A Corpus-Based Grammar Tutor for Education in Language and Speech Technology

Lars Borin and **Mats Dahllöf**
Department of Linguistics, Uppsala University,
Box 527, SE-751 20 UPPSALA, SWEDEN
E-mail: Lars.Borin@ling.uu.se, Mats.Dahllof@ling.uu.se

Abstract

We describe work in progress on a corpus-based tutoring system for education in traditional and formal grammar. It is mainly intended for language and speech technology students and gives them the opportunity to learn grammar and grammatical analysis from authentic language material. The exercises offered by the system are based on pedagogically adapted versions of formalisms and tools that are likely to be of relevance to the students also later in their professional life. The system will be continuously evaluated in university-level courses, both in order to assess its effectiveness as a learning aid and to provide guidance in its further development.

1 Background

In this paper, we describe work in progress on a corpus-based grammar tutor. The inspiration comes from the authors' extensive experience of teaching traditional and formal grammar to students specialising in computational linguistics and language engineering or—to use the terminology adopted for this workshop—Language and Speech Technology (LST). This practical teaching experience, from both elementary and advanced university courses, provided the original impetus to undertake the work described here. In particular, we were motivated by the following considerations. We wished:

- to raise the generally poor level of grammar skills in our students, in an economic reality where the resources to provide individual instruction are conspicuously lacking;
- to base the grammar learning on realistic data, both as to the kind of language analysed and as to the formalism used;
- to separate for didactical purposes two conceptually, but not practically, independent components of a natural language grammatical description, viz. grammar and lexicon, without reducing either component to triviality;
- to go from the simple to the complex in terms of the expressiveness of the formalism;
- to support a hypothetico-deductive learning style "learning by inquiry" (McArthur et al. 1995) based on *intrinsic feedback* (Laurillard 1996).

1.1 LST students' grammar skills

Grammar, which used to play a leading role in teaching and learning languages, has been relegated to a fairly subordinate position in contemporary language pedagogy. Instead, language learning theorists currently favour so-called communicative approaches to language teaching. While it is true that most researchers in the field of second language learning (e.g. Ellis 1985; Lightbown and Spada 1993) recognise learning situations and learner types for which grammar may prove the most effective means of learning at least some aspects of a second or foreign language (see, e.g., Batstone 1994), even here it is not always *explicit* grammatical reasoning that the authors have in mind (see Underwood 1984). Theoretical insight in grammar is, on the whole, not regarded as very useful for language learners.

We do not wish to take issue with the standpoint that foreign or second languages in general are better learned through conversational interaction than through the study of grammar. It is however obvious that the subject (both morphology and syntax) is a matter of vital importance for LST (and linguistics) students. They consequently need a solid grasp of the fundamental grammatical and lexicological concepts and the ability to apply them to the analysis of texts.

They also need to be able to use grammar formalisms to state general grammatical principles.

From our point of view, the ideological shift in language pedagogy has brought with it the unfortunate consequence that the students who enroll on our courses generally know less grammar than used to be the case. This in turn has forced our department to restructure both the language engineering and general linguistics curricula, through the addition of courses covering elementary grammatical concepts and techniques, knowledge of which we earlier could take for granted in our students.

Grammatical analysis is a skill which—like many other skills—is best acquired and honed through its application to concrete reality, e.g. to words and sentences as found in authentic contexts, and by writing grammatical descriptions of (fragments of) natural language. At the same time, practical exercise sessions with a tutor are among the more costly modes of teaching. This is a pedagogical problem in times when many educational institutions experience financial cutbacks and the number of teacher hours per student decrease.

Much would thus be gained if we could offer the students good self-study materials for grammar practice, good in the sense that they would be pedagogically sound, but also in the sense that they actually would be used by the students. Because of this, we are interested in investigating how computer-based grammar instruction material should be designed. For theoretical inspiration we have turned to the findings of the research in Computer-Assisted Language Learning (CALL). The reasons for this is that both CALL and our present aims are about computer-based language-related training and that CALL is a large and growing research area.[1] And, indeed, as we will se below, many of the insights of the CALL community seem to be directly relevant to the case at hand.

2 Pedagogical considerations

There are some pedagogical points that we wish to raise in connection with the design of a grammar tutoring system for LST students, and which we feel are inadequately addressed in existing systems of this kind.

[1]This connection is natural to us also because we offer CALL as one of the specialisations in language engineering.

2.1 The importance of authenticity

Several pedagogical systems support training in formal grammar writing (Gazdar and Mellish 1989; Antworth 1990; von Klopp and Dalton 1996; McConnel 1995; Beskow et al. 1997; see also Rogers 1998). In most cases these systems only deal with grammars from an abstract point of view, without calling attention to the issue how well a grammar accounts for real language. These systems do however offer the students valuable facilities, e.g. allow them to evaluate a grammar by using it to parse arbitrary strings or for random generation. For our purposes, these systems are "realistic" in one sense, namely in that they let students express linguistic generalisations in formalisms which are similar to those actually used by language technologists.

In another sense, however, systems of this kind are spiritually kindred to the "intuitive" method in generative grammar, rather than to the goals of language engineering. The issue of how relevant data is to be found and used is normally left out of the picture altogether. This is a major pedagogical defect as the step from understanding grammars as formal systems to understanding them as theories about existing language use is both crucial and intellectually demanding. It is our experience that this is one of the most difficult aspects of education in formal grammar. We consequently think that there is much to gain by the use of a tutoring system that helps the student to see how a grammar relates to a morphosyntactically annotated corpus. The aim of the work described here is to develop a system which will introduce grammar writing as an empirical process with the aim of accounting for *authentic* language.

2.2 Divide and conquer

The use of a tagged corpus as a testing ground for fledgling formal grammar writers confers another advantage which is often absent from the systems referred to above. Since the aim of these systems is to train the students in writing syntactic or morphological rules, the lexicon is more often than not reduced to the absolute minimum—both in the number and in the complexity of entries—needed to illustrate how the syntactic or morphological rule system works. This is indeed a problem, but it can not be solved simply by urging the students to compile extensive lexicons. On the contrary, there is a clear pedagogical point to the separation of the grammar from the lexicon for training purposes. Generally, it is a good principle to present new material a little at a time, in conceptually coherent portions. Otherwise, the students may get confused, and as a consequence frustrated. In this case, you would like to offer them a ready-made lexicon which should be flexible enough to accommodate a number of grammar formalisms (a "poly-theoretic" lexicon). A morphosyntactically tagged corpus can be made to stand in for such a lexicon, at least in some respects; in addition to the purely linguistic information contained in it, there is also (implicit) information about frequencies of occurence in authentic language, about collocations, etc. Even if there is lexical information which will not, as a rule, be found even in a fairly richly annotated corpus (e.g. valency information and semantics), the information that you *can* find there still constitutes a vast improvement over the typical lexicons of grammar training systems.

Conversely, the tagged corpus makes an excellent basis for exercises aiming at learning to identify the "atoms" of grammar, i.e. parts of speech and inflectional categories, in a realistic context. There are some tutoring systems for this purpose (e.g. Qiao 1996), including one (Mats 1999) that we have been trying out in our department recently. McEnery et al. (1995) compare another such system (the one described by McEnery et al. 1997) to traditional human teaching in a controlled evaluation procedure, and reach the conclusion that the corpus-based computer-assisted method yields slightly better learning results.

2.3 First things first

It is a good pedagogical principle not only to divide that which is to be learned into manageable chunks, but also to proceed from simpler to more complex knowledge. Ideally, the tutoring program should impose exactly this ordering for those students that need it (see Laurillard 1996). The morphosyntacially annotated corpus puts at the students' disposal a "lexicon" which will tag along, as it were, as

- they learn to identify not only which part of speech a certain text word is, but also which inflectional information should be associated with it;
- their grammars evolve in terminal complexity from simple phrase structure rules with atomic terminal categories, to unification-based grammars with feature structures encoding the full morphosyntactic information for each lexical unit;
- their grammars evolve in nonterminal complexity, enabling them to analyse increasingly larger portions of the corpus.

2.4 Learning by inquiry

A corpus-based grammar tutor shares with corpus-based CALL in general the trait of being eminently suited for hypothetico-deductive, problem- and data-driven learning ("serendipity learning"; cf. Flowerdew 1996, or "learning by inquiry"; see McArthur et al. 1995). By working with the program the student will develop his skills in evaluating a grammar as an account of the syntactic phenomena found in a corpus. The system will support a process of thinking that highlights important aspects of scientific reasoning. Abstract concepts such as theory, data, precision, recall and prediction are illustrated in a fairly concrete manner, as are (other) basic aspects of formal grammar.

3 The grammar tutor

With these aims in mind, we are developing a corpus-based grammar tutoring system. We are aiming at first for a system with limited functionality—both in order not to overreach ourselves and to facilitate evaluation—which will undergo several rounds of *formative evaluation* (see Laurillard 1996).

The system will provide a learning context that in important respects is a realistic one. The students will work with authentic linguistic material, in the form of a tagged corpus, and use pedagogically adapted versions of formalisms and tools that they will be using also later in their professional life. This is similar in spirit to the approach taken by McArthur et al. (1995), who argue persuasively for the use in education of so-called ESSCOTS (Educational Support Systems based on Commercial-Off-The-Shelf software). They report both an unusually short system development time and good learning results (in an experiment where they adapted a commercial Geographic Information System (GIS) for use in an educational setting).

The system will be used and evaluated in the context of one or more of our LST and linguistics courses (formal syntax and computational syntax, at least, possibly also basics of grammar), starting in the autumn term of 1999. The evaluation will not be carried out as a test group–control group setup. This is mainly for practical reasons, our student population being too small for this kind of experiment.[2] Instead, we will use in-class observa-

[2]There are also theoretical motivations for this, as there have been serious concerns voiced in the literature about the meaningfulness of such "experiments" in the context of computer-assisted learning (see Borin 1998).

tion, questionnaires and interviews with the students and teachers, and logging of student activity as our main evaluation instruments. The evaluations will, hopefully, yield two kinds of result. Firstly, we expect to learn something about the effectiveness of using a corpus-based computerised grammar tutor, and, secondly, we will see what should be changed and what added in the system (this is what the "formative" part is about).

3.1 Corpus and exercise types

As just stated, any annotated corpus could form the basis of the grammar tutor. As our point of departure, we have chosen to use a Swedish one-million-word balanced corpus, the *Stockholm Umeå Corpus* (SUC; Ejerhed and Källgren 1997).[3]

For the first version, there are two grammar exercise types under development: The most basic exercise is to assign part of speech and morphosyntactic features to words in the corpus. This exercise exists in a preliminary version (Mats 1999), which has been used at our department with encouraging results.[4]

The second step is the formulation of grammatical rules and applying them to the corpus with the help of a built-in parser. Random analysis and generation with the same grammar will also be supported. The system will eventually support two formalisms, plain context-free grammar and a feature-structure based one.

The parser helps the student to evaluate his/her grammar by making clear which analyses the grammar assigns or fails to assign to the substrings of the corpus. One grammatical category (non-terminal symbol) is selected as the one being of particular interest for the moment. The parser locates all strings that are generated as instances of that category. The corpus provides the lexical nodes, i.e. the text word–category pairings. By inspecting these analyses the student will be in a position to decide, with respect to a certain category, to what extent the grammar accounts for the instances of the category and to what extent it overgenerates. The tokens found may be listed (with context) or graphically indicated in the running corpus text. This exercise will encourage the student to evaluate a grammar in terms of its precision and recall with respect to the selected category. The student's own grammar-related intuitions are, of course, important in this kind of corpus-oriented setting, as only the words of the corpus are tagged (it is not a treebank). In other words, there is no predefined right answer available (but see below). The evaluation of the student's performance is rather based on his/her own judgments. This is an example of so-called *intrinsic feedback*, which is the best kind of feedback, according to several CALL practitioners; see Laurillard 1996. Nevertheless, the system will ensure that the application of these intuitions and the reasoning about the grammar will be supported by considerations of concrete data.

The tagged corpus may also be used for random generation. The text word–category pairings define a lexicon which generates expressions of various categories in conjunction with the student's grammar. In this way the lexical material of the corpus and the grammar are used to make grammaticality predictions. The generation exercise will mainly throw light upon how overgeneration problems are discovered and dealt with.

The system gives some feedback about the status of the grammar. Warnings are issued if some category is left undefined. The number of rules, categories, and features used is also reported. This is intended to alert the student to the issue of how simple/complicated the grammar is, which is important as simplicity is one of the most important aspects of theoretical adequacy.

3.2 Parts of speech and grammars

The tagging provides the link between the student's grammar and the given corpus data. It is therefore crucial which categories are used. As the empirical material is a selection from a particular corpus, the tags visible to the student must be derivable from the tagset used in that corpus. Of course, these tags may be mapped onto the tags of the tutoring system in various ways.

The system comes with two predefined mappings from corpus tags to grammar categories, to context-free categories on the one hand and to feature structures on the other. These mappings are defined in a file and can be revised by the teacher. Manipulation of this mapping can, of course, also be a part of more advanced exercises for the student.

[3]SUC was compiled and semi-automatically tagged in the years 1989–1996 (Ejerhed and Källgren 1997). The corpus follows the Brown Corpus format: There are 500 text chunks of approximately 2000 words each, with a genre distribution similar to that in other balanced corpora, although only the written standard language is represented. A corrected second version of the corpus is due to appear before the end of 1999.

[4]It has been tried out during the spring term of 1999 with a group of computer science students taking a course in language engineering in our department (Mats 1999). The students were largely positive in their evaluation of the exercises, but they also suggested some improvements in the user interface and in the way the material was presented to the user. We will incorporate some of these suggestions in the next version of the exercise.

As mentioned, the system will support two grammar formalisms—corresponding to the two tagset mappings just mentioned—pure context-free grammar and a feature-structure formalism, the latter in the style of PATR-II (Shieber 1986).

A context-free grammar is (by definition) used together with a flat taxonomy of lexical categories. As the default option, the program operates with such an inventory of categories which is related to the traditional part-of-speech system, but more fine-grained.

The feature structure tags used with the PATR-II-style formalism correspond to the full information in the tagset used, i.e., they contain primarily inflectional information, in addition to the syntactic category. This means that the corpus mainly will support constraint-based accounts of agreement phenomena. However, the system as such will allow descriptions dealing with arbitrary aspects of grammar.

3.3 Implementation

As the implementation language we have chosen Java, primarily because of its platform-independence and because it is an excellent language for rapid prototyping of applications with sophisticated GUIs, but also to some extent because of its association with the Internet and the WWW (see below).

3.4 Planned developments

Explicit evaluation of the students' actual use of the system will, needless to say, provide the main indication of how the system should be improved and extended. The implemented exercises have nevertheless been designed to fit into a scheme of logically linked exercises, which step-wise lead the students on to more complicated and difficult tasks.

The present system could in a natural way be extended to deal with a corpus which is preanalysed also with regard to constituent structure. A less advanced task for the student would then be to write a grammar that agrees with the given structure(s). The system would provide detailed feedback evaluating the ability of the grammar to generate the given syntactic structures. This exercise would illustrate the purely formal aspect of grammar formulation. It could preferably be used as a preparation for the exercises relying on intrinsic feedback from the student's own grammatical intuitions.

Another valuable addition to the system would be a module that encourages the student to organise the empirical evaluation in a systematic way. The compilation and use of *test suites* provide an often used and simple method with this advantage. A test suite for a certain category is a list of known instances of the category and a list of strings that are known not to belong to the category. A test suite thus provides a collection of data against which a grammar may be automatically evaluated. The system reports the number of positive instances the grammar fails to account for and the number of overgenerations. This exercise shows how the empirical evaluation of a grammar may proceed in a more systematic fashion and encourages trial and error experimentation with the grammar formulation.

Another dimension of difficulty is given by the two grammar formalisms. The basic idea is that the system should be a pedagogically organised toolbox for grammar formulation and corpus inspection (taking the ideas presented in Lager 1995 one step further) and this idea makes it natural to integrate various extensions into the system, such as new inspection tools and other grammar formalisms and parsers, e.g. that described in Dahllöf 1999 or finite-state formalisms for syntax (e.g. Karlsson et al. 1994) or morphology (e.g. Karttunen 1993).

In the context of feature-structure grammars, a *unification-failure explanation generator* is useful. This component indicates which feature mismatch(es) made it impossible for the grammar rules to assemble a certain phrase. A simple version of this facility is implemented in Dahllöf (1999) and it has turned out to be very useful during grammar construction. Pedagogically developed versions of it would likely be valuable for students (and professionals) as it often is very difficult to see how feature-assignments interact in a constraint-based grammar and to locate the source of unwanted unification failures.

A longer-term goal would be to provide the system with intelligent error analysis and help facilities. This is an exciting but largely unexplored research topic in CALL, known as Intelligent CALL, or ICALL, which draws on research in the fields of Artificial Intelligence and Computational Linguistics.

It would also be desirable to develop some kind of authoring interface to the system. Direct manipulation of the system's Java code would presuppose fairly advanced programming skills and this would presumably make it impossible for most teachers to adapt the system to new learning tasks. An authoring facility, allowing users to define new exercises in a suitable authoring language, would consequently extend the usefulness of the system. Such an interface can also be given a more direct pedagogical motivation: There are CALL applications where students step into the role of the teacher, as it were, designing exercises (as if) for their fellow students, and learning about the subject matter in doing so (see Borin 1998).

In its first version, the grammar tutor will, for practical reasons, be accompanied by written instructions and conventional coursebooks. We do however intend to integrate this information in the system.[5]

3.5 Benefits from Internet use

As we mentioned above, our choice of Java as the programming language for the grammar tutor was only partially motivated by its status as *the* programming language of the World Wide Web. Rather, we chose it because it is platform-independent and because the GUI capabilities we need are built into the language.[6] Thus, the application was not built with the WWW in mind, although it is fully feasible to use it over the Internet. In this case, a possible division of labour could be implemented, where the exercise programs are Java applets locally executed in the student's computer, while the corpus resides in a server-side database.

From the experiences of the CALL community, we know that the Internet can bring two distinctly different kinds of pedagogical added value to a learning situation:

1. In this case, the pedagogical value is only incidental upon the general advantage of a client-server setup, i.e. that it is easier to maintain and upgrade an application if you only have to do it once and in one location. For a CALL application, this means that data and exercises probably can be updated more often than otherwise would have been the case.

2. The other case turns around using the Internet as a widely accessible time-of-day-independent communications network. Thanks to the Internet, students and teachers, who may be geographically far apart, can collaborate both asynchronously and synchronously in creating an optimal virtual learning environment for some types of learning tasks (Pennington 1996; Warschauer 1996; Levy 1997; Borin 1998).

The grammar tutoring system has been designed with self-study in mind, so that it is hard to see how it could benefit pedagogically other than incidentally—i.e. as in (1) above—from being made into an Internet application. On the other hand, positive learning effects have been noted in situations where students cooperate in front of the computer to do the exercises in a CALL program designed for self-study (Chapelle et al. 1996). This points to the possibility of designing for a more central role of the Internet even in a program such as the one discussed here. Thus, for instance, grammatical analysis could be carried out collaboratively (or competitively) by several students over the network.

4 Conclusions and future prospects

Summing up, we propose to let LST students learn grammatical analysis and formal grammar writing by practicing these skills with the help of a tutoring system which provides a learning environment which in many ways is a realistic one, both as regards data and formal methods. The grammar exercises will have an empirical connection to authentic language in the form of a tagged corpus; the formalisms and tools will be of the same kind as those used in "real-life" LST; and grammar formulation will be presented as a case of hypothetico-deductive problem solving.

The pedagogical adaptation consists, as of now, in the following: (1) arranging the learning situation so that the students' practice at each moment is focussed on one component of the subject which is to be learned, while keeping the other components as realistic as possible; (2) going from the simple to the complex; (3) making sure that there is adequate feedback (preferrably intrinsic) at all times. Further, we plan to evaluate the grammar tutor continuously in actual LST and linguistics courses in our department, both in order to assess its effectiveness as a learning aid and to give us guidance in its further development.

[5]This matter may deserve some deliberation. Benyon et al. (1997) point out that turning written coursebooks directly into hypertext rarely yields good results, and in Nygren (1996), on the basis of practical experiences of medical information systems, we are warned that paper-based information often loses in lucidity and navigability as a result of it being poured into a computer.

[6]The AWT and JFC class libraries.

References

Antworth, Evan L. 1990. PC-KIMMO: A two-level processor for morphological analysis. *Occasional Publications in Academic Computing* 16. Dallas: Summer Institute of Linguistics.

Batstone, Rob 1994. *Grammar.* Oxford: Oxford University Press.

Benyon, David, Debbie Stone and Mark Woodroffe 1997. Experience with developing multimedia courseware for the World Wide Web: the need for better tools and clear pedagogy. *International Journal of Human-Computer Studies*, 47 (1), 197–218.

Beskow, Björn, Torbjörn Lager and Joakim Nivre 1997. Linguistic Instruments: Grammar Laboratories for the Macintosh. `http://www.ling.gu.se/~li/`.

Borin, Lars 1998. Datorstödd språkinlärning. Dept. of Linguistics, Uppsala University. MS.

Chapelle, Carol, Joan Jamieson and Yuhsoon Park 1996. Second language classroom research traditions: How does CALL fit? In *The Power of CALL*, ed. by Martha C. Pennington. Houston, Texas: Athelstan, 33–53.

Dahllöf, Mats 1999. Flexible typed feature structure grammar. Dept. of Linguistics, Uppsala University. MS. `http://stp.ling.uu.se/~matsd/ftfsg/`.

Ejerhed, Eva and Gunnel Källgren 1997. Stockholm Umeå Corpus Version 1.0, SUC 1.0. Dept. of Linguistics, Umeå University.

Ellis, Rod 1985. *Understanding Second Language Acquisition.* Oxford: Oxford University Press.

Flowerdew, John 1996. Concordancing in language learning. In *The Power of CALL*, ed. by Martha C. Pennington. Houston, Texas: Athelstan, 97–113.

Gazdar, Gerald and Chris Mellish 1989. *Natural Language Processing in LISP.* Wokingham: Addison-Wesley.

Karlsson, Fred, Atro Voutilainen, Juha Heikkilä and Arto Anttila (eds) 1994. *Constraint Grammar: A Language-Independent Formalism for Parsing Unrestricted Text.* Berlin: Mouton de Gruyter.

Karttunen, Lauri 1993. Finite-State Lexicon Compiler. Technical Report ISTL-NLTT-1993-04-02. Xerox PARC, Palo Alto, California.

Lager, Torbjörn 1995. *A Logical Approach to Computational Corpus Linguistics.* Diss. Dept. of Linguistics, Göteborg University.

Laurillard, Diana 1996. The TELL Consortium – formative evaluation report. `http://www.hull.ac.uk/cti/formeval.doc`.

Levy, Michael 1997. *Computer-Assisted Language Learning.* Oxford: Clarendon Press.

Lightbown, Patsy M. and Nina Spada 1993. *How Languages are Learned.* Oxford: Oxford University Press.

Mats, Erik 1999. Språktåget: en webbaserad programvara för datorstödd språkinlärning. MS. `http://stp.ling.uu.se/~erikm/spraktaget/`.

McArthur, David, Matthew W. Lewis and Miriam Bishay 1995. ESSCOTS for learning: Transforming commercial software into powerful educational tools. *Journal of Artificial Intelligence in Education*, 6, 3-34.

McConnel, Stephen 1995. PC-PATR Reference Manual. `http://www.sil.org/pcpatr/manual/pcpatr.html`.

McEnery, Tony, John Paul Baker and Andrew Wilson 1995. A statistical analysis of corpus based computer vs. traditional human teaching methods of part of speech analysis. *Computer-Assisted Language Learning*, 8, 259–274.

McEnery, Tony, John Paul Baker and John Hutchinson 1997. A corpus-based grammar tutor. In *Corpus Annotation*, ed. by Roger Garside, Geoffrey Leech and Anthony McEnery. London: Longman, 209–219.

Nygren, Else 1996. *From Paper to Computer Screen. Human Information Processing and User Interface Design.* Uppsala: Acta Universitatis Upsaliensis.

Pennington, Martha C. 1996. The power of the computer in language education. In *The Power of CALL*, ed. by Martha Pennington. Houston, Texas: Athelstan, 1–14.

Qiao, Hong Liang 1996. Processing the Lancaster Parsed Corpus as data for a CALL program: The design and implementation of the Word Class Drills v1.0. *Computer Assisted Language Learning*, 9, 163–180.

Rogers, Henry 1998. Education. In *Using Computers in Linguistics. A Practical Guide*, ed. by John M. Lawler and Helen Aristar Dry. London: Routledge, 62–100.

Shieber, Stuart 1986. *An Introduction to Unification-Based Approaches to Grammar.* Lecture Notes No. 4. Stanford, California: CSLI.

Underwood, John H. 1984. *Linguistics, Computers and the Language Teacher. A Communicative Approach.* Rowley, Massachusetts: Newbury House Publishers.

von Klopp, Ana and Chris Dalton 1996. Interactive teaching material for the World Wide Web: A linguistics tutor written in Java. Department of Linguistics Research Reports UWB-LING/RR96-01. University of Wales, Bangor. `http://www.bangor.ac.uk/ling/staff/avk/icalm.html`

Warschauer, Mark 1996. Computer-assisted language learning: An introduction. In *Multimedia Language Teaching*, ed. by S. Fotos. Tokyo: Logos International, 3–20.

WWW documents as accessed on April 19th, 1999.