Computational Tools for Lexicographers

Mary S. Neff and Leonard Cantor

ABSTRACT

This paper describes a set of computational tools developed for displaying, manipulating, and analyzing machine-readable monolingual and bilingual dictionaries. The tools illustrate some of the possibilities of on-line reference systems for lexical data. The paper also shows a configuration of these tools in a workstation for terminologists in the creation of bilingual technical term dictionaries for a variety of applications. Finally, it shows how the workstation, as well as further tools under development, point the way toward the implementation of a general workstation for lexicographers.

1. Introduction

The development of large numbers of natural language interfaces and processing systems, most of them requiring a dictionary system or subsystem to provide information about words, has brought with it increased interest in the computational exploitation of large machine-readable dictionaries to satisfy requirements for information about large numbers of words. The exploitation of machine-readable dictionary tapes supplied by publishers has proceeded in two major directions: for dictionary subsystems and for on-line reference systems. Most natural language interfaces and processing systems, such as natural language query, text critiquing, speech synthesis and recognition, information retrieval, and machine translation, require a dictionary (sub)system with information about large numbers of words. Some of this information is only implicit in machine-readable dictionaries and must be extracted using sophisticated techniques (Atkins et. al. 1987, Klavans 1988), some of it would be arcane to the casual user, and most—if not all—of it is never seen by the user of the natural language processing system. The group of on-line referencing systems includes telephone books, poster systems, on-line documentation or help systems, dictionaries, and even almanacs and encyclopedias. On-line dictionaries for people present information for direct use at the user's workstation.

Lexicographers can benefit in a number of ways from the exploitation of machine readable dictionaries in the computational sphere. Better dictionaries will result from awareness of both the requirements of dictionaries for people and systems and the problems encountered in the process of building these dictionaries. Such problems range from formatting errors in the tapes (Michiels 1982, Boguraev 1987), to inconsistency in lexicographical methodology (Ravin et al. 1988) to lack of a lexicographical theory underlying the structure of definitions (see Atkins and Levin 1988).

Computational tools for lexicographers are being developed as a result of the recent activity in computational lexicography. For development of future diction-

aries, lexicographers will need, among other things, tools for research and exploration of existing lexicographical data, and tools for management of new lexical data under development. In this paper we describe an on-line dictionary reference system, a data base management system for bilingual technical terminology, and a lexical data base system and query language. Originally developed separately to serve independently motivated goals in lexicographical research and terminology management, they share some components and philosophy, and, taken together, identify and meet some of the needs of lexicographers.

WordSmith is an on-line dictionary referencing system built by the Lexical Systems Project at Thomas J. Watson Research Center in Yorktown Heights, New York. The experience in building and experimenting with this system has given us insights into the needs of users of dictionaries and ways that on-line reference can satisfy those needs, and has suggested further lines of lexicographical research (Byrd 1988). WordSmith itself has become a tool for that research by facilitating on-line exploration of lexical information, encouraging exploitation of the results of otherwise independent lexicological analyses, and motivating new analyses aimed at supplying the different lexical access paths which human users of dictionaries require.

Terminology File System (TFS), developed at IBM European Language Services in Copenhagen, Denmark, is a workstation for terminologists in the creation of bilingual technical terminology data bases. WordSmith has recently been incorporated into TFS to allow the terminologist full access to any of the available on-line dictionaries, including other TFS dictionaries and the dictionary under development. While WordSmith provides access to the TFS dictionary as a dictionary, TFS utilities provide access to it as a *data base*. Both types of access are fundamental to the terminologist's workstation, providing a rich and flexible environment for the creation of technical term glossaries.

TFS manages terminology dictionaries that are created by hand or have been converted from other data in data base format. The concept of converting any machine-readable dictionary in text format to a data base has been implemented in the Dictionary Entry Parser (DEP), Lexical Data Base (LDB) architecture, and Lexical Query Language (LQL), developed by Lexical Systems. With these tools, existing machine-readable dictionaries have been parsed and stored as data bases that can be queried by an analyst or lexicographer. These tools are the center of an ever-expanding computational tool kit for the creation of future dictionaries for systems and for people.

2. WordSmith

The rather straightforward process of transforming sequential tapes provided by publishers into random access files that can be queried on-line by users has been described many times in the literature (see, for example, Akkerman et al. 1985, Amsler 1987, CELEX 1987, Cowie 1987). Commercial products embodying this technology include versions of Merriam dictionaries sold by Proximity, Inc., the Collins bilingual dictionaries prepared by Linguatech, Inc., and various dictionaries in the Bookshelf product sold by Microsoft, Inc.

Most of these products present the dictionary as an on-line version of a printed dictionary. However, random access to dictionary entries by headword does not meet all the needs of human users of dictionaries, nor does it exploit the possibilities offered by the computer for alternate access paths to lexical data, for example by sound, meaning, or, in the case of bilingual dictionaries, translation. It is well known that people require access to word information by sound, meaning, and other aspects of a word (Fox et al. 1980, Kay 1983, and Miller 1985). Further, lexicographers and terminologists require access to the same information in existing sources and their own dictionaries under development by imaginably any piece of information associated with the words. The WordSmith on-line dictionary referencing system facilitates on-line exploration of lexical information, because it offers the user not only random access by headword, but by many other access paths as well.

Originally, WordSmith was designed as a tool to allow members of the Lexical Systems project to conveniently inspect the contents of dictionary entries. WordSmith allows fast and easy access to a large number of machine-readable dictionaries, including WEBSTER'S SEVENTH COLLEGIATE DICTIONARY (henceforth W7), the LONGMAN DICTIONARY OF CONTEMPORARY ENGLISH (LDOCE), two synonym dictionaries, four bilingual dictionaries, English dictionaries of technical terms and acronyms, and, most recently, a large number of bilingual technical term dictionaries created in IBM's translation centers. WordSmith has gone beyond its original charter; it has a world-wide user community within IBM as an on-line dictionary referencing system and has been incorporated into TFS.

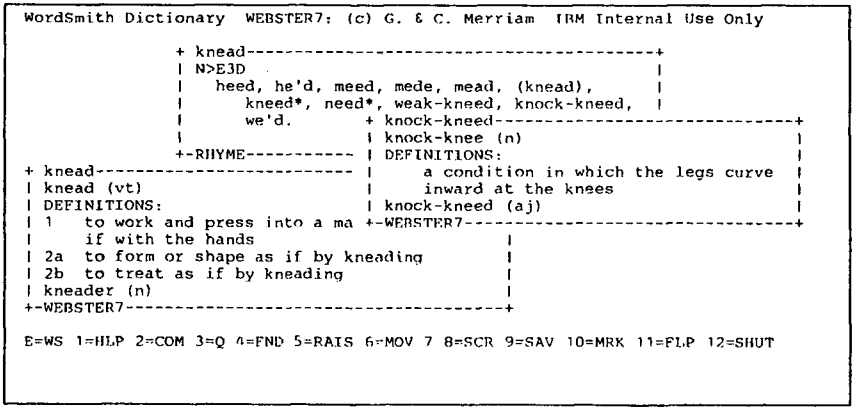


Figure 1 A sample WordSmith screen: multi-window interface

Typically, the WordSmith user requires access to a dictionary from an editor, while performing a task such as writing or translating a document. The user can access any of the dictionaries from within the editor by pointing to the requested word with the cursor and pressing an appropriately defined function key. WordSmith offers the user a choice between a multi-window interface and a multi-dimensional interface. The multi-window interface is illustrated in Fig. 1, where the user

has looked up the entries for *kneed* in W7 and in a rhyming dictionary, and the entry for *knock-kneed* in W7. The interface allows the user to open new windows by pointing with the cursor at any word on the screen (in the file under edit or in a window) and pressing the FIND key. Windows can also be moved, raised, or closed by means of function keys. Multi-word entries can be accessed by marking the first word with the cursor and the MARK function key and the last with the cursor and FIND.

Depending on the dictionary, entries may be accessed by morphological variants of their headwords or by run-on forms, as in the case of *knock-kneed.*

Fig. 2 illustrates one of the same words on the multi-dimensional interface. The dimension architecture offers alternate access paths to words in the dictionary by allowing the user to explore dimensions, or neighborhoods of words. Neighborhoods might represent criteria such as the spelling of the word (as in alphabetical order), reverse spelling of the word, or the pronunciation (as in a pronunciation-based rhyming dictionary). Since users might want to manipulate multiple dimensions simultaneously, the interface allows up to four dimensions to be displayed at once. In Fig. 2, dimension 1 is the list of alphabetical neighbors from the same dictionary as the word in the box ("knead" from W7). Dimension 2 is a list of rhyming words based on a linear rhyming dimension (described in Byrd and Chodorow 1985); dimension 3 is a reverse spelling list, showing words that end the same as the word in the box. Dimension 4, emerging from the left of the box, lists words recently looked up by the user.

As with the multi-window interface, the user can navigate the space of one or multiple dictionaries by pointing at the next word with the cursor and pressing the FIND key. Because users may want to look up words in different dictionaries in the same session, WordSmith supports the definition of a "flip ring", a circular list of dictionaries which can be accessed in sequence by using the FLIP function key, a particularly useful feature for users of bilingual dictionaries or indexed terminology dictionaries.

Dimensions offer one kind of alternate access path to dictionary data; indexes offer another. A simple dictionary index application is one that we have implemented for IBM's Dictionary of Computing, a technical term dictionary. Because technical term dictionaries contain large numbers of multi-word entries, we have found it useful to build an index over the blank-delimited words in the headwords of that dictionary. WordSmith treats the index the same as any dictionary. Fig. 3 shows the result when a user has looked up the list of words containing the word *radix* in the index, and then flipped to the technical term dictionary to look up one of them.

A significant aspect of the WordSmith architecture is its flexibility. The user can easily add new dictionaries and dimensions to the system. Definition of a "metadictionary" is also possible, where, for example, the dictionary is defined as the union of two other dictionaries or interleaving of several dictionaries with a defined precedence relationship. This feature is useful to the user who wants to define a single (logical) monolingual English dictionary, for which WordSmith displays in one box definitions for the same word from a list of dictionaries (for example, W7 and LDOCE) or the definition from the first dictionary in the list that has the word. The user who wants to define a single logical dictionary consisting of a base dictionary and one or more addendum dictionaries containing updates (a so-called "cascade" of files) can do that as well.

Figure 2 A sample WordSmith screen: multi-dimensional interface.

There is a defined interface which the dictionary access programs must meet; but within the programs, the dictionary designers may store and access their data in any way that they desire. The WordSmith toolkit contains a standard entry formatting program and storage and access method.

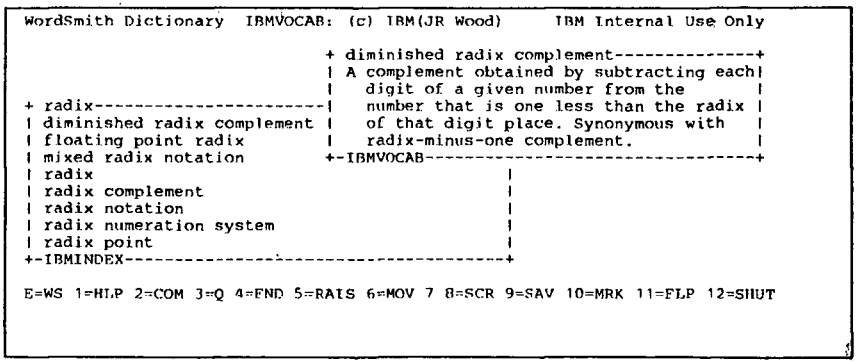


Figure 3 Indexed access to the IBM Dictionary of Computing.

The Dictionary Access Method, DAM (Byrd et al. 1986b) is a file format and access method used for the commercial dictionaries, as well as those used by TFS. It offers facilities needed by dictionary data, such as true alphabetical order in a variety of languages, random and forward and backward sequential access, and support for indexing and addenda.

3. Terminology File System

The problems of managing, standardizing, updating, and disseminating bilingual technical terminology in a corporation that has to translate more than 250,000 pages of technical documentation per year into 30 languages at 22 translation centers worldwide led to the development of Terminology File System (TFS), a data base system for entering, viewing, and analyzing technical terminology.

```
TFSSHORT ---      TFS -- Quick Update of term - - ---          13:48
Command ===>
English
Term :
Usage  =>
Cat(s) ->                              Feature     ->
Comnt. =>
Source =>


Danish
Term   =>
Subj. cds.(sep. by ; ) ->
Cat.   =>                              Feature(s) ->
Prompt =>
Comnt. =>
Source =>
Eng. cat. (if > 1)      ->



PF1 = Help/Info     PF2  = new NL term  PF3  = upd/return  PF4  = WordSmith
PF5 = Get usage     PF6  = new usage    PF7  = next usage  PF8  = next NL term
PF9 = Opt. Par. CD  PF10 =              PF11 =             PF12 =cancel/return
```

Figure 4 TFS Quick Path entry panel.

TFS features a series of data entry panels for entering and updating English terminology, definitions, translations, abbreviations, aliases, part of speech, subject area, and bookkeeping or control information. Because requirements and practices vary in the many Translation Centers where TFS is installed, TFS offers in addition to the standard data entry panel a quick path panel (shown in Fig. 4), containing the most commonly used data fields. The user can choose to be prompted for other data fields as well. The TFS dictionary file, like the Lexical Data Base described in the next section, is a data base in DAM format and supports hierarchically structured entries. This hierarchical structure reflects the structure of dictionaries where an entry can have any number of sense ("usages" in TFS terminology) and each sense can have any number of target language terms. Unlike the younger LDB format,

where the hierarchy can be arbitrarily deep and may have any number of different node types (attributes and values) at any level, TFS format allows two levels of hierarchy and a fixed number of data fields. Fig. 5 shows the structure of the TFS record; for clarity, some data fields have been omitted from the figure. For each headword or term there can be any number of senses or usages. For each usage there is, among other things, control information, part of speech and morphological class, aliases, formal definition, subject area, and any number of target language terms. For each target language term there is, among other things, control information, formal definition of the term in the target language, target term morphological class and part of speech, and aliases. It is significant that, with the hierarchical structure allowing an arbitrary number of like-type usage (sense) and translation nodes, TFS can support entry and maintenance of lexical data in a way that a conventional relational data base containing normalized relations (for example, any of the data base systems now available on personal computers) cannot. The TFS user interface allows the user to cycle through these multiple instances on each hierarchical level with a touch of a function key.

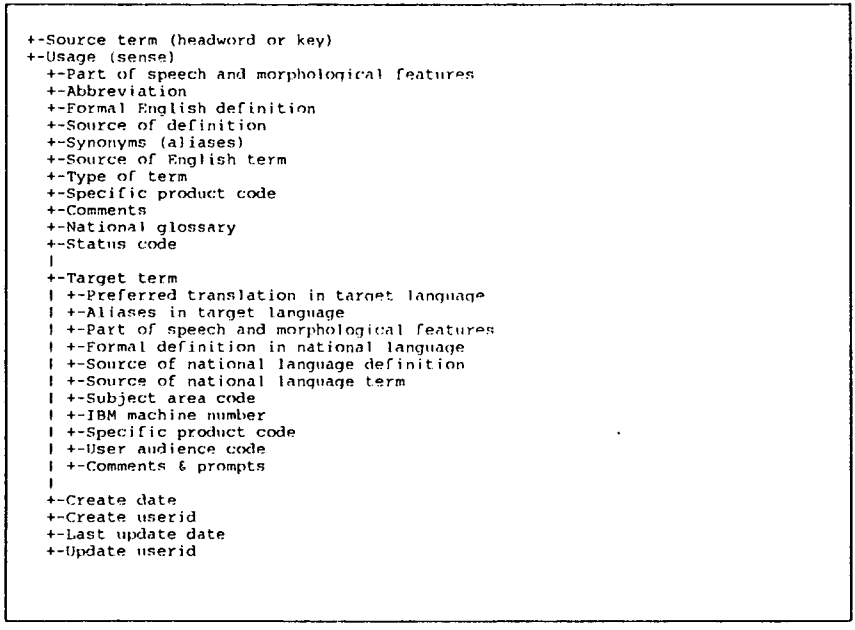


Figure 5 Data fields in TFS record.

Because a TFS data base is also a DAM file, it can be treated as a dictionary in the WordSmith system. WordSmith has been installed as an integral part of the TFS system to give terminologists free and flexible access to as many dictionaries as they wish. WordSmith is used to display the terminologist's own TFS dictionary entries in the Browse option of TFS; further, WordSmith can be entered from most points in TFS to allow the user to look up a word in any WordSmith dictionary, including available TFS dictionaries.

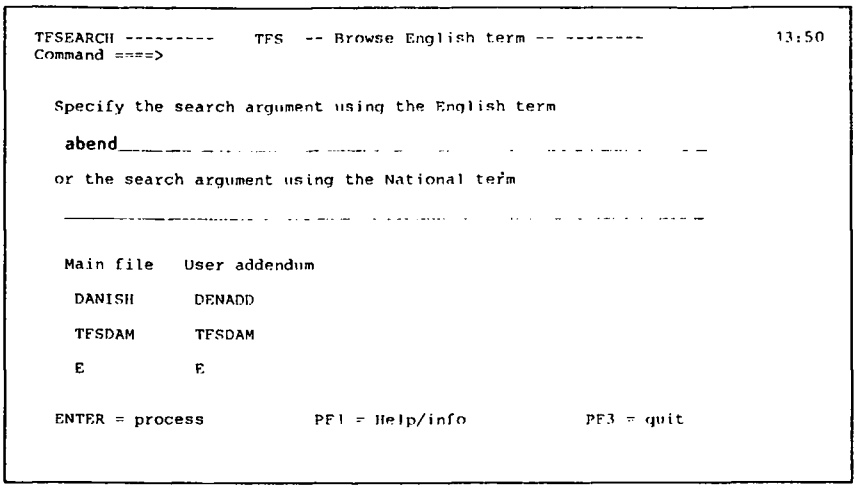


Figure 6 TFS Browse panel.

TFS makes extensive use of DAM's addendum and indexing facilities. In an environment where the terminology dictionaries are continually being updated by many terminologists, who often have to use a consensus strategy to arrive at their national language version of a newly coined English technical term, it is useful to have a base dictionary of "accepted" terms that can be accessed by the community at large, including translators, managers, and marketing people, and an "addendum" or update dictionary containing current work that is visible only to terminologists. The addendum mechanism in DAM allows both WordSmith and TFS to view a cascade of base dictionary and addendum as a single logical file, with the addendum taking precedence. TFS allows definition of up to four addendum files plus a base file. The selection and definition of the cascade is part of the user profile facility and can be changed by the user at any time.

TFS also uses DAM's built-in indexing capability. In principle, an index could be built on any field, subfield, or combination of these. In practice, TFS has two built-in indexes, an index on blank-delimited words in source terms, abbreviations, and aliases, and one on words in target language terms, target language abbreviations, and target language aliases. From the Browse panel (see Fig. 6) a user can initiate a search for a term by a word or combination of words in the source or target term (or abbreviation or alias). Specification of a search argument in Browse initiates a WordSmith lookup with the index as the first dictionary, two versions of the main dictionary (short and long display form), and the other index on the flip ring. Fig. 7 shows the results of a sequence that began in TFS: the user requested a source term index search for the term *abend,* flipped to the main dictionary to look up one of the words displayed in the index (*system abend code*), and then flipped to the Danish target term index to determine what *other* English terms are translated with the same Danish term.

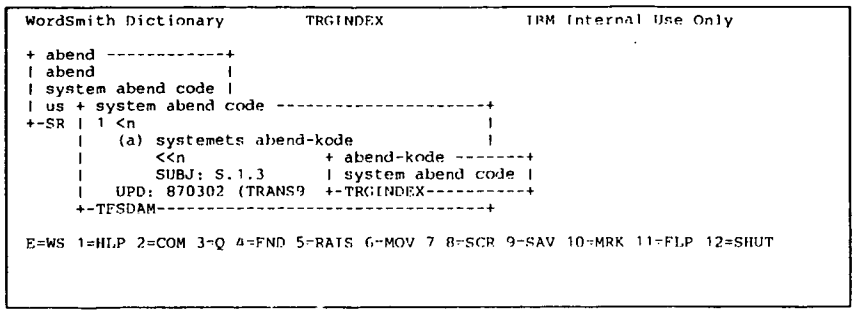


Figure 7 WordSmith use in TFS.

The index search is not sensitive to case, and in a future release, it will not be sensitive to inflectional affixation, either. Variations on specification of the search argument allow the user a larger range of fuzzy matching. A wild card character at the end of the search argument allows retrieval of all index terms matching up to the wild character. An "or" operator (";") allows for display of the results of several index retrievals in one box. For example, if the search argument is "data;base" the index retrieval lists all terms having either "data" or "base" in them. The "and" operator (blank), on the other hand, narrows the index search; for example, specification of "data base" retrieves all terms having both "data" and "base" in them.

While WordSmith provides access to the TFS dictionary as a dictionary, the TFS utilities provide access to it as a data base. In addition to the edit/update option, TFS offers the terminologist utilities which assist in terminology management. Terminologists using corpus analysis programs to discover new terminology in documents to be translated can submit the results of the corpus analysis to TFS to determine which terms are already present or missing from their TFS terminology dictionary. Extractions suitable as input for other computational programs are provided, as well as format conversion utilities. Listings of various kinds, displaying only some of the data fields like English term and national language term, are available. Users can also request listings in a markup language format, suitable for printing as text. All listings and extractions can be restricted by subject area code, product code, or alphabetical section of the dictionary. One of the capabilities of TFS—and this should be of particular interest to print lexicographers—are options for printing, with GML markup tags added, any subset of the words in a TFS dictionary. Data subsets can be defined as a subset of which words to print, depending on the value contained in a given field or combination of fields, or as a subset of which fields to print, or both. Typesetting information for a bilingual technical term dictionary published by IBM Germany (1985) was produced in exactly this way using another data base storage system. In the future, such printed dictionaries can be produced from TFS on-line dictionaries. A consequence of this approach is that terminologists do not have to concern themselves with typesetting conventions or font codes and can instead concentrate on the data, thus saving time and effort. It also means that the dictionary data base can contain any amount of information

desired or needed for a variety of applications, even if it is not all wanted in the printed book.

Many of the desiderata for a workstation for terminologist's are satisfied with TFS: easy and flexible access along several kinds of access paths to the terminologist"s own data as well as to all available on-line dictionaries and glossaries using the same user interface, logical merging of terminology files as cascaded addenda, data entry panels with field prompts, management of hierarchically structured dictionary data, and varied listing capabilities. Desired improvements lie in the area of increased generalization, allowing the user full flexibility in defining the hierarchical structure of the dictionary entry including the depth and fields desired, and in defining the format and data content of listings, WordSmith displays, and data entry panels.

4. Lexical Data Bases and Lexical Query Language

In its current form, TFS performs well as a system for entering, managing, and displaying bilingual technical terminology. Because of its limited flexibility, however, there is still a large leap from TFS to a general data base system which any lexicographer could use to maintain entries for a general dictionary. The attempt to convert typesetting tapes from several COLLINS bilingual dictionaries, whose format is essentially running text, to a data base that can be analyzed, queried, and even updated, has helped to define some of the requirements in data structures and access that might be required for such a general workstation. We will briefly describe the Dictionary Entry Parser (DEP), the Lexical Data Base (LDB) system, and the Lexical Query Language (LQL); readers interested in full descriptions of their capabilities and applications should consult Byrd et al. (1988) and Neff, et al. (1988b). These tools were originally developed to facilitate research and analysis of our on-line dictionary resources; however, it is possible to imagine extensions toward a lexicographer's workstation.

Dictionary entries are typically organized as shallow hierarchies of information with a variable number of instances of certain items at each level, e.g. multiple homographs within an entry or multiple senses within a homograph. More formally, they can be characterized as finite depth hierarchies of attribute-value pairs with a variable number of instances of specific nodes at each level. At each node there is an attribute name and a value, which can be either a simple value, or an entire subtree. Fig. 9 shows a simplified sample outline or template for dictionary entries in the COLLINS SANSONI ENGLISH-ITALIAN DICTIONARY. Recognizing that for dictionary entries, relational data bases with normalized relations do not capture the hierarchical relationships of data items to one another in an intuitively satisfying way, we have defined and built a Lexical Data Base (LDB) architecture within the DAM format, a Dictionary Entry Parser (DEP) to parse dictionary entries into hierarchically structured trees of attribute value pairs (Neff et al. (1988b)), and a Lexical Query Language (LQL) for querying, extracting from, and updating the Lexical Data Bases (LDB's) built from the parsing procedure. The Dictionary Entry Parser uses a grammar made up of parsing rules; from the grammar rules, DEP determines the Lexical Data Base design, or hierarchical structure of attributes (field names) used in the data base and parses the typesetting text into

structured trees. Fig. 8 shows a sample Lexical Data Base entry from the COLLINS GERMAN-ENGLISH DICTIONARY.

The data base template or design, like the one in Fig. 9, plays a central role in the user interface of the Lexical Query Language. The user manipulates a sample template, marking it with example elements to define a query in a process that might be called "query by structure". In this way, the user can precisely specify information to be retrieved from an LDB. Fig. 10 shows a simple LQL query that will extract for printing a German-English dictionary of musical terms with their parts of speech from the German-English Lexical Data Base (derived from the COLLINS GERMAN-ENGLISH DICTIONARY). The example shows a pruned template tree with example elements to reference the nodes. The condition box specifies that the node referenced with **-n** (domain) be equal to the string "Mus." Retrieval of entries may be made subject to a wide variety of conditions on the values of nodes in the hierarchy. It is possible, for example, to ask for only entries that have a verb sense, or that have "rope" in one of their translations. Further, users can control what attributes are shown in the answer to the query, creating a new LDB out of just those elements that are part of the answer. Users may request entire entries, only certain nodes, or just the values at particular leaf nodes. The output may be formatted for display or for printing using a two-dimensional formatting language which is part of LQL. Both text format and column layout are possible. In the example, the user has specified a simple formatting of output suitable for printing, with the addition of GML tags for changing fonts. The result of putting the LQL output through a formatter is shown at the bottom of the figure. Support for even more sophisticated output specifications should make it possible to automatically customize and print in dictionary text format any selected contents of an LDB.

```
entry
+-hdw Tau
|
+-superhom
| +-pronunc Tau_
| +-supernum 1
| +-hom
|   +-pos n
|   +-gender m
|   +-nmorph -(e)s no pl
|   +-sens
|     +-tran
|       +-word dew
|     +-collocat
|       +-source vor ~ und Tag
|       +-style poet
|       +-targ
|         +-target
|           +-word at break of day
|           +-style poet
|
+-superhom
  +-pronunc Tau_
  +-supernum 2
  +-hom
    +-pos n
    +-gender nt
    +-nmorph -(e)s, -e
    +-usage_note Seil
    +-sens
      +-tran
        +-word rope
      +-tran
        +-domain Naut
        +-word hawser
```

Figure 8 Sample LDB — COLLINS GERMAN-ENGLISH entry for *Tau*.

```
entry                   /*root of the tree*/
+-hdw                   /*English headword; key to the DAM file*/
|
+-superhom              /*superscript homograph*/
  +-hum                 /*superscript number from printed dictionary*/
  +-pron                /*encoded pronunciations*/
  |
  +-altspel             /*alternate spelling*/
  | +-note              /*e.g."U.S. English"*/
  | +-spel              /*the alternate spelling string*/
  |
  +-hom                 /*homograph*/
    +-hnum              /*homograph number*/
    +-pos               /*part-of-speech*/
    +-morph             /*irregular inflections, etc.*/
    |
    +-sens              /*sense*/
      +-snum            /*sense number*/
      |
      +-xlat            /*translation information for headword*/
      | +-note          /*usage note for the English term*/
      | +-spel          /*the Italian translation*/
      | +-gnd           /*grammatical information about the Italian*/
      |
      +-xmp             /*"example" phrases containing headword*/
        +-note          /*usage note for the English phrase*/
        +-gloss         /*the English phrase*/
        +-pos           /*part-of-speech*/
        +-expl          /*more usage information*/
        +-tran          /*Italian translation of phrase*/
        +-gnd           /*grammatical information about the Italian*/
```

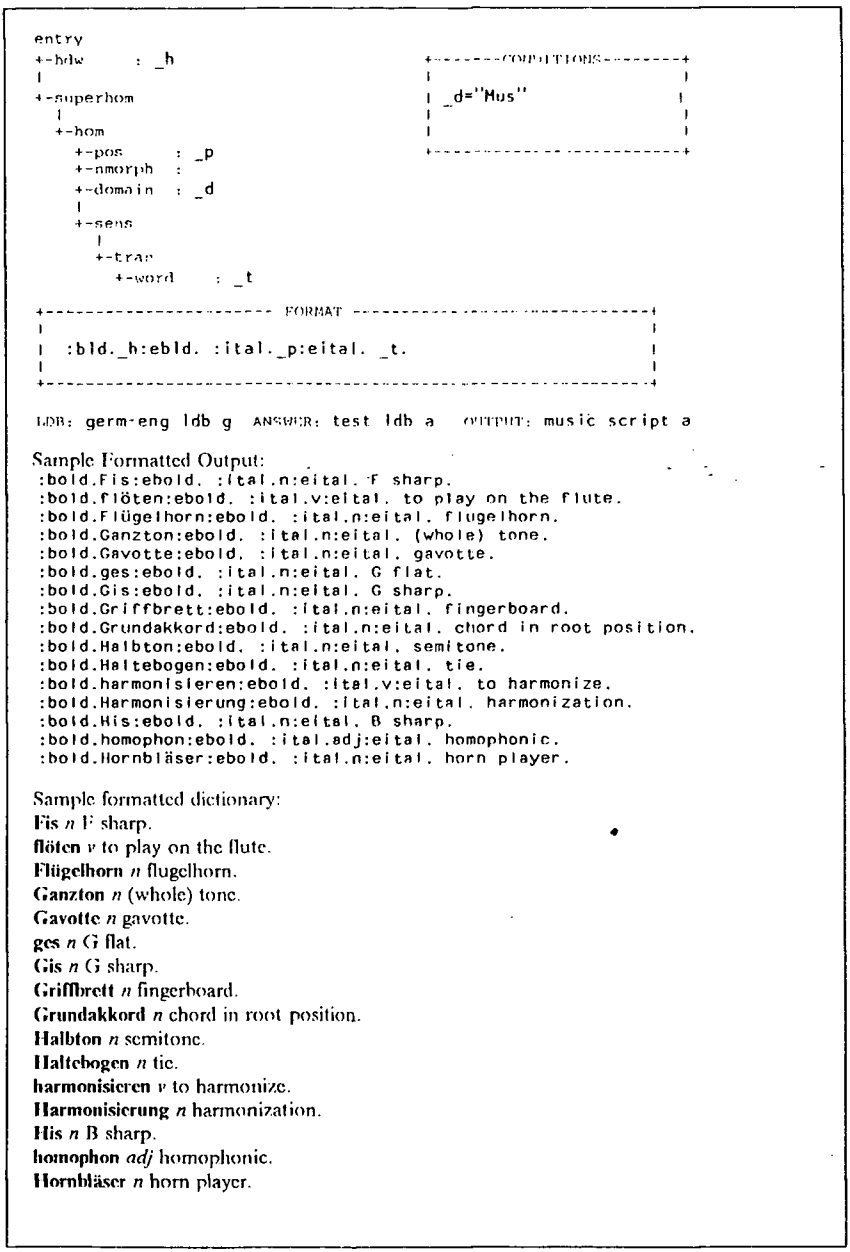
Figure 9 Simplified LDB design for the COLLINS SANSONI ENGLISH-ITALIAN DICTIONARY.

LQL will eventually allow users to update entries. Currently, users can insert, delete, or change nodes in an entry tree. We are currently exploring requirements and mechanisms for using LQL as a dictionary entry editor.

LQL is currently being used in the Lexical Systems project to extract information of various kinds from LDB's. One project is the extraction of data on verb-particle constructions from Lexical Data Bases built from several of our on-line dictionary resources for use in a computational dictionary for systems. We envision using LQL also in the on-line display of dictionaries for people. The implementation of a linear query format which does not depend on interactive user query construction will enable programs, notably WordSmith, to specify selection of data and formatting of LDB data. This will allow users to customize many different dictionaries from a single LDB.

5. Conclusion

Until recently, lexicographers writing dictionaries have had few tools available beyond their typewriters, pencils, file boxes, and piles of books. As computer hardware has become cheaper, access to it is expanding beyond the publishers' accounting offices, reaching the people who actually write the books. With this hardware come software tools in the form of word processing and desktop publishing. For lexicographers, these tools are only a beginning. We have described in this paper a set of independently motivated referencing and management tools which can be applied to tasks of interest to lexicographers, showing the promise of computational tools for lexicography and the publication of dictionaries.



Sample Formatted Output:

```
:bold.Fis:ebold. :ital.n:eital. F sharp.
:bold.flöten:ebold. :ital.v:eital. to play on the flute.
:bold.Flügelhorn:ebold. :ital.n:eital. flugelhorn.
:bold.Ganzton:ebold. :ital.n:eital. (whole) tone.
:bold.Gavotte:ebold. :ital.n:eital. gavotte.
:bold.ges:ebold. :ital.n:eital. G flat.
:bold.Gis:ebold. :ital.n:eital. G sharp.
:bold.Griffbrett:ebold. :ital.n:eital. fingerboard.
:bold.Grundakkord:ebold. :ital.n:eital. chord in root position.
:bold.Halbton:ebold. :ital.n:eital. semitone.
:bold.Haltebogen:ebold. :ital.n:eital. tie.
:bold.harmonisieren:ebold. :ital.v:eital. to harmonize.
:bold.Harmonisierung:ebold. :ital.n:eital. harmonization.
:bold.His:ebold. :ital.n:eital. B sharp.
:bold.homophon:ebold. :ital.adj:eital. homophonic.
:bold.Hornbläser:ebold. :ital.n:eital. horn player.
```

Sample formatted dictionary:

Fis *n* F sharp.
flöten *v* to play on the flute.
Flügelhorn *n* flugelhorn.
Ganzton *n* (whole) tone.
Gavotte *n* gavotte.
ges *n* G flat.
Gis *n* G sharp.
Griffbrett *n* fingerboard.
Grundakkord *n* chord in root position.
Halbton *n* semitone.
Haltebogen *n* tie.
harmonisieren *v* to harmonize.
Harmonisierung *n* harmonization.
His *n* B sharp.
homophon *adj* homophonic.
Hornbläser *n* horn player.

Figure 10 Sample LQL query with GML markup formatted output.

6. Acknowledgements

We are grateful to Roy Byrd, Judith Klavans, Martin Chodorow, Omneya Rizk, Zofia Roberts, and Nina Wacholder, members of the Lexical Systems project at IBM Research, who have all contributed to the ideas discussed here.

References

Cited Dictionaries

COLLINS GERMAN DICTIONARY: GERMAN-ENGLISH, ENGLISH-GERMAN. 1980. GLASGOW: COLLINS.

COLLINS SANSONI ITALIAN DICTIONARY: ITALIAN-ENGLISH, ENGLISH-ITALIAN. 1980. Glasgow: COLLINS

FACHAUSDRÜCKE DER INFORMATIONS-VERARBEITUNG: WÖRTERBUCH UND GLOSSAR ENGLISCH-DEUTSCH/DEUTSCH-ENGLISCH. (IBM Germany). 1985. IBM Deutschland, GmbH.

LONGMAN DICTIONARY OF CONTEMPORARY ENGLISH (LDOCE). 1987. Della Summers et al. (eds.).Harlow: Longman.

WEBSTER'S SEVENTH NEW COLLEGIATE DICTIONARY (W7). 1963. Springfield, MA: Merriam.

Other Literature

Akkerman, E., P. Masereeuw, and W. Meijs. 1985. 'Designing a Computerized Lexicon for Linguistic Purposes'. *ASCOT Report* No. 1, Amsterdam: Rodopi.

Amsler, R. A. 1987. 'How do I Turn This Book On? Preparing Text for Access as a Computational Medium' in *Proceedings of the Conference on the Uses of Large Text Data Bases.* Waterloo, Ontario: University of Waterloo, Centre for the New OED. 75—88.

Atkins, Beryl T., Judy Kegl, and Beth Levin. 1986. 'Explicit and Implicit Information in Dictionaries' in *Advances in Lexicology Second Annual Conference of the UW Centre for the New Oxford English Dictionary. Proceedings of the Conference, November 9—11, 1986.* Waterloo, Canada: University of Waterloo.

Atkins, Beryl T. and Beth Levin. 1988. To appear in *Proceedings of the Conference on Information in Text.* Waterloo, Ontario: University of Waterloo, Centre for the New OED.

Boguraev, Branimir. 1987. 'Experiences with a Machine-Readable Dictionary' in *The Uses of Large Text Databases, Proceedings of the Third Annual Conference of the UW Centre for the New Oxford English Dictionary.* Waterloo, Canada: University of Waterloo.

Byrd, Roy J. and Martin S. Chodorow. 1985. 'Using an On-Line Dictionary to Find Rhyming Words and Pronunciations for Unknown Words' in *Proceedings of the Association for Computational Linguistics.* 277—283.

Byrd, Roy J. 1986a. 'Dictionary Systems for Office Practice' in *Proceedings of the Grosseto Workshop 'On Automating the Lexicon'*, also available as *IBM Research Report* RC 11872.

Byrd, Roy J., Gustaf Neumann and Karl Seved B. Andersson. 1986b. *DAM — A Dictionary Access Method,* IBM Research Report, in preparation.

Byrd, Roy J., Nicoletta Calzolari, Martin S. Chodorow, Judith L. Klavans, Mary S. Neff, and Omneya A. Rizk. 1987. 'Tools and Methods for Computational Lexicology' in *Computational Linguistics* 13: 219—240.

Byrd, Roy J. 1988. 'Computational Lexicology for Building On-Line Dictionaries: The WordSmith Experience' to appear in *Festschrift for Bernard Quemada.* Pisa: Giardini.

CELEX *Newsletter* No. 20. Nijmegen: Centre for Lexical Information, University of Nijmegen. October, 1987.

Chodorow, Martin S., Roy J. Byrd and George E. Heidorn. 1985. 'Extracting Semantic Hierarchies from a Large On-Line Dictionary.' in *Proceedings of the Association for Computational Linguistics.* 229—304.

Cowie, J. R. 1987. 'A Direct Access Technique for Sequential Files with Variable Length Records' in *Software Practice and Experience* XVII (10): 719—728.

Fox, M. S., J. D. Bebel and A. C. Parker. 1980. 'The Automated Dictionary' in *IEEE Computer* XIII (7): 35—48.

Kay, Martin. 1983. 'The Dictionary of the Future and the Future of the Dictionary' in A. Zampolli and A. Cappelli (eds.). *The Possibilities and Limits of the Computer in Producing and Publishing Dictionaries.* Pisa: Giardini. 161—174.

Michiels, Archibal. 1982. *Exploiting a Large Dictionary Data Base.* Unpublished PhD Dissertation. Liege, Belgium: University of Liege.

Miller, George A. 1985. 'Wordnet: A.Dictionary Browser' in *Proceedings of the Conference on Information in Data.* Waterloo, Ontario: University of Waterloo Centre for the New OED. 25-28.

Neff, Mary S. and Roy J. Byrd. 1988a. 'WordSmith Users Guide.' *IBM Research Report* RC 60031.

Neff, Mary S., Roy J. Byrd, and Omneya A. Rizk. 1988b. 'Creating and Querying Hierarchical Lexical Data Bases' in *Second Conference on Applied Natural Language Processing.* Association for Computational Linguistics, 84—92.

Ravin, Yael, Martin S. Chodorow, and Howard E. Sachar. 1989. 'Tools for Lexicographers Revising an On-Line Thesaurus' in *BudaLEX '88: Papers from the EURALEX 3rd International Congress, Budapest, September 1988'*