 Distinctions between the Ontology and the Onomasticon

It must be noted here that the manual acquisition methodology described here applies mainly to the acquisition of concepts. Instances of concepts such as information about people, PLACEs, ORGANIZATIONs, and important EVENTs are assigned a separate store, called the *onomasticon,* and its acquisition is designed to be essentially automatic with the potential of bootstrapping by feeding back text meaning representations produced by the analyzer to augment the store. Onomasticon acquisition is much more amenable to automation and is very expensive if done manually. The methodology for acquiring a large-scale multilingual onomasticon starting from sources such as various proper name lists, gazetteers and so on is currently under development. This methodology has a lot more in common with lexicon acquisition than with ontology acquisition and as such will not be outlined here. It is worth noting, however, that certain classes of instances such as GEOPOLITICAL-ENTITYs were available to us attached to previous ontologies

[27] Developed by Ralf Brown at the Center for Machine Translation, Carnegie Mellon University.

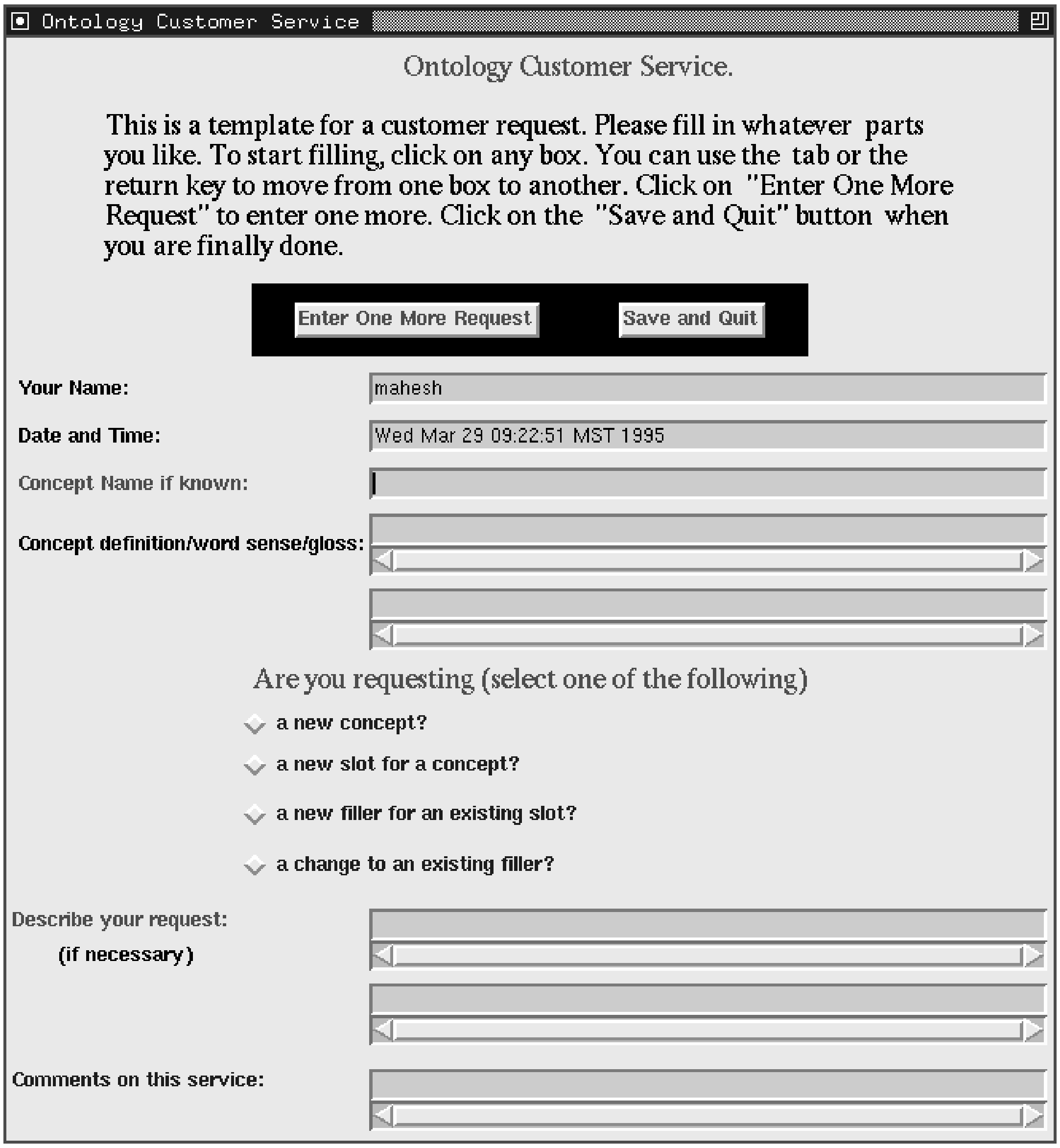


Figure 19: The Customer Interface: Main Window.

and have been retained in the database containing the μK ontology. They are clearly distinguishable from concepts and must be considered part of the onomasticon. It is, however, impractical to represent all instances as full blown frames in the onomasticon, given the large number of proper names (such as names of ORGANIZATIONs, people, etc.).

7.5 Case Study: What Concepts to Add to the Ontology

In this case study, we take the example of English verbs ‘‘swim’’ and ‘‘float’’ and show how we decided what concepts to add and how to relate them to other concepts. In particular, we show how ontology developers could do a detailed and expensive analysis of the science behind the meanings of these words but we choose not to do such an analysis in most cases.

A possible first step in deciding what concepts to add to represent the meanings of ‘‘swim’’ and ‘‘float’’ is to consider their meanings (intuitive or the ones you may find in a dictionary) as well as those of equivalent words in several other languages. For example, Figure 20 shows the equivalent words in Spanish and Russian. It is clear that some languages including English and Spanish have separate words for ‘‘swim’’ and ‘‘float’’ but Russian does not. This information does not tell us whether to add just one concept (say LIQUID-CONTACT-MOTION) or two separate ones. On the other hand, if we had looked at just English or just Russian, we were more likely to go wrong; now we at least realize that we need to do more analysis before we can decide what concepts to add to the ontology.

English: Swim ⇒ SWIM; Float ⇒ FLOAT
Spanish: Nadar ⇒ SWIM; Flotar ⇒ FLOAT
Russian: Plyt ⇒ SWIM or FLOAT; Plavat ⇒ SWIM or FLOAT.

Figure 20: Words for SWIM and FLOAT in Different Languages

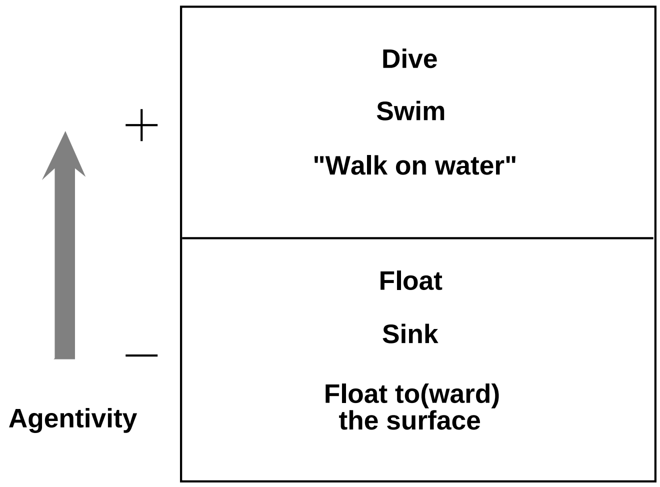


Figure 21: SWIM and FLOAT: The ‘‘Agentivity’’ Dimension.

The next step is to consider some related meanings and analyze the dimensions along which the meanings differ from one another. For example, Figure 21 shows the dimension of "agentivity" (i.e., being active or having a causal role in an EVENT versus being a passive undergoer) along which we immediately realize that ‘‘swim’’ is active while ‘‘float’’ is not. This dimension also suggests related meanings such as ‘‘dive,’’ ‘‘sink,’’ and so on.

Lexicographers and linguists do a different kind of analysis. They look at different meanings (or senses) of the same words and the linguistic contexts in which those words occur. For example, they come up with meanings of ‘‘float’’ such as in "The meat is swimming in gravy" or "John floated the raft out of the dock" and argue about what distinct senses of float exist. The danger in ontologists taking the same approach is that the resulting ontology is highly likely to be specific to the language in question. Ontologists should use whatever insight linguists might provide but they must look at physical, domain, or inherent distinctions between meanings quite apart from how corresponding words are used in a language.

The ontologist might not be able to decide from the ‘‘agentivity’’ dimension alone whether to add one concept or two for ‘‘swim’’ and ‘‘float.’’ At this point, other dimensions that could distinguish between ‘‘swim’’ and ‘‘float’’ immediately come to mind. It may be noted nonetheless that the exercise has already produced a benefit: it led the ontologist to think about related meanings, such as ‘‘dive’’ and ‘‘sink,’’ which can also be acquired at this time proactively (i.e., without being requested by customers). Such proactive acquisition has proved very effective in our experience. Requests for new concepts tapered off earlier than expected in the evolution of the μK ontology and yet, there are no large unused parts in the ontology.

At this point, the ontologist can easily get carried away and spend a lot of time and effort doing a detailed analysis of the microworld of ‘‘swim’’ and ‘‘float.’’ For example, Figures 22, 23, and 24 show the addition of three more conceptual dimensions---specific gravity ratio between the object in motion and the substrate, the direction of motion, and the upward/downward orientation of motion---to distinguish between ‘‘swim’’ and ‘‘float.’’ While such domain analyses may be interesting, useful, and even lead to (re)discoveries of laws of nature (such as that if the specific gravity of the object is higher than that of the substrate and the object is passive, it is impossible for the object to move upward, as shown by the dark regions in Figure 24), they are certainly very expensive. In a situated methodology, we cannot afford to conduct such detailed investigations whenever we have to decide what concepts to add to the ontology.[28]

Our analysis of ‘‘swim’’ and ‘‘float’’ led to the construction of a MOTION-EVENT hierarchy which is shown in Figure 25. It may be noted that we in fact decided to have separate concepts for ‘‘swim’’ and ‘‘float.’’ These concepts are both classified under HORIZONTAL-LIQUID-CONTACT-MOTION, but are distinguished only along the ‘‘agentivity’’ dimension. We did not acquire properties of these events with respect to specific gravity ratios, since such information is unlikely to be of sufficient value for the MT task or in the domain of company mergers and acquisitions to justify their acquisition. Also shown in Figure 25 are the EVENTs WALK and RUN, which are not distinguished along any dimension in our ontology (as explained in Section 4.2).

7.6 Guidelines

The basic methodology for concept acquisition employed in the μK project involves a fine-grained cycle of requests for concepts from the lexicon acquisition team and the resulting responses which may involve

[28]The analysis of ‘‘swim’’ and ‘‘float’’ shown above was an exception; we were asked to analyze these meanings for presentation and discussion at the Workshop organized by the International Federation of Information Processing, Special Interest Group on Lexical Semantics, April 28-29, 1995, Philadelphia, PA.

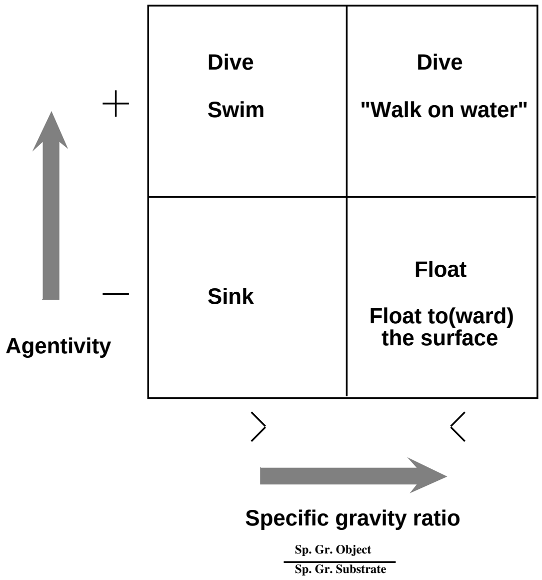


Figure 22: SWIM and FLOAT: The Specific Gravity Dimension.

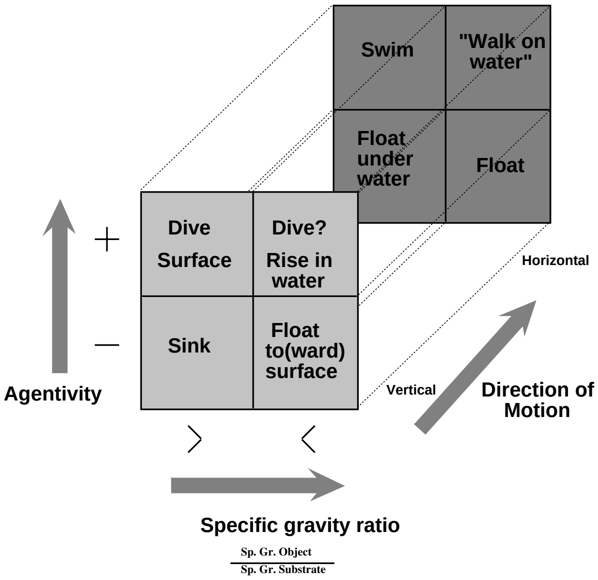


Figure 23: SWIM and FLOAT: The Vertical/Horizontal Dimension.

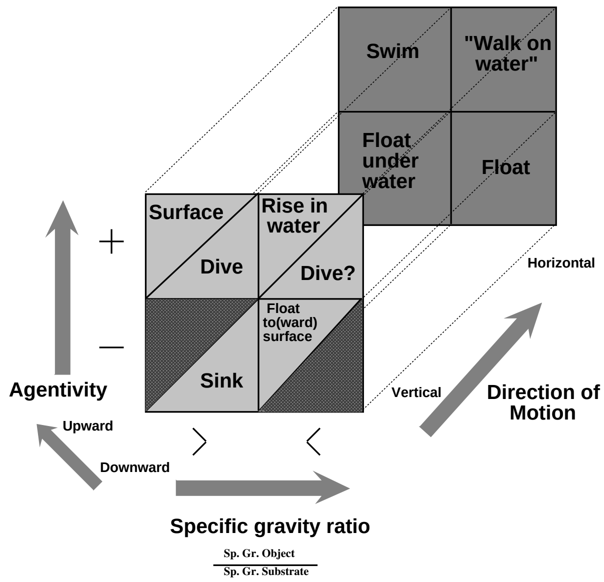


Figure 24: SWIM and FLOAT: The Upward/Downward Dimension.

pointing out an existing concept, adding a new concept, enhancing the internal structure of one or more concepts, or suggesting a different lexical mapping for the word in question. If it is determined that a word sense requires a new concept in the ontology, the ‘‘algorithm’’ applied for adding the new concept hinges on viewing the ontology as a discrimination tree. The acquirer discriminates from the top down until at some point there is no child that subsumes the meaning in question. A new concept is added as a child at that point.

In the μK project, sets of guidelines have emerged for making various kinds of decisions in ontology acquisition. These guidelines, some of which are shown below, collectively define the methodology for ontology building. The following is an evolving set of guidelines and is largely derived by generalizing from the decisions and challenges we faced in our experience.

Guidelines have been developed in μK for deciding:

- what to add as a concept; what not to add.
- where to place a concept;
- what to name a concept;
- how to write a definition string;
- how to rearrange a part of the ontology; and
- possible actions for a lexicon request.

Samples from the various sets of guidelines are shown below. A more complete set can be found in the accompanying report by Mahesh and Wilson (forthcoming).

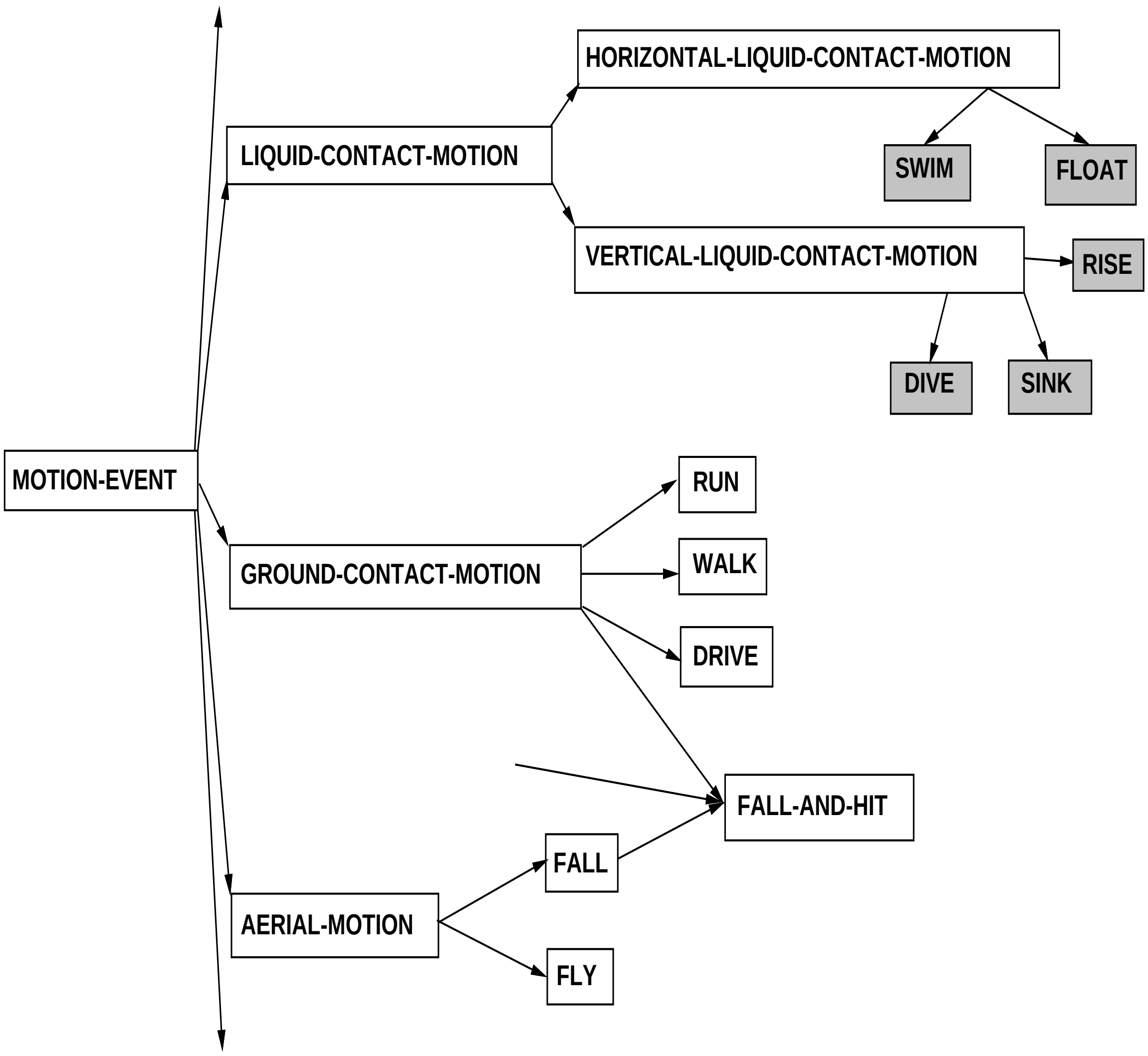


Figure 25: A Classification of MOTION-EVENTs in the μK Ontology.

7.6.1 Guidelines for Deciding What Concepts to Put in

It is very important in building an ontology to consider the ontological status of a symbol and decide whether the symbol should be a concept in the ontology. Often, a quick and easy solution to a problem in representing the meaning of a word or in getting the language processor to work on an input is to throw a symbol into the ontology as a concept. An ontology builder must resist this temptation and add only those symbols as concepts that deserve to be concepts in the ontology. In addition, an ontology developer is often under pressure from a lexicographer and sometimes also from an analyzer builder to provide a ‘‘concept’’ for a symbol that does not really fit as a concept, or is not really different from another concept, or is redundant with extra-ontological representational apparatus already in the system (such as sets, attitudes, aspect, etc.). Reckless addition of dubious ‘‘concepts’’ will inevitably lead to a database of poor ontological quality.

One such decision that the ontology developer has to make is whether a given symbol becomes a concept or an instance. The following rules of thumb help in deciding whether a symbol is a concept or an instance:

- **Instance-Rule1:** See if the thing can have its own instance. If yes, it is a concept. If not, it is an instance. Instances do not have their own instances.
- **Instance-Rule2:** See if the thing has a fixed position in time and/or space in the world. If yes, it is an instance. If not, it is a concept. For example, SUNDAY is a concept, not an instance, because it is not a fixed position in time (‘‘last Sunday,’’ ‘‘the first Sunday of the month,’’ etc.). On the other hand, ‘‘Paris, France’’ is an instance because it is a fixed position in space.

A few other guidelines for deciding what to add to the ontology are shown below:

1. Do not decompose and add further concepts just because you can. In general, if we do not need a concept for our immediate goals in language processing, we do not want to put that concept in. Because ontologies are largely unbounded, it is important for us to focus our efforts on building those parts that we need immediately.

   For example, EVENTs like BUY or MARKETING can be decomposed to a great extent. However, unless we have an indication that we need the details of such decomposition for our task, we do not decompose the concepts.

2. Do not add a concept if there is already one ‘‘close’’ to or slightly more general than the meaning in question. Consider the expressiveness of the representation provided by gradations (i.e., ATTRIBUTE values) before adding separate concepts.

   For example, we do not need separate concepts for ‘‘suggest,’’ ‘‘urge,’’ and ‘‘order.’’ They are all gradations of the same concept, a DIRECTIVE-ACT, with various degrees of force which can be captured in an appropriate ATTRIBUTE.

3. Do not add specialized EVENTs with particular arguments as new concepts (unless they are significantly different from the existing concepts).

   For example, we do not need separate concepts for ‘‘walk-to-airport-terminal’’ and ‘‘walk-to-parking-lot.’’

7.6.2 General Guidelines and Constraints

Here is a subset of a growing list of general guidelines for ontology development.

1. If you add a slot, the slot name is probably already a PROPERTY in the ontology. Verify this and consider adding the PROPERTY to the ontology. If the slot is a RELATION, consider also adding the Inverse link from the filler back to the concept. Add the inverse RELATION if it does not exist already.
2. Keep inheritance in mind. If most siblings have a common slot and filler, that piece of information belongs in their common parent. Do any change at the highest level possible in the hierarchy so that it propagates through inheritance to all the desired places. For example, to block a slot using `*nothing*` do it at the highest level appropriate.
3. It helps to keep our ultimate objective in mind. We are building the ontology for KBMT. If a concept or its further decomposition does not seem necessary for KBMT, let us not spend time adding it. We are not building an encyclopedic knowledge base like Cyc (Lenat and Guha, 1990).
4. Keep other languages in mind. Just because English names are used for the concepts in the ontology, we must not forget that the ontology must be equally useful for interpreting or generating any of a set of natural languages.
5. Create children concepts if you can clearly classify instances of a concept. If it is a continuous difference, use an ATTRIBUTE. For example, to work with concepts concerning aesthetic ARTIFACTs, such as PAINTINGs, we do not create a concept "aesthetic-artifact." Instead, we create AESTHETIC-ATTRIBUTE, which can be attached to any ARTIFACT.

7.6.3 Guidelines for Naming Concepts

Deciding an appropriate name for a new concept is a difficult and time consuming task. Here are some guidelines to help the acquirer find good names for concepts:

1. Use an English word whose word sense maps to the concept when possible.
2. Use only alphabetic characters and the hyphen sign; avoid using other characters such as the underscore or other control characters.
3. If you must include arguments (such as default THEMEs, for instance) in EVENT names, use the argument name after the EVENT name (and a hyphen). Do not put the argument first and then the EVENT name. For example, if there is a need to distinguish between preserving FOODs and preserving antique PAINTINGs, name the concepts ‘‘preserve-food’’ and ‘‘preserve-painting’’ (as opposed to ‘‘food-preserve’’ and ‘‘painting-preserve’’).
4. Do not use plurals in concept names. For example, call it ‘‘preserve-painting,’’ not ‘‘preserve-paintings.’’
5. When there is a large discrepancy in frequencies between different word senses of a word, name the most frequent one with just the word and add hyphenated extensions to others. For example, the concept BANK will stand for the ‘‘money holding place’’ sense of ‘‘bank’’; RIVER-BANK and ‘‘device-bank’’ will stand for a ‘‘river bank’’ and a ‘‘bank of generators or disk drives,’’ respectively.

6. Keep in mind that no two frames can have the same name in the ontology. Many PROPERTYs and OBJECTs tend to suggest the same name. We must use different names for PROPERTYs.[29] For example, if ‘‘employee’’ is both an OBJECT and a PROPERTY, name the OBJECT ‘‘employee’’ and the PROPERTY EMPLOYED-BY (and its Inverse EMPLOYER-OF).

7. If necessary, append a parent’s name (or a part of a parent’s name) to a word to get a name for a child concept. Else, append a typical child’s name to a word to get a name for a parent. Do not append -N or -V (for ‘‘noun,’’ ‘‘verb’’), etc. to concept names since they are language specific items.

8. Whenever possible, use ‘‘scientific’’ rather than lay terms.

9. Try to be consistent in the names of ontological concepts while going up or down a subtree. For example, EVENT has Subclasses MENTAL-EVENT, PHYSICAL-EVENT, and SOCIAL-EVENT.

10. Whenever possible, try to include an indication of some distinguishing characteristic of the concept in its name. It is especially useful to include a characteristic that distinguishes the concept from its immediate siblings. For example, VOLUNTARY-VISUAL-EVENT and INVOLUNTARY-VISUAL-EVENT indicate EVENTs that involve ‘‘vision,’’ with ‘‘voluntary’’ or ‘‘involuntary’’ participation (corresponding to the English verbs ‘‘look’’ and ‘‘see’’).

11. English words must be used consistently in only one sense throughout the ontology. For example, we should not have DEPARTMENT-STORE as well as ‘‘store-medicine’’ (the former an OBJECT and the latter an EVENT). Perhaps we should name them DEPARTMENT-STORE and ‘‘preserve-medicine’’ (but then we must rule out ‘‘peach-preserve’’ for a jam made of peaches).[30]

12. Consistency across concepts is more important to us than conformance with a dictionary. Since there is no single word in English for "forprofit" we have no choice but to hyphenate both ‘‘for-profit’’ and ‘‘non-profit.’’

13. Do not use names longer than four words. If the name gets longer as we go down a particular subtree in the hierarchy, drop one of the most obvious words in a longer name to shorten it.

14. Avoid compound nouns in concept names unless the relationships between the meanings of the nouns are unambiguous and obvious. For example, do not use ‘‘time-unit’’; use ‘‘unit-of-time’’ instead.

15. Violate naming order if the concept refers to a commonly used phrase or is misleading. For example, use DINING-TABLE, not ‘‘table-dining.’’

8 The Development of the Mikrokosmos Ontology

The μK ontology has grown rapidly over the last year. We started with an older ontology developed in earlier KBMT projects at Carnegie Mellon University (Carlson and Nirenburg, 1990). This had about 2000 concepts, a significant number of which were either irrelevant to our domain or were not really concepts (such as for example, some instances that were encoded as concepts). We started an initial acquisition phase and added about 700 concepts. In the next stage, we did a thorough clean up (‘‘Spring cleaning’’) of the entire ontology during which time we deleted over 500 concepts. After that we doubled the size

[29] Our acquisition tools help us by complaining if a concept by the same name already exists.

[30] One of our quality control programs finds common substrings across concept names and thereby helps us find uses of the same word in multiple senses.

of the ontology to about 4300 concepts. At the present time, the ontology covers (almost) all concepts needed to represent the meanings of Spanish words collected from a 400-text corpus of newswire articles on company mergers and acquisitions. As such, we have temporarily stopped further concept acquisition and are focusing our attention on purifying the ontology to further improve its content and quality. Figure 26 shows the rate of growth of the μK ontology.

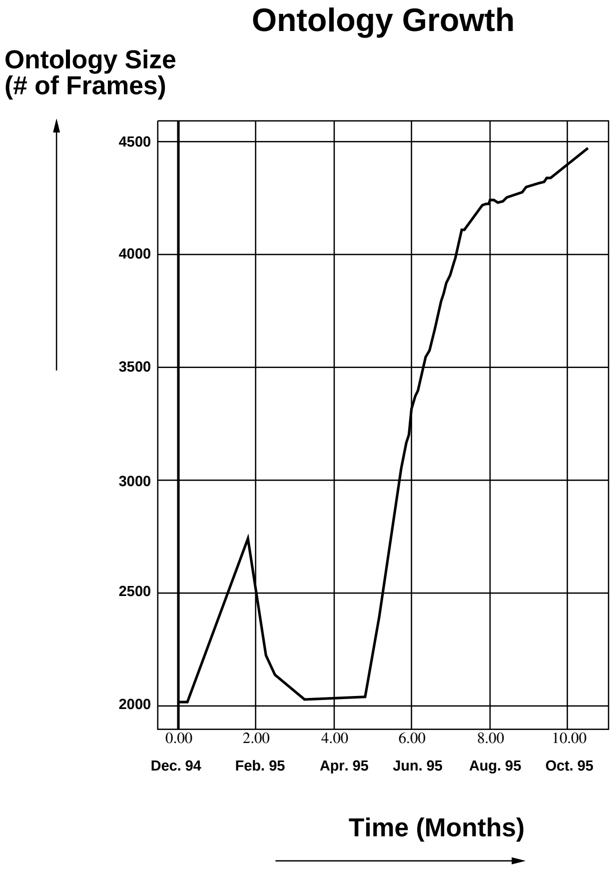


Figure 26: Rate of Growth of the μK Ontology.

8.1 Quality Improvement

The ontology we started with had 300 concepts that had no definition strings, 500 undefined symbols (apart from valid symbols intended to be literals), and many concepts that were of questionable ontological status.[31] Moreover, the majority of concepts had inappropriate slots inherited from parents as a result of placing slots at inappropriate levels in the hierarchies. For example, a ''FISH-DISH'' had a ''LUMINANCE'' slot; it also had an ''AGE'' slot with ''YEAR'' as a ''MEASURING-UNIT'' for ''AGE''! There were also several thousand cases of missing inverse links or inappropriate facets or fillers in slots.

[31]The ontology in question (Carlson and Nirenburg, 1990) was much smaller than the current μK ontology. It was constructed for a smaller MT project using tools that were not as sophisticated as our current set of tools. More than all, this ontology was modified more than once by different people to adopt to new domains such as computer repair and restaurants. As such, it is not surprising that it had to be cleaned up before thousands of new concepts from our domain could be added to it.

We developed a range of interactive programs for detecting some of the above problems and for fixing them with minimal manual effort. These programs essentially operationalize and implement the axioms that define the structure and semantics of the ontology (Appendix B). Using the programs, we were able to completely eliminate problems such as missing definitions and frames that were ‘‘orphans’’ in the ontology. However, the maintenance and improvement of the ontology’s quality is an ongoing task. We apply the programs periodically to maintain the quality. For example, we regularly check for "bad fanouts" of 10 children or more and reorganize the trees around them as needed. Also, we generate periodic listings of discrepancies between related concepts and missing elements within frames. Some of our best feedback on quality management and improvement, however, comes from our customers who browse the ontology; it is in this way that we often discover redundancies or inconsistencies.

8.1.1 Ways of Controlling the Quality of the Ontology

In order to improve the quality of the ontology, we have done the following:

- Developed a set of guidelines (some of which are shown above) for deciding whether something is a concept, where to place it, what to name it, and so on.
- Deleted nearly 500 ‘‘non-concepts’’ from the ontology. We erased many duplicates, merged many similar ones, and deleted many irrelevant ones.
- We have also been reorganizing the hierarchies to emphasize depth and eliminate pockets where some concepts had over 25 children.

Given the size of the ontology and the slowness of using the acquisition tools to navigate to every corner of it, the best method to find problems and correct things in the ontology seems to be to build separate programs that work on the databases off-line and provide interactive help in making corrections. We have built the following programs to maintain the quality of the ontology:

1. to detect missing definition strings and interactively add definitions to all concepts in the ontology.
2. to detect and edit all undefined symbols including misspelled concept names. We have a known list of valid literals in the ontology. All undefined symbols are either deleted or replaced by appropriate concept names.
3. to detect ‘‘orphans’’ (concepts with no parents) and interactively attach them to the appropriate parent or delete them from the ontology.
4. to find missing inverse links between pairs of concepts (taking inheritance and some of the structural complexities (Sec. 2.5) into account).
5. to find inappropriate use of Sem and Value facets and move fillers to appropriate facets.
6. to find inappropriate slots in concepts, such as PROPERTYs in other PROPERTYs.
7. to find RELATIONs with missing Inverse RELATIONs.

8. to report all words that are common across concept names. That is, if any two concepts have the same word (or substring) as part of their names, this program finds and reports them. This is of great potential use in enforcing certain naming guidelines (shown above).
9. to check both a lexicon and the ontology and report any discrepancies in lexical mappings to concepts.
10. to find pockets with bad ‘‘fan out’’s where there are more than 10 children.
11. to present concepts one after another in a top-down order in a graphical interface and allow the ontology developer to semi-automatically make quick notes about inaccuracies in various slots and fillers. This program (called ‘‘Onto Comment’’) also allows the user to name a concept and view it. It is very useful in checking the quality of the ontology, especially with regard to the appropriateness of inherited slots.

In addition, a number of security and computing issues come up in administering and maintaining large ontological databases. A few interesting ones along with comments on our solutions to the problems are listed in the accompanying report by Mahesh and Wilson (forthcoming).

Acknowledgements

I would like to thank all the present and former members of the Mikrokosmos team. This is as much their work as mine. I would like to thank Lori Wilson for doing most of the acquisition work and for handling so well the bulk of lexicographer requests. She has also provided many useful comments on this report. I am also highly grateful to Sergei Nirenburg, Steve Beale, Evelyne Viegas, Victor Raskin, Boyan Onyshkevych, and Jonathan K Davis. Many ideas in this report are theirs and without their input and the many discussions with them it would not have been possible for me to write such a detailed report. I also want to thank Lynn Carlson for building the previous ontology and letting us borrow it. I have merely tried to continue the work that all of these people have been doing for years. Research presented in this report was supported in part by Contract MDA904-92-C-5189 from the U.S. Department of Defense. However, the opinions expressed in this report are those of the author and do not necessarily reflect the opinions of CRL, NMSU, or the Department of Defense.

References

Bateman, J. A. (1993). Ontology construction and natural language. In Proc. International Workshop on Formal Ontology. Padua, Italy, pp. 83-93.

Bateman, J. A., Kasper, R. T., Moore, J. D., and Whitney, R. A. (1990). A general organization of knowledge for NLP: the PENMAN upper model. Technical Report, USC/Information Sciences Institute, 1990.

Beale, S., Nirenburg, S., and Mahesh, K. (1995). Semantic Analysis in the Mikrokosmos Machine Translation Project. In the Proceedings of the Second Symposium on Natural Language Processing (SNLP-95), August 2-4, 1995, Bangkok, Thailand.

Bouaud, J., Bachimont, B., Charlet, J., and Zweigenbaum, P. (1995). Methodological Principles for Structuring an ''Ontology.'' In Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, August 1995.

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

Casati, R. and Varzi, A. (1993). An Ontology for Superficial Entities I: Holes. In Proc. International Workshop on Formal Ontology. Padua, Italy, pp. 127-148.

de Kleer, J. and Brown, J. (1984). A Qualitative Physics Based on Confluences. AI, 24:7-83.

Dowell, M., Stephens, L., and Bonnell, R. (1995). Using a domain knowledge ontology as a semantic gateway among databases. In Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, August 1995.

Farwell, D., Guthrie, L., and Wilks, Y. (1993). Automatically creating lexical entries for ULTRA, a multilingual MT system, *Machine Translation,* vol. 8:3, pp. 127-146.

Fillion, F., Menzel, C., Mayer, R., and Blinn, T. (1995). An integrated, ontology-based environment for knowledge sharing. In Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, August 1995.

Genesereth, M. and Fikes, R. (1992). Knowledge Interchange Format Version 3.0 Reference Manual. Computer Science Department, Stanford University, Stanford, CA.

Goel, A. (1992). Representation of Design Functions in Experienced- Based Design. In Brown, D., Waldron, M., and Yoshikawa, H., (Eds.), Intelligent Computer Aided Design. pp. 283-308. North-Holland, Amsterdam, Netherlands.

Gruninger, M. (1995). Methodology for the design and evaluation of ontologies. In Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, August 1995.

Guarino, N. (1995). Ontology development and ontology content: some basic distinctions. In Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, August 1995.

Hayes, P. J. (1979). Naive physics manifesto. *Expert Systems in the Microelectronic Age.* Edinburgh: Edinburgh University Press.

Hayes, P. J. (1985). Naive Physics I: Ontology for liquids. In *Formal theories of the common sense world,* ed. J. Hobbs and R. C. Moore, pp. 71-107. Norwood, NJ: Ablex.

IJCAI Ontology Workshop. (1995). Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence, Montreal, Canada, August 1995.

Knight, K. and Luk, S. K. (1994). Building a Large-Scale Knowledge Base for Machine Translation. In *Proc. Twelfth National Conf. on Artificial Intelligence,* (AAAI-94).

Lenat, D. B. and Guha, R. V. (1990). *Building Large Knowledge-Based Systems.* Reading, MA: Addison-Wesley.

Mahesh, K. and Nirenburg, S. (1995a). A situated ontology for practical NLP. In Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, August 1995.

Mahesh, K. and Nirenburg, S. (1995b). Semantic Classification for Practical Natural Language Processing. To appear in *Proc. Sixth ASIS SIG/CR Classification Research Workshop: An Interdisciplinary Meeting.* October 8, 1995, Chicago IL.

Mahesh, K. and Nirenburg, S. (1996). Meaning Representation for Knowledge Sharing in Practical Machine Translation. To appear in Proc. FLAIRS-96 special track on Information Interchange, Florida AI Research Symposium, Key West, FL, May 19, 1996.

Mahesh, K., Nirenburg, S., and Beale, S. (Submitted). Exploiting Dynamic Contexts for Word Sense Disambiguation. Submitted to the Eighteenth Annual Conference of the Cognitive Science Society, July 12-15, 1996, San Diego, CA.

Mahesh, K. and Wilson, L. (forthcoming). Ontology Acquisition: Guidelines and Technology. Technical Report, Computing Research Laboratory, New Mexico State University.

Mars, N. (1993). An ontology of measurement units. In Proc. International Workshop on Formal Ontology. Padua, Italy, pp. 297-303.

Miller, G. (1990). WordNet: An on-line lexical database. *International Journal of Lexicography* 3(4) (Special Issue).

Nirenburg, S. (1989). Knowledge-Based Machine Translation. *Machine Translation,* Vol. 4, No. 1, March 1989, pp. 5-24.

Nirenburg, S. and Levin, L. (1992). Syntax-driven and ontology-driven lexical semantics. In *Lexical semantics and knowledge representation,* Pustejovsky, J. and Bergler, S. Eds. Heidelberg: Springer Verlag.

Nirenburg, S., Raskin, V. and Onyshkevych, B. (1995). Apologiae Ontologiae. Proceedings of the Conference on Theoretical and Methodological Issues in Machine Translation. Leuven, Belgium, July.

Onyshkevych, B. and Nirenburg, S. (1994). The lexicon in the scheme of KBMT things. Technical Report MCCS-94-277, Computing Research Laboratory, New Mexico State University.

Onyshkevych, B. (1995). A generalized lexical-semantics-driven approach to semantic analysis. Dissertation proposal. Program in Computational Linguistics, Carnegie Mellon University.

Pangloss. (1994). The PANGLOSS Mark III Machine Translation System. A Joint Technical Report by NMSU CRL, USC ISI and CMU CMT, Jan. 1994.

Rich, E. and Knight, K. (1991). Artificial Intelligence. Second edition. McGraw-Hill, Inc.

Russell, S. J. and Norvig, P. (1995). Artificial Intelligence: A Modern Approach, Prentice Hall.

Skuce, D. (1995). Viewing ontologies as vocabulary: merging and documenting the logical and linguistic views. In Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, August 1995.

Smith, B. (1993). Ontology and the logistic analysis of reality. In Proc. International Workshop on Formal Ontology. Padua, Italy, pp. 51-68.

Standard Industrial Classification Manual. (1987). Executive Office of the President, Office of Management and Budget, US Government. National Technical Information Service.

Van Baalen, J. and Looby, J. (1995). Translating between human genome databases. In Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, August 1995.

Zarri, G. (1995). An overview of the structural principles underpinning the construction of ''Ontologies'' in NKRL. In Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, August 1995.

Appendix A: BNF for the Mikrokosmos Ontology

Notation: { } are used for grouping; [ ] means optional (i.e., 0 or 1); + means 1 or more; and * means 0 or more.

```
<ontology> :=
  <concept>+

<onomasticon> :=
  <instance>+

<concept> :=
  <root> | <object-or-event> | <property>

<root> :=
  ALL
  <def-slot>
  <time-stamp-slot>
  <subclasses-slot>

;; ALL is the name of the root concept

<object-or-event> :=
  <concept-name>
  <def-slot>
  <time-stamp-slot>
  <isa-slot>
  [<subclasses-slot>]
  [<instances-slot>]
  <other-slot>*

<instance> :=
  <instance-name>
  <def-slot>
  <time-stamp-slot>
  <instance-of-slot>
  <instance-other-slot>*

<property> :=
  <relation> | <attribute>

<relation> :=
  <relation-name>
  <def-slot>
  <time-stamp-slot>
  <isa-slot>
```

```
  [<subclasses-slot>]
  <domain-slot>
  <rel-range-slot>
  <inverse-slot>

<attribute> :=
  <attribute-name>
  <def-slot>
  <time-stamp-slot>
  <isa-slot>
  [<subclasses-slot>]
  <domain-slot>
  [<attr-range-slot>]

<concept-name> := <name-string>

<instance-name> := <name-string>

<relation-name> := <name-string>

<attribute-name> := <name-string>

<name-string> :=
  <alpha> {<alpha> | <digit>}* {- {<alpha> | <digit>}+ }*

;; Note: The same <name-string> cannot be more than one of a
;; <concept-name>, an <instance-name>, a <relation-name>, and an
;; <attribute-name>.

<def-slot> :=
  DEFINITION Value <an English definition string>

<time-stamp-slot> :=
  TIME-STAMP Value <time-date-and-username>+

<isa-slot> :=
  IS-A Value { ALL | <concept-name>+ | <relation-name>+ | <attribute-name>+ }

<subclasses-slot> :=
  SUBCLASSES Value { <concept-name>+ | <relation-name>+ | <attribute-name>+ }

<instances-slot> :=
  INSTANCES Value <instance-name>+

<instance-of-slot> :=
  INSTANCE-OF Value <concept-name>+

<other-slot> :=
```

```
  <relation-slot> | <attribute-slot>

<relation-slot> :=
  <relation-name> <facet> <concept-name>+

<attribute-slot> :=
  <attribute-name> <facet> { <number> | <literal> }+

<facet> :=
  Sem | Default | Measuring-Unit

<instance-other-slot> :=
  <instance-relation-slot> | <instance-attribute-slot>

<instance-relation-slot> :=
  <relation-name> Value <instance-name>+

<instance-attribute-slot> :=
  <attribute-name> Value { <number> | <literal> }+

<domain-slot> :=
  DOMAIN Sem <concept-name>+

<rel-range-slot> :=
  RANGE Sem <concept-name>+

<attr-range-slot> :=
  RANGE Sem { <number> | <literal> }*

<inverse-slot> :=
  INVERSE Value <relation-name>

<alpha> :=
  A | B | C | ... | Y | Z

;; <number> and <literal> have not been expanded here but have obvious
;; definitions: a number is any string of digits with a possible
;; decimal point and a possible +/- sign; a literal is any alphanumeric
;; string starting with an <alpha>.
```

Appendix B: Axiomatic Specification of the Mikrokosmos Ontology

Introduction

Shown below is an axiomatic specification of the Mikrokosmos ontology and onomasticon in First-Order Predicate Calculus. The set of concepts in the database constitutes the ontology and the set of instances the onomasticon. Predicates used in this specification are completely disjoint from the set of symbols used in the ontological database (except, perhaps, for the equality predicate '=') and are therefore called ''meta-ontological predicates.''

There is a single, fundamental, meta-ontological predicate, namely, **slot.** Using this and a set of constants from the ontology, 36 axioms underlying the ontology have been specified below. A few basic predicates---**concept, instance, frame,** and **ancestor**---are defined using the meta-ontological predicate **slot.** The 36 axioms together define what constitutes a correct and consistent representation in the ontology and what does not. These axioms define the core ontology as it exists today, excluding such possible enhancements as complex concepts, properties having other properties, reification within the ontology, and so on.[32] I believe that this core set of axioms provides a precise framework for discussing the implications of introducing additional features and complexities in ontological representations.

Notation

The axioms below use the following symbols:

Variables: T, U, V, W, X, Y, and Z.

Meta-ontological predicates: frame, concept, instance, slot, and ancestor. Frame, concept, and instance are one-place predicates; ancestor is a two place predicate, indicating whether the second argument is an ancestor of the first. Slot is a 4-place predicate, its argument being the concept, the slot, the facet, and the filler.

Other predicates: =, subset-of, ϵ, string, literal, and scalar. The predicate ϵ is to be read as ''belongs to'' and indicates membership in a set. String, literal, and scalar are one-place predicates indicating whether a constant is a string, a scalar (i.e., a number or a range of numbers), or a literal symbol. Subset-of is used in a generic sense and includes the relationship between a scalar range and its subranges.

Logical symbols: $\forall$, $\exists$, $\neg$, $\wedge$, $\vee$, $\Longrightarrow$, $\Rightarrow$, and $\Longleftrightarrow$.

The special logical symbol $\vee_x$ is used for an *exclusive or.* That is,

$$A \vee_x B \quad \Longleftrightarrow \quad ((A \vee B) \wedge \neg(A \wedge B))$$

Constants from the ontology: ALL, OBJECT, EVENT, PROPERTY, RELATION, ATTRIBUTE, LITERAL-ATTRIBUTE, SCALAR-ATTRIBUTE, IS-A, INSTANCE-OF, SUBCLASSES, INSTANCES, DEFINITION, TIME-STAMP, DOMAIN, RANGE,

[32] I think the axioms do allow reification of properties for producing TMRs.

INVERSE, *NOTHING*, VALUE, SEM, and DEFAULT.

The Axioms

1. A frame is a concept or an instance.

$$frame(X) \iff concept(X) \vee_x instance(X)$$

Notes: Each concept or instance is represented in a single frame. Therefore, there are no complex concepts. If we want to allow complex concepts, a number of the following axioms will have to be revised and augmented (apart from significant changes to tools and the semantic analyzer).

2. Every concept except ALL must have another concept that is its IS-A.

$$concept(X) \iff (X = ALL) \vee_x (\exists Y\, concept(Y) \wedge slot(X, IS-A, VALUE, Y))$$

Notes: There are no disjoint parts in the ontology. Each hierarchy must ultimately attach to the top-level classification of concepts. This guarantees that a search in the ontology for checking selectional constraints will always succeed. Also, a concept may have multiple parents. It is the responsibility of the ontology acquirer or the semantic analyst to make sure that there are no conflicts in inheritance between such multiple parents.

3. No concept is an INSTANCE-OF anything.

$$concept(X) \Longrightarrow \neg\exists Y\, slot(X, INSTANCE-OF, VALUE, Y)$$

Notes: We do not allow frames that are both instances and concepts. Allowing such frames may be desirable for representing such things in the world as the Great Lakes (which have five instances but are not really concepts). See also #6 below.

4. If a concept X IS-A Y, then X is in the SUBCLASSES of Y.

$$slot(X, IS-A, VALUE, Y) \iff slot(Y, SUBCLASSES, VALUE, X)$$

Notes: Taxonomic links are bi-directional.

5. Every instance must have a concept that is its INSTANCE-OF.

$$instance(X) \iff \exists Y\, concept(Y) \wedge slot(X, INSTANCE-OF, VALUE, Y)$$

Notes: It is, however, possible for an instance to have more than one concept as its INSTANCE-OF. It is the responsibility of the ontology acquirer or the semantic analyst to make sure that there are no conflicts in inheritance between such multiple parents.

6. No instance is an IS-A of anything.

$$instance(X) \Longrightarrow \neg\exists Y\, slot(X, IS-A, VALUE, Y)$$

Notes: See #3 above.

7. If an instance X is an INSTANCE-OF a concept Y, then X is in the INSTANCES of Y.

$$slot(X, INSTANCE-OF, VALUE, Y) \Longleftrightarrow slot(Y, INSTANCES, VALUE, X)$$

Notes: Links between the ontology and the onomasticon are also bidirectional.

8. Instances do not have instances or subclasses.

$$instance(X) \Longrightarrow (\neg\exists Y\, slot(Y, INSTANCE-OF, VALUE, X)) \wedge (\neg\exists Y\, slot(Y, IS-A, VALUE, X))$$

9. If Y is an ancestor of X, then X and Y are concepts and either X = Y, or X IS-A Y, or X IS-A some Z and Y is an ancestor of Z.

$$\begin{aligned} ancestor(X,Y) \quad \Longleftrightarrow \quad & concept(X) \wedge concept(Y) \wedge \\ & ((X = Y) \vee_x slot(X, IS-A, VALUE, Y) \vee_x \\ & (\exists Z slot(X, IS-A, VALUE, Z) \wedge ancestor(Z,Y))) \end{aligned}$$

Notes: Note that this is just taxonomic ancestry, not subsumption. A descendant may not share all of the properties of one of its ancestors (due to nonmonotonic inheritance; see #27 and #28 below).

10. A concept is either ALL or has one of OBJECT, EVENT, and PROPERTY as an ancestor.

$$\begin{aligned} concept(X) \quad \Longleftrightarrow \quad & (X = ALL) \vee_x ancestor(X, OBJECT) \vee_x \\ & ancestor(X, EVENT) \vee_x ancestor(X, PROPERTY) \end{aligned}$$

Notes: The topmost classification is into OBJECT, EVENT, and PROPERTY. There are no *state*s in the ontology.

11. No concept has more than one of OBJECT, EVENT, and PROPERTY as ancestors.

$$
\begin{aligned}
concept(X) &\Longrightarrow \neg(ancestor(X, OBJECT) \wedge ancestor(X, EVENT)) \\
concept(X) &\Longrightarrow \neg(ancestor(X, OBJECT) \wedge ancestor(X, PROPERTY)) \\
concept(X) &\Longrightarrow \neg(ancestor(X, EVENT) \wedge ancestor(X, PROPERTY))
\end{aligned}
$$

Notes: The topmost classification defines three non-overlapping partitions. Although there are dualities between OBJECTs and EVENTs (etc.) in the world, the same concept may not have a dual status in the ontology.

12. Every frame has a DEFINITION slot filled by a string and a TIME-STAMP slot also filled by a string.

$$
\begin{aligned}
frame(X) \Longrightarrow\ & slot(X, DEFINITION, VALUE, Y) \wedge string(Y) \wedge \\
& slot(X, TIME-STAMP, VALUE, Z) \wedge string(Z)
\end{aligned}
$$

13. If Y is a slot in an OBJECT or EVENT X with a filler Z, then either Y is a property or it is one of IS-A, SUBCLASSES, INSTANCE-OF, INSTANCES, DEFINITION, and TIME-STAMP.

$$
\begin{aligned}
slot(X, Y, W, Z) \wedge\ & (ancestor(X, OBJECT) \vee ancestor(X, EVENT)) \Longrightarrow \\
& (ancestor(Y, PROPERTY) \vee_x \\
& Y \epsilon \{IS-A, SUBCLASSES, INSTANCE-OF, INSTANCES, \\
& DEFINITION, TIME-STAMP\})
\end{aligned}
$$

$$
\begin{aligned}
Y \ \epsilon\ & \{IS-A, SUBCLASSES, INSTANCE-OF, INSTANCES, DEFINITION, \\
& TIME-STAMP, DOMAIN, RANGE, INVERSE\} \Longrightarrow \\
& \neg ancestor(Y, PROPERTY)
\end{aligned}
$$

Notes: This allows the topmost concepts among PROPERTYs such as PROPERTY, RELATION, or ATTRIBUTE themselves to be slots in other concepts. This feature must of course be used only when there is no further information to indicate a more specific (and meaningful) slot, such as in representing unknown relationships among the meanings of nouns in a compound noun.

14. Only OBJECTs and EVENTs can have PROPERTYs as slots. A PROPERTY cannot have other PROPERTYs as slots.

$$
\begin{aligned}
slot(X, Y, W, Z) \wedge ancestor(Y, PROPERTY) &\Longrightarrow ancestor(X, OBJECT) \vee_x ancestor(X, EVENT) \\
slot(X, Y, W, Z) \wedge ancestor(X, PROPERTY) &\Longrightarrow \neg ancestor(Y, PROPERTY)
\end{aligned}
$$

Notes: This does not preclude reification. However, it does not permit nesting of slots or adding other slots to reified properties within the ontology. Such a thing may be necessary and permissible in Text Meaning Representations.

15. Every PROPERTY is either a RELATION or an ATTRIBUTE. No PROPERTY is both.

$$\begin{aligned} slot(X, IS-A, VALUE, PROPERTY) &\implies (X = RELATION) \vee_x (X = ATTRIBUTE) \\ ancestor(X, RELATION) &\implies \neg ancestor(X, ATTRIBUTE) \\ ancestor(X, ATTRIBUTE) &\implies \neg ancestor(X, RELATION) \end{aligned}$$

Notes: Once again, this classification is exclusive. No slot may take concepts as fillers at one time and literal or numerical values at other times. This is a limitation in meaning representation since there are often properties that are ''n-place'' and link a concept to other concepts as well as add numerical or literal values to the link.

16. Only RELATIONs have an INVERSE slot.

$$slot(X, INVERSE, VALUE, Y) \implies ancestor(X, RELATION)$$

Notes: Inverse slots are used to keep links bidirectional in the ontology. Since attributes map concepts to numbers or literals which are not defined as concepts in the ontology, it does not make sense to have an inverse link from them back to concepts.

17. Fillers of INVERSE slots are always RELATIONs.

$$slot(X, INVERSE, VALUE, Y) \implies ancestor(Y, RELATION)$$

18. Every RELATION (other than RELATION itself) has an INVERSE slot filled by another RELATION.

$$\begin{aligned} ancestor(X, RELATION) \implies\ & (X = RELATION) \vee_x \\ & (\exists Y\, slot(X, INVERSE, VALUE, Y) \wedge ancestor(Y, RELATION)) \end{aligned}$$

Notes: It is possible for a RELATION to be its own inverse. There are many such reflexive RELATIONs in the world (and in the ontology).

19. If Y is the INVERSE of X, then X is the INVERSE of Y.

$$slot(X, INVERSE, VALUE, Y) \iff slot(Y, INVERSE, VALUE, X)$$

Notes: Inverse links between relations are always reciprocal. The inverse ''relation'' (as in mathematics) partitions all RELATIONs in the ontology into a set of pairs.

20. There is only one INVERSE for every RELATION.

$$slot(X, INVERSE, VALUE, Y) \Longrightarrow \neg\exists Z(slot(X, INVERSE, VALUE, Z) \wedge (Y \neq Z))$$

21. Every PROPERTY has a DOMAIN slot and a RANGE slot.

$$\begin{aligned} ancestor(X, PROPERTY) &\Longrightarrow \exists Y\, slot(X, DOMAIN, W, Y) \\ ancestor(X, PROPERTY) &\Longrightarrow \exists Y\, slot(X, RANGE, W, Y) \end{aligned}$$

Notes: There must be ontological constraints on the domains and ranges of PROPERTYs.

22. Fillers of DOMAIN slots must be frames.

$$slot(X, DOMAIN, W, Y) \Longrightarrow frame(Y)$$

Notes: They must in fact be OBJECTs or EVENTs.

23. Fillers of RANGE slots of RELATIONs must be a frame or the special symbol **nothing** used to block inheritance. Those of ATTRIBUTEs must not be frames. Fillers of a DEFAULT facet of a slot that is a RELATION must have an ancestor in the SEM facets of the same slot.

$$\begin{aligned} slot(X, RANGE, W, Y) \wedge ancestor(X, RELATION) &\Longrightarrow frame(Y) \vee_x (Y = *NOTHING*) \\ slot(X, RANGE, W, Y) \wedge ancestor(X, ATTRIBUTE) &\Longrightarrow \neg frame(Y) \\ slot(X, Y, DEFAULT, Z) \wedge ancestor(Y, RELATION) &\Longrightarrow \exists W(slot(X, Y, SEM, W) \wedge ancestor(Z, W)) \end{aligned}$$

Notes: The special symbol **nothing** makes inheritance nonmonotonic. A child concept that uses this symbol may not have a property that its parent has.

24. If X has a slot Y that is a PROPERTY, then X must have an ancestor W that is in the DOMAIN slot of the concept Y.

$$slot(X, Y, T, Z) \wedge ancestor(Y, PROPERTY) \Longrightarrow \exists W\, slot(Y, DOMAIN, SEM, W) \wedge ancestor(X, W)$$

Notes: A concept cannot have a PROPERTY unless it falls in the domain of the PROPERTY.

25. If X has a slot Y that is a RELATION filled by Z, then Z must have an ancestor W that is in the RANGE of the concept Y or Z must be the special symbol *NOTHING* used to block inheritance.

$$slot(X,Y,T,Z) \wedge ancestor(Y, RELATION) \Longrightarrow (\exists W slot(Y, RANGE, SEM, W) \wedge ancestor(Z,W)) \vee_x (Z = *NOTHING*)$$

Notes: The filler of a RELATION slot must fall within the range of the RELATION (or be the special symbol **nothing**).

26. Every slot has an inverse slot as follows. The inverse slot is not necessarily direct; it may be inherited from higher up. Thus, If X has a slot Y that is a RELATION filled by Z, then Z has a slot U filled by V where V is an ancestor of X and Y has an INVERSE W that is an ancestor of U. If it is not even inherited, then V may be present implicitly in the RANGE slot of the INVERSE of Y.

$$\begin{aligned} slot(X,Y,T,Z) \wedge {} & ancestor(Y, RELATION) \wedge (Z \neq *NOTHING*) \Longrightarrow \\ & (\exists U \exists V slot(Z,U,T,V) \wedge ancestor(X,V) \wedge \\ & \exists W (slot(Y, INVERSE, VALUE, W) \wedge \\ & (ancestor(U,W) \vee_x ancestor(W,U)))) \vee_x \\ & (\exists W \exists V slot(Y, INVERSE, VALUE, W) \wedge \\ & slot(W, RANGE, SEM, V) \wedge ancestor(X,V)) \end{aligned}$$

Notes: In most cases, U and W are in fact identical. However, we do want to allow for the possibility that Z has a more general or more specific slot U where U and W have an ancestor relationship between them. This allows more flexibility in representing inverse links. It must be noted here that inverse links are not always directly reciprocal in the μK ontology; they may be inherited or implicit in complex ways as per the above axioms.

27. Inheritance of RELATION slots: If X has a RELATION Y as a slot filled by Z and X is an ancestor of W, then W also has a slot Y that is filled by a U that has Z as one of its ancestors or is the special symbol *NOTHING* used to block inheritance.

$$\begin{aligned} slot(X,Y,SEM,Z) \wedge {} & ancestor(Y, RELATION) \wedge ancestor(W,X) \Longrightarrow \\ & \exists U (slot(W,Y,SEM,U) \wedge \\ & (ancestor(U,Z) \vee_x (U = *NOTHING*))) \end{aligned}$$

Notes: Note that only SEM facets are inherited. Note also that these axioms do not take into account any conflicts that may arise when there are multiple parents, nor do they specify any ordering of multiple parents or a particular inheritance method such as depth-first or breadth-first.

28. Inheritance of ATTRIBUTE slots: If X has an ATTRIBUTE Y as a slot filled by Z and X is a ancestor of W, then W also has a slot Y that is filled by a U that is either equal to Z or is a subset of Z or is the special symbol *NOTHING* used to block inheritance.

$$slot(X,Y,SEM,Z) \wedge ancestor(Y, ATTRIBUTE) \wedge ancestor(W,X) \Longrightarrow$$

$$\exists U(slot(W, Y, SEM, U) \wedge ((U = Z) \vee_x subset-of(U, Z) \vee_x (U = *NOTHING*)))$$

Notes: The same observations as in #27 above apply here.

29. Inheritance to instances: an instance has all the slots that its parent concept does. The fillers in the instance are more specific than those in the concept. The instance is allowed to have a SEM facet in a slot only when it inherits the slot from a concept without making any changes to the filler of the SEM facet in the inherited slot.

$$\begin{aligned} slot(X, INSTANCE-OF, VALUE, Z) \implies & (\forall Y\, slot(Z, Y, T, W) \Rightarrow \\ & (slot(X, Y, U, V) \wedge \\ & ((ancestor(V, W) \wedge (U = VALUE))EX - \vee \\ & (subset-of(V, W) \wedge (U = VALUE)) \vee_x \\ & ((V = W) \wedge (V \neq *NOTHING*) \\ & \wedge(T = SEM) \wedge (U = SEM)) \vee_x \\ & ((V = *NOTHING*) \wedge (U = VALUE))))) \end{aligned}$$

Notes: Instances always use VALUE facets. The only exception is when a SEM facet is inherited from a parent concept without a further narrowing of the constraint. Note also that this axiom does not allow an instance in the onomasticon to have a filler that is outside the constraint specified in the parent concept. This ensures that the ontology and the onomasticon are always consistent with each other. Nevertheless, it may be desirable and possible to have fillers outside the ontologically specified constraints in an instance that is part of a (lexical or text) meaning representation.

30. Every slot has at least one of VALUE, SEM, and DEFAULT facets.

$$slot(X, Y, T, Z) \implies T \epsilon \{VALUE, SEM, DEFAULT\}$$

31. No slot can have both a VALUE and a SEM facet.

$$\begin{aligned} slot(X, Y, VALUE, Z) &\implies \neg slot(X, Y, SEM, W) \\ slot(X, Y, SEM, Z) &\implies \neg slot(X, Y, VALUE, W) \end{aligned}$$

32. Some slots only have VALUE facets.

$$\begin{aligned} slot(X, Y, T, Z) \wedge\ & Y \epsilon \{DEFINITION, TIME-STAMP, IS-A, SUBCLASSES, \\ & INSTANCE-OF, INSTANCES, INVERSE\} \implies \\ & (T = VALUE) \end{aligned}$$

33. Concepts only have SEM or DEFAULT facets in their PROPERTY slots.

$$
\begin{aligned}
concept(X) \quad \wedge \quad & slot(X, Y, T, Z) \wedge ancestor(Y, PROPERTY) \Longrightarrow \\
& (T \neq VALUE)
\end{aligned}
$$

34. Every ATTRIBUTE is either a SCALAR-ATTRIBUTE or a LITERAL-ATTRIBUTE. No ATTRIBUTE is both.

$$
\begin{aligned}
slot(X, IS-A, VALUE, ATTRIBUTE) &\Longrightarrow \\
& \quad (X = SCALAR-ATTRIBUTE) \vee_x \\
& \quad (X = LITERAL-ATTRIBUTE) \\
ancestor(X, SCALAR-ATTRIBUTE) &\Longrightarrow \neg ancestor(X, LITERAL-ATTRIBUTE) \\
ancestor(X, LITERAL-ATTRIBUTE) &\Longrightarrow \neg ancestor(X, SCALAR-ATTRIBUTE)
\end{aligned}
$$

Notes: This partitioning is again a simplification. It is often desirable to have attributes that take scalar and literal values at different times. Such ambiguity is not permitted in the ontology to enable us to specify constraints on attribute fillers precisely.

35. The RANGE of a SCALAR-ATTRIBUTE can only be filled by a scalar.

$$ancestor(X, SCALAR-ATTRIBUTE) \wedge slot(X, RANGE, T, Y) \Longrightarrow scalar(Y)$$

36. The RANGE of a LITERAL-ATTRIBUTE can only be filled by a literal symbol.

$$ancestor(X, LITERAL-ATTRIBUTE) \wedge slot(X, RANGE, T, Y) \Longrightarrow literal(Y)$$