What's needed for lexical databases? Experiences with Kirrkirr

Christopher D. Manning
Stanford University
Department of Computer Science, Gates 4A
Stanford CA 94305-9040, USA
manning@cs.stanford.edu

Kristen Parton
Stanford University
Department of Computer Science, Gates 4A
Stanford CA 94305-9040, USA
kparton@stanford.edu

Abstract

This paper discusses what is required from dictionary databases, and one approach, based on experience with *Kirrkirr*, a dictionary browser originally developed for Warlpiri, an Indigenous Australian language. The paper suggests that there is something of a disconnect between the data access needs of lexical databases and most work on semi-structured databases within the database community.

1 Introduction

This paper discusses what is desirable or necessary in the way of database technology in order to provide browsing interfaces to lexical databases. We are particularly concerned with interfaces that are usable by speakers of indigenous languages, although many of the issues extend to most lexicons. The discussion is based in part on our development of *Kirrkirr*, a dictionary browser for indigenous languages which has been developed over the last several years, and used with a large dictionary for Warlpiri, an Australian language. Before discussing the general issues in lexical databases, we would like to frame the problem by saying a little bit more about the context of *Kirrkirr*.

The aim of Kirrkirr is to let people explore the richness of the lexicon of a language – how words relate to one another, group in semantic fields and so on. In particular, we wish to make this facility available to a broad audience: indigenous language speakers, learners, school teachers and others, as well as linguists. At the time the Kirrkirr project was begun (1998), linguists accessed and maintained the large Warlpiri dictionary (Laughren and Hale, forthcoming) through a text editor, while other groups had no effective method of access.[1] In particular, all available print and online dictionaries were organized as Warlpiri-English dictionaries, and many – most vocally non-Warlpiri speaking white school teachers – felt the lack of an English-Warlpiri dictionary.

A picture of the Kirrkirr interface appears in Fig. 1. The program has met with at least modest success as a tool people actually can and do use:

> Hi Jane and Chris, Just letting you know that two literacy workers here (Rhonda and Nanginarra) use Kirrkirr quite a lot now, for checking spelling when checking written work including transcriptions. They switch between windows, Word / Kirrkirr or Pagemaker / Kirrkirr. Rhonda uses it without my prompting or involvement, Nanginarra still needs help moving between windows, but once she's going she checks everything.
>
> Today working with a teacher we came to a word she didn't know and she said "look it up on that thing" and she read through and discussed the synonyms she did know, so there's the beginnings of an impact. I can't say that in the past she wouldn't have reached for the paper dictionary – I didn't record paper dictionary usage pre-Kirrkirr, shame. Yesterday a teacher used it as a reading

[1] There is a separate small printed Warlpiri dictionary (Hale 1995), and printed dictionaries of several of the Warlpiri dialects, but these are fairly limited in their lexical coverage.

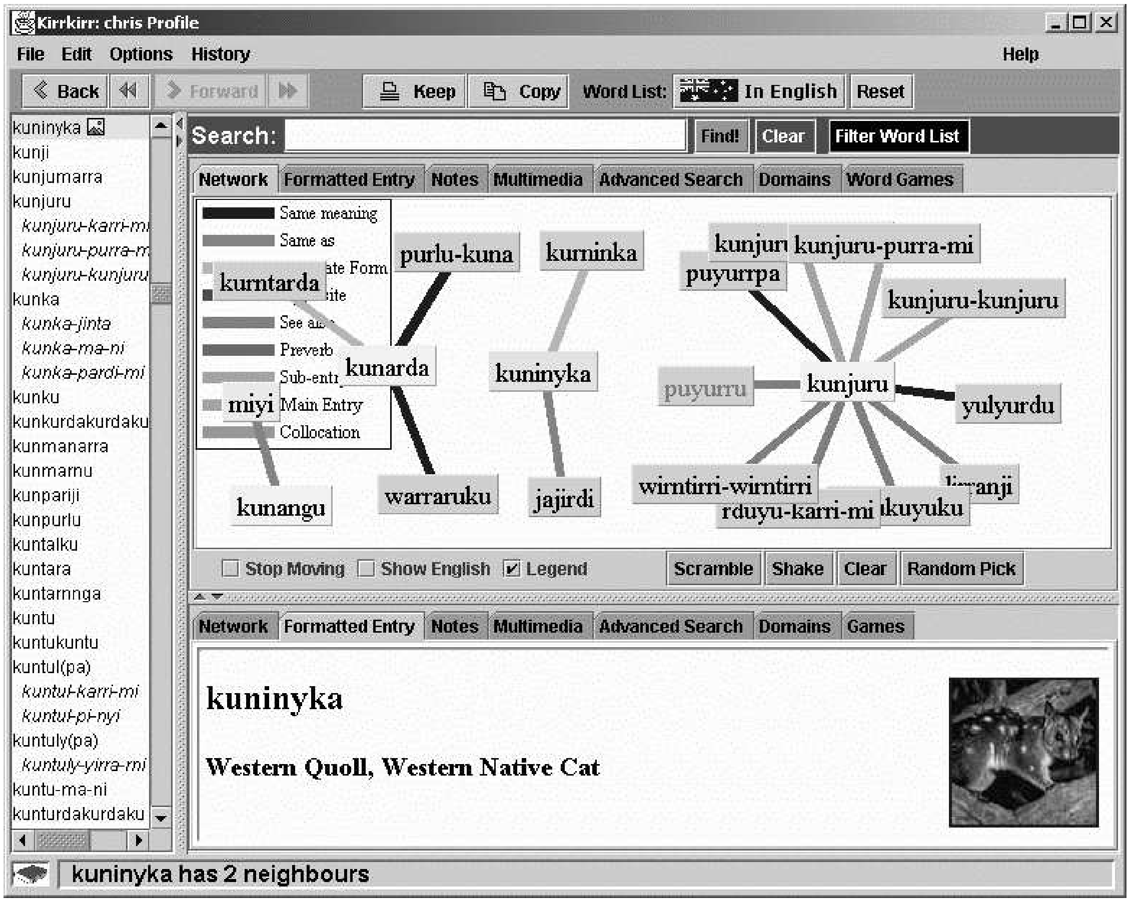

Fig. 1. One view of Kirrkirr

> skills development exercise and enjoyed reading the tree then the Warlpiri examples for about 30 mins. Ngulajuku.

This has required: a lot of careful HCI work to make the system approachable to, accessible to, and easily usable by children and novices; the traditional concerns of software engineering; considerable work in somewhat unexpected directions (such as getting the application to perform well on a 640x480 screen – many computers get set to this resolution as the easiest way to compensate for the poor levels of vision which are unfortunately common among Indigenous Australians); and finally work on having the necessary sort of lexical database to support the functionality that we seek to provide.

Here, we focus on this last area. We first discuss general issues of how dictionary databases connect to and differ from other work in semi-structructed databases, and secondly we provide some more details of the data model and data access in Kirrkirr.

2 Dictionary data access

While dictionaries have sometimes been represented in, and accessed through, regular relational databases (for example, Nathan and Austin 1992), dictionaries are best thought of as *semi-structured data.* While there is considerable systematicity to dictionary entries, there are numerous variant formats that are used to accommodate the perceived lexicographic needs of different entries, and in practice there is usually no strict schema control to stop a lexicographer from using variant or hybrid structures. To take just one example, the Warlpiri dictionary has a SRC element, for giving the source from which something is drawn, and lexicographers feel that it is completely appropriate to attach this element wherever it is needed – to an example, a synonym, a comment, even to someone's proposal as to how a word should be glossed.

In recent years, there has been much work on semi-structured data and databases for such data (inter alia, Abiteboul et al. 1999). Much of it has focused on the development of XML, although the general issue of the treatment of semi-structured data is more general, and predates XML (McHugh et al. 1997). However, the term 'semi-structured data' spans a continuum between completely structured data, which people have simply chosen to encode in XML, to moderately structured data, to quite unstructured, often textual, data. Linguistic databases, for both good and bad reasons, tend to be at this unstructured end. Unfortunately for the builders of linguistic databases, most of the research on semi-structured databases has focused on the quite structured end (McHugh et al. 1997, Florescu and Kossman 1999), with only limited work aimed at text databases (Rizzolo and Mendelzon 2001).

We believe that a crucial, insufficiently addressed observation is that in quite unstructured databases, the content of fields is also likely to be quite free form. Because of this, conventional database indices are of limited use. Dictionaries contain fields like definitions, which can only be usefully indexed by building full text indices, using standard techniques such as inverted files (Baeza-Yates and Ribeiro-Neto 1999). For other kinds of questions, such as the questions linguists often want to ask ("are there any cases of a velar between front vowels?"), pre-indexing is even less possible. Querying over such data is often much more effectively addressed by regular expression searching (perhaps because of its utility, this is not infrequently something that linguists have surprising expertise in, but we have in mind here use of regular expressions on behalf of the user, so as to make this functionality available even to naïve users). Regular expression searching allows one to easily make available possibilities such as "fuzzy spelling" to allow for frequent spelling mistakes by the user. In recent work, we have been looking at doing online morphological processing of the indigenous language, which is again well handled as a finite state transduction (Kaplan and Kay 1994).[2]

Conversely, with modern computer technology, the algorithmic search issues for dictionary databases are not particularly dire, certainly not when dealing with indigenous languages. At about 10 megabytes (or 1 million words), the Warlpiri dictionary is one of the largest indigenous language dictionaries, with encyclopedic definitions, and detailed grammatical notes. It is also larger than the databases that seem to be used most commonly for benchmarking semi-structured data (DBLP, IMDB; e.g., Rizzolo and Mendelzon 2001). With a modern (but in no way high-end) personal computer, this amount of data can be searched by regular expressions in 2–3 seconds, for a search through the entire dictionary database. In our experience, users are quite willing and expecting to wait that sort of length of time for a whole database search. Although faster performance would be available with indexing (Rizzolo and Mendelzon 2001), from a speed perspective, indexing is quite optional.[3]

Thirdly, most of the work on querying over semi-structured databases has focused on the highly structured end of the problem. It has focused on indexing the path structure of the database, and then matching and doing relational operations over such path expressions. Often this work has assumed the ability to do exact matching of paths from the root and exact matching of field contents. However, for lexical databases, not much of the querying makes interesting use of path expressions. Most of the queries are primarily aimed at textual content, delimited by XML entities, with simple intersection or alternation, rather than complex join conditions. Realistic search needs do not add excessive combinatoric complexity, and are

[2] Some databases, such as MySQL, do support regular expressions, but such flexible text search facilities are not part of standard SQL nor of any of the XML query languages of which we are aware.

[3] The Oxford English Dictionary, at around 550 Mb, does provide a reasonable case for indexing, but even there, at a cost to functionality. For example, the venerable PAT search engine for the OED (Salminen and Tompa 1994) allowed only a restricted form of wildcarding, where one had to specify a word prefix (since this is what is easily possible with an inverted file full text index). As a result, it was quite impossible to pose many queries that frequently occur ("what prefixes does the word *develop* occur with?", "how many words are there that end in *–ism*?").

usually amenable to a simple linear search of relevant entities with appropriate conjunction and disjunction of match conditions.

Thus there is something of a disconnect between what lexical databases need and the research done in the semi-structured database community (though see Barbosa et al. (2000) for work that emphasizes the dimension of structuredness and giving equal emphasis to textual XML documents). The Kirrkirr project has experimented with XML databases and query languages. In particular, we used the GMD IPSI XQL engine (GMD-IPSI 1999) in the version of Kirrkirr described in (Jansz et al. 2000). The GMD IPSI database software maintains a disk-based PDOM (persistent document object model) over which queries are made using XQL (XQL 1999), one of several proposed XML query languages. However, in practice it proved slower, and more diskspace intensive than simply using a text XML file, while only allowing a subset of the queries we wanted.

In principle, we would much prefer to be using a well-defined query language rather than doing ad hoc indexing and retrieval from files. But we have not been able to find an option that offers convincing advantages across speed, functionality, and memory footprint, so, in practice, the latter is exactly what we do at present. In part this is for parochial reasons, but many of the reasons are going to recur in lexical database projects, particularly ones aimed at indigenous languages.[4] We hope that the future will bring semi-structured databases better suited for textual XML files, even though the majority of commercial interest is in highly structured XML files (commonly actually derived from relational databases or similar sources).

[4] We might note that there are also some purely practical concerns that might recur. Firstly, for most indigenous dictionary projects, it is important that the dictionary can be given (to native speakers, linguists, etc.) at a low cost, or preferably free, and this makes it impractical to use expensive commercial solutions. Secondly, we have been somewhat constrained from even exploring newer Java object databases by the fact that we need to keep our software compatible with JDK 1.1 so that it will run on the (MacOS 9) Macintoshes which are used by the Northern Territory Education Department and the linguists and lexicographers with whom we have been working.

3 Data access in Kirrkirr

This section provides a brief description of the current lexicon structure, indexing and lexical access in Kirrkirr (see Manning et al. 2001 for more on the goals and interface of Kirrkirr). The design of Kirrkirr is general, but since we have principally used it with one Warlpiri-English dictionary, we will for simplicity refer to “Warlpiri” and “English” throughout.

3.1 Original data and XML DTD

The Warlpiri dictionary data that we have used continues to be maintained by the lexicographers in text files (the lexicographers are used to, and like, that format, despite all the problems of data consistency, validation, and so on). This dictionary data is converted to XML by a stack-based error-correcting Perl parser. While the error correction is heuristic with regard to content decisions, it guarantees that the output is both well-formed XML and valid according to the Warlpiri dictionary DTD we use, and allows us to feed corrections back to the dictionary authors. The complete current Warlpiri dictionary DTD (minus some comments) is shown as an unnumbered figure on the final page of this paper. It basically follows the dictionary structure that has evolved for the Warlpiri Dictionary (Laughren and Nash 1983), and will not be discussed in detail here. Most elements end up as mixed content, in part because the XML is seen as traditional text mark-up, which merely augments the dictionary text, and so, for example, all lists become mixed content because there is some form of punctuation between the list items. The DTD could also be made more compact by using the same entity to represent the items in all the various kinds of crossreference lists towards the end of the DTD; there is no good reason for us not having done that. The dictionary is kept as one XML file, and comprises a bit over 10 megabytes of text (one character per byte).

3.2 Indexing

Kirrkirr builds and stores on disk two (ad hoc) indices/tables over the XML file. One is an index by Warlpiri headwords to file positions. This table also holds a few additional bits of information (whether words have pictures, sounds, are subentries) – the information that is needed to be able to draw the scroll list down the left hand side of the interface, since scrolling has to be rendered quickly without XML parsing. The second index is of English glosses, with references to the corresponding Warlpiri words that can be glossed in a certain way. This is used to provide English-Warlpiri dictionary functionality, despite the fact that the underlying dictionary is only Warlpiri-English. While the program is running, these indices are kept in memory.

3.3 Data access

During operation of the program, various sorts of data needs are dealt with in different ways. Simple lookups, scroll list display, and searches over headwords or glosses can be done purely using the in-memory indices. However, most operations require more than this. For such operations as getting crossreferenced items for the network display, domains for the semantic domain browser, or pictures and sounds for the multimedia components, the program uses the headword index to jump to the right place in the file, and then invokes an XML parser (the Xerces-J parser, http://xml.apache.org/xerces-j/, using SAX) to extract the required information. It stops running at the end of a dictionary entry. For generating formatted dictionary entries, the same mechanism of processing the large XML dictionary file is used, but the content is fed together with one of a variety of XSL style files to an XSLT processor (Xalan-J, http://xml.apache.org/xalan-j/). For doing more complex searches across the dictionary, we simply run regular expression matches (using Jakarta ORO, http://jakarta.apache.org/oro/), across either the whole file or the entries that the search is restricted to (found via the headword index). Operations are similar when operating the dictionary in English-Warlpiri mode, except that another level of indirection is needed to gather Warlpiri headwords that have the required English glosses.

3.4 Genericity

How specific is this setup to our current dictionary? Kirrkirr needs to know element names that it can treat in specified ways (such as ones that represent crossreferences). And certain things need to be provided on a language or dictionary specific basis (suitable fuzzy spelling rules, and suitable XSL style files). Specifying the element names of interest is at present hardwired, but we believe these constants could easily be exported to an XML metadata file that specifies how elements of the dictionary can be mapped to Kirrkirr functionality. We intend to do this in future work.

4 Conclusion

In this paper we have briefly addressed the database needs for dictionary databases, how they are not being particularly addressed by current work in semi-structured databases, and have looked concretely at the data structuring and data access methods that are used in one particular dictionary exploration tool.

References

Abiteboul, S., Buneman, P., and Suciu, D. 2000. *Data on the Web: From Relations to Semi-structured Data and XML*. San Francisco, CA: Morgan Kaufmann.

Baeza-Yates, R. and Ribeiro-Neto, B. 1999. Modern Information Retrieval. Addison Wesley.

Barbosa, D., A. Barta, A. Mendelzon, G. Mihaila, F. Rizzolo, P. Rodriguez-Gianolli. 2001. ToX - The Toronto XML Engine, International Workshop on Information Integration on the Web, Rio de Janeiro.

Florescu, D. and D. Kossmann. 1999. Storing and Querying XML Data using an RDMBS. IEEE Data Engineering Bulletin 22: 27–34 .

GMD-IPSI 1999. http://xml.darmstadt.gmd.de/xql/.

Hale, K. L. 1995. An Elementary Warlpiri Dictionary. Revised edition. Alice Springs: IAD Press. [Revision of 1974 and 1977 versions.]

Kevin J., J. W. Sng, N. Indurkhya, and C. Manning. 2000. Using XSL And XQL For Efficient Customised Access To Dictionary Information. Proceedings of AusWeb 2000, the Sixth Australian World Wide Web Conference. pp 167–181.

Kaplan, R. and Kay, M. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20:331–379.

Laughren, M. and Hale, K. Forthcoming. Warlpiri Dictionary. Department of English, University of Queensland, ms.

Laughren, M. and Nash, D. G. 1983. Warlpiri Dictionary Project: Aims, method, organisation and problems of definition. In *Papers in Australian Linguistics No. 15: Australian Aboriginal Lexicography*. Pacific Linguistics, Canberra, pp. 109–133.

Manning, C. D., K. Jansz, and N. Indurkhya. 2001. Kirrkirr: Software for browsing and visual exploration of a structured Warlpiri dictionary. *Literary and Linguistic Computing* 16: 123–139.

McHugh, J., S. Abiteboul, R. Goldman, D. Quass, and J. Widom. 1997. Lore: A Database Management System for Semistructured Data. *SIGMOD Record*, 26:54–66.

Nathan, D. & Austin, P. 1992. Finderlists, Computer-generated, for bilingual dictionaries. *International Journal of Lexicography* 5:1, 32–65.

Rizzolo, F. and A. Mendelzon. 2001. Indexing XML Data with ToXin. Fourth International Workshop on the Web and Databases (in conjunction with ACM SIGMOD 2001). Santa Barbara, CA.

Salminen, A and F. W. Tompa. 1994. PAT Expressions: an algebra for text search. *Acta Linguistica Hungarica* 41(1–4): 277–306.

XQL. 1998. W3C-QL '98 workshop paper. http://www.w3.org/TandS/QL/QL98/pp/xql.html

```
<!DOCTYPE DICTIONARY [
<!-- The Warlpiri dictionary is a series of entries -->
<!ELEMENT DICTIONARY (ENTRY*)>
<!-- The 3 main recursive structures are ENTRY, SENSE, and PDX
     (paradigm examples - see Laughren and Nash (1983)) -->
<!ELEMENT ENTRY
 (HW,IMAGE?,SOUND?,FREQ?,POS?,POS?,DIALECTS?,REGISTERS?,
        (CRITERION|DOMAIN|DEF|GL|EXAMPLES|PDX|SENSE|
      CM|CSL|CMP|LAT|XS|REF|REFA|RUL|DERIV|REM|SRC|
                SYN|ANT|CF|XME|ALT|PVL|CME|SE)*)>
<!ELEMENT SENSE (POS|DIALECTS|REGISTERS|
              CRITERION|DOMAIN|DEF|GL|EXAMPLES|PDX|
              CM|CSL|CMP|LAT|XS|REF|REFA|RUL|
              SYN|ANT|CF|XME|ALT|PVL)+>
<!ELEMENT PDX (DIALECTS|REGISTERS|
             CRITERION|DOMAIN|DEF|GL|EXAMPLES|SENSE|
             CM|CSL|CMP|SYN|ANT|CF|XME|ALT|PVL)+>
<!-- Dictionary headword -->
<!ELEMENT HW (#PCDATA)>
  <!ATTLIST HW HNUM CDATA #IMPLIED
             TYPE (SUB) #IMPLIED>
<!-- Picture and sound files -->
<!ELEMENT IMAGE (IMGI+)>
<!ELEMENT IMGI (#PCDATA)>
  <!ATTLIST IMGI CREDITS CDATA #IMPLIED>
<!ELEMENT SOUND (SNDI+)>
<!ELEMENT SNDI (#PCDATA)>
<!-- Word frequency -->
<!ELEMENT FREQ (#PCDATA)>
<!-- Part of speech -->
<!ELEMENT POS (#PCDATA)>
<!--(Elaborated) Definition -->
<!ELEMENT DEF (#PCDATA|CT|SRC|LATIN|REM)*>
<!-- Glosses (suitable translations)-->
<!ELEMENT GL (#PCDATA|GLI)*>
<!ELEMENT GLI (#PCDATA|SRC|LATIN|CT|REM)*>
<!-- Comments -->
<!ELEMENT CM (#PCDATA|CT|LATIN|SRC|REM)*>
<!-- Derivation -->
<!ELEMENT DERIV (#PCDATA|CT)*>
<!-- References (to other works) -->
<!ELEMENT REF (#PCDATA|CT|SRC)*>
<!-- References (to the appendix) -->
<!ELEMENT REFA (#PCDATA)>
<!-- (Grammatical) Rules -->
<!ELEMENT RUL (#PCDATA)>
<!-- Remarks -->
<!ELEMENT REM (#PCDATA|SRC|CT)*>
<!-- Comparative linguistic notes -->
<!ELEMENT CMP (#PCDATA|CT|REM|SRC)*>
<!-- Extra sources -->
<!ELEMENT XS (#PCDATA|SRC)*>
<!-- Sign language crossreferences -->
<!ELEMENT CSL (#PCDATA)>
<!-- Criterion (for sense/paradigm examples) -->
<!ELEMENT CRITERION (#PCDATA|CT|REM|SRC)*>
<!-- Example blocks consist of a series of examples,
  each example containing warlpiri sentences and translations -->
<!ELEMENT EXAMPLES (EXAMPLE|CM)*>
<!ELEMENT EXAMPLE (WE, ET?)>
<!ELEMENT WE (#PCDATA|BF|SRC|CT|REM)*>
  <!ATTLIST WE TYPE (DEFN) #IMPLIED>
 <!ELEMENT ET (#PCDATA|CT|LATIN|REM|SRC)*>
<!-- Source of something -->
<!ELEMENT SRC (#PCDATA)>
<!-- Latin name for flora/fauna -->
<!ELEMENT LAT (#PCDATA|REM|SRC)*>
<!-- Bold face for emphasis -->
<!ELEMENT BF (#PCDATA)>
<!-- Semantic domains -->
<!ELEMENT DOMAIN (#PCDATA|DMI)*>
<!ELEMENT DMI (#PCDATA)>
  <!ATTLIST DMI HENTRY CDATA #IMPLIED
             HNUM CDATA #IMPLIED>
<!-- Dialects and Special Registers (baby talk, etc.)-->
<!ELEMENT DIALECTS (#PCDATA|DLI)*>
<!ELEMENT DLI (#PCDATA)>
<!ELEMENT REGISTERS (#PCDATA|RGI)*>
<!ELEMENT RGI (#PCDATA)>
<!-- Latin embedded in another field -->
<!ELEMENT LATIN (#PCDATA)>
<!-- Citation of Warlpiri in an English field -->
<!ELEMENT CT (#PCDATA|CTI)*>
<!ELEMENT CTI (#PCDATA)>
  <!ATTLIST CTI HENTRY CDATA #REQUIRED
             HNUM CDATA #IMPLIED
             SNUM CDATA #IMPLIED>
<!-- The standard crossreference types are structured identically:
     SYN=synonym, ANT=antonym, XME=crossreference to main
     entry, CF=see also, ALT=alternate form, PVL=preverb list,
     SE=subentry, CME=crossreference to main entry -->
<!ELEMENT SYN (#PCDATA|SYNI)*>
<!ELEMENT SYNI (#PCDATA|DIALECTS|REGISTERS|SRC)*>
  <!ATTLIST SYNI HENTRY CDATA #REQUIRED
             HNUM CDATA #IMPLIED
             SNUM CDATA #IMPLIED>
<!ELEMENT XME (#PCDATA|XMEI)*>
<!ELEMENT XMEI (#PCDATA|DIALECTS|REGISTERS|SRC)*>
  <!ATTLIST XMEI HENTRY CDATA #REQUIRED
             HNUM CDATA #IMPLIED
             SNUM CDATA #IMPLIED>
<!ELEMENT ANT (#PCDATA|ANTI)*>
<!ELEMENT ANTI (#PCDATA|DIALECTS|REGISTERS|SRC)*>
  <!ATTLIST ANTI HENTRY CDATA #REQUIRED
             HNUM CDATA #IMPLIED
             SNUM CDATA #IMPLIED>
<!ELEMENT CF (#PCDATA|CFI)*>
<!ELEMENT CFI (#PCDATA|DIALECTS|REGISTERS|SRC)*>
  <!ATTLIST CFI HENTRY CDATA #REQUIRED
             HNUM CDATA #IMPLIED
             SNUM CDATA #IMPLIED>
<!ELEMENT ALT (#PCDATA|ALTI)*>
<!ELEMENT ALTI (#PCDATA|DIALECTS|REGISTERS|SRC)*>
  <!ATTLIST ALTI HENTRY CDATA #REQUIRED
             HNUM CDATA #IMPLIED>
<!ELEMENT PVL (#PCDATA|PVLI)*>
<!ELEMENT PVLI (#PCDATA|DIALECTS|REGISTERS|SRC)*>
  <!ATTLIST PVLI HENTRY CDATA #REQUIRED
             HNUM CDATA #IMPLIED>
<!ELEMENT SE (#PCDATA|SEI)*>
<!ELEMENT SEI (#PCDATA)>
  <!ATTLIST SEI HENTRY CDATA #REQUIRED
             HNUM CDATA #IMPLIED>
<!ELEMENT CME (#PCDATA|CMEI)*>
<!ELEMENT CMEI (#PCDATA)>
  <!ATTLIST CMEI HENTRY CDATA #REQUIRED
             HNUM CDATA #IMPLIED>
]>
```