Expressing the Same Piece of Information Using Different Elements of Language: A Text-Based System

István Kecskés and László Agócs
Kossuth University, Debrecen, Hungary

ABSTRACT: STOBI is a text-based system combining the positive features of both authoring and generative techniques. The main goal of the system is to help students understand the interrelation between form and meaning and practice expressing the same piece of information by using different elements of language. The program runs on IBM PC (compatible).

KEYWORDS: formalizing meaning, artificial intelligence, text-generation, vocabulary lists, interactive mode, services, information-reconstruction, creativity, exploration

Meaning Rather Than Form

If we consider the present state of CALL software development, we may conclude that there is a strong tendency to produce programs which concentrate on meaning rather than on form.

No one has ever denied the importance of meaning in language learning. In Halliday's words, learning a language is learning how to mean, and to learn a foreign language is to learn how to mean in a new way.

Program designers have to face at least two difficult problems when developing *meaningful* software:

a) What are the criteria for *meaningful* programs?

b) How do we formalize meaning for the computer?

To answer the first question is not easy at all. Let us illustrate with an example. If, the students are invited to do a storyboard type of exercise or a cloze test on the computer we can hardly say that it is mainly meaning they have to concentrate on. This is because the machine accepts only one form at one place in the text, in spite of the fact that the context may make it possible for them to use at least one alternative in the very same place. For example:

The weather was miserable/terrible.

There was a fierce/harsh/wild wind blowing.

The main task in these types of exercises is to reconstruct the original text word by word and learners may often feel trapped when *playing the guessing game* and trying to find only the particular word which is acceptable to the computer. This

is true even if we acknowledge that the nearer the text gets to completion, the more meaning comes to the fore.

In fact, one of the most important characteristic features of meaning is that it can be expressed by using different language forms: the information these forms convey remains approximately the same. According to Culioli, language forms and patterns do not *have* meanings, rather utterances take on meaning to the extent that the operations that they effect are differentiated, and the meanings which can be associated with words are derived from the situational function of these operations (cited by Farrington in Farrington 1988:16). This fact cannot be ignored when we are trying to develop a really *meaningful* exercise on the computer.

On the other hand, we are still very far from being able to formalize meaning. So far we have managed to do this only if we avoid attempting to model the competence of a native speaker and instead try to remain within a restricted domain of the language. Only in this case can the computer be expected to process any sentence (almost every sentence) that is at all likely to occur. In this way, the computer program can produce *the illusion* that it is able to cope with a wide range of real linguistic inputs. This kind of illusion is very important for the learner. Only the linguist and the programmer are aware of how unintelligent the system is.

We agree with Farrington when he says that a CALL program does not have to be all intelligent (Farrington 1988:25). But we must add that this statement can be accepted only as far as programming is concerned. It is not always possible to find elegant solutions to problems when a program with a language teaching aim is being developed.

The organization of the linguistic material, however, is another question. This should be made as intelligent as possible.

Requirements

When we started to develop our STOBI program, we wanted it to comply with the following requirements:

1) It should be a text-based system which can handle a wide range of meanings within the domain of each particular text. (Text is to be understood here in the narrow sense of written narrative.)

2) Generative techniques must be combined with authoring ones so that the system is open for teachers to enter their own material (text).

3) The system must offer as many *services*, options and exercises as possible so that the learning process can be organized by the users according to their needs.

Text-Generation

The whole structure of the program is inspired by the fact that any piece of information can be conveyed by using different elements of the language: the

content of the information remains approximately the same. So a text-generator has been developed which is able to produce the same information, using a great variety of forms of the language. Our earlier generative programs are equipped with a built-in grammar and a lexicon, which enable the computer to produce the linguistic material (Kecsk6s 1986; 1988). Several endeavors similar to ours have been reported so far (see, for example, Bailin 1988:38) and these types of programs are considered to be quite valuable. However, in STOBI the structure of the generating part of the program has been simplified to a great extent so that teachers are able to use it to enter their own text.

A special word processor is used when constructing the structure of a particular text. (It is a combination of the *traditional* spread-sheets and word processors.) Each sentence in the text has its own scheme (or schemata), which resembles the substitution table that is well-known to every language teacher.

Synonymous words, phrases and structures are used within each scheme to express the same information. These elements should be chosen with exceptional care because coherence and cohesion must be maintained among the elements of each possible sentence and the sentences of each possible text.

The database can be constructed on the screen, which is usually a time-consuming and laborious process. Much depends on the inventiveness and creativeness of the *author* (the teacher). The more elements and structures the author finds to express the same piece of information, the more varied and interesting the work will be with the text that has been constructed.

The generator works in a simple way. It follows one of the possible paths, all of which are bound to be successful, so it never needs to backtrack. When it comes to the end of a sentence, the sentence appears on the screen. Depending on the path that has been followed, the machine has alternative ways of continuing. This means that the structure of the following sentence always depends on the previous one. In this way, the coherence of the whole text can be ensured.

When a text is ready, a check-test is available in the program. Its task is to check whether each scheme works well, i.e. whether all the generated sentences are correct and acceptable. If a false branching or path is detected, the error can easily be corrected.

We have completed the structure of two texts in the program so far. The number of texts is not restricted because they can be saved on floppy disks or hard disks. If need be, any of the texts can be deleted and another added.

The Lexicon

The other central part of the program is the lexicon, which is closely related to the generator.

The lexicon must contain all the items that can occur in any of the possible variations of the texts the computer is able to generate.

Every item has a wide range of characteristics including a part of speech marker and its possible morphological forms. There is also room to write in the definition of every lexical unit in the users' first language. For example:

go (v.)	goes, went, gone, going	.....
(definition)		
woman (n.)	women	.....
(definition)		

The first step in entering a new text into the database is to construct the generation part of the program. The check-test is followed by a vocabulary test. This means that the computer tries to identify all the lexical items that occur in the generation part of the program. Every form (whether finite or non-finite) is compared to the items the lexicon already contains.

An important feature of the lexicon is that it works in an interactive mode. If a certain form has been identified, the computer reveals the entry to which this work may belong. It is the user who has to decide whether this form is the one he requires. If not, a new entry must be opened in the lexicon for the new word.

This interactive procedure is of great importance for various reasons. For example, this is how the problem of homonyms can be solved.

There are different entries for homonyms, for example:

design (n.)	designs
design (v.)	designs, designed, designing

A special marker is used to indicate in which text the given word occurs and which part of speech it belongs to in that text. This piece of information is needed in the exercise that is based on the reconstruction of various parts of speech which are deleted from the text (example follows). Therefore, there is no ambiguity if the same form with different functions appears in the same text or two different texts.

New Entries

When the need to open a new entry in the lexicon occurs, it is not necessary for the teacher to type in all the possible forms of a verb or a noun if they have only regular forms. The computer can generate them be adding the necessary morphological endings to the word stem:

open (v.) -->	open-s
	open-ed
	open-ing

book (n.) --> book-s

Irregular forms must be entered by the teacher. (Forms are considered to be regular only if the word-stem does not change at all.)

When the program is running, the lexicon can be consulted at any time by pressing the F2 key. The required word has to be typed in the form it is found in the given text. For example:

Search pattern: given

If the word which this form belongs to is in the lexicon, its entry is revealed in a few seconds:

give (v.) gives, gave, given, giving

In the case of homonyms, all the possible entries can be asked for:

Search pattern: orders

order (v.) orders, ordered, ordered, ordering
order (n.) orders

Utilities

Several attempts have been made to outline the most important requirements for an intelligent CALL system (Higgins & Johns 1984; Davies 1985; Kecsk6s 1988). All the authors agree that a program must be intelligent not only linguistically but pedagogically as well. Complex and flexible programs are needed which concentrate on the kind of pedagogically valid activities that can be provided by the computer during the teaching/learning process.

To meet these requirements, we have made several *utilities* available for the user (teacher) within our program. They can support both classroom activities and individual work. We call them *utilities* because their main function is to prepare the user for the tasks which are available in the program.

The program has a cyclical structure with a menu system, which makes it possible to move back and forth whenever needed (see Figure 1). Work with the program begins by choosing one of the texts from the list.

Generation of a certain text can be guided either by the user or done by the computer at random. In the case of individual work, it is advisable for the learner to ask the machine to make up the text. An important feature of the program is that the text can be hidden until the type of exercise has been chosen. This can make the task more difficult if missing elements are to be reconstructed or jumbled sentences reordered. But there is, of course, an option to see the text

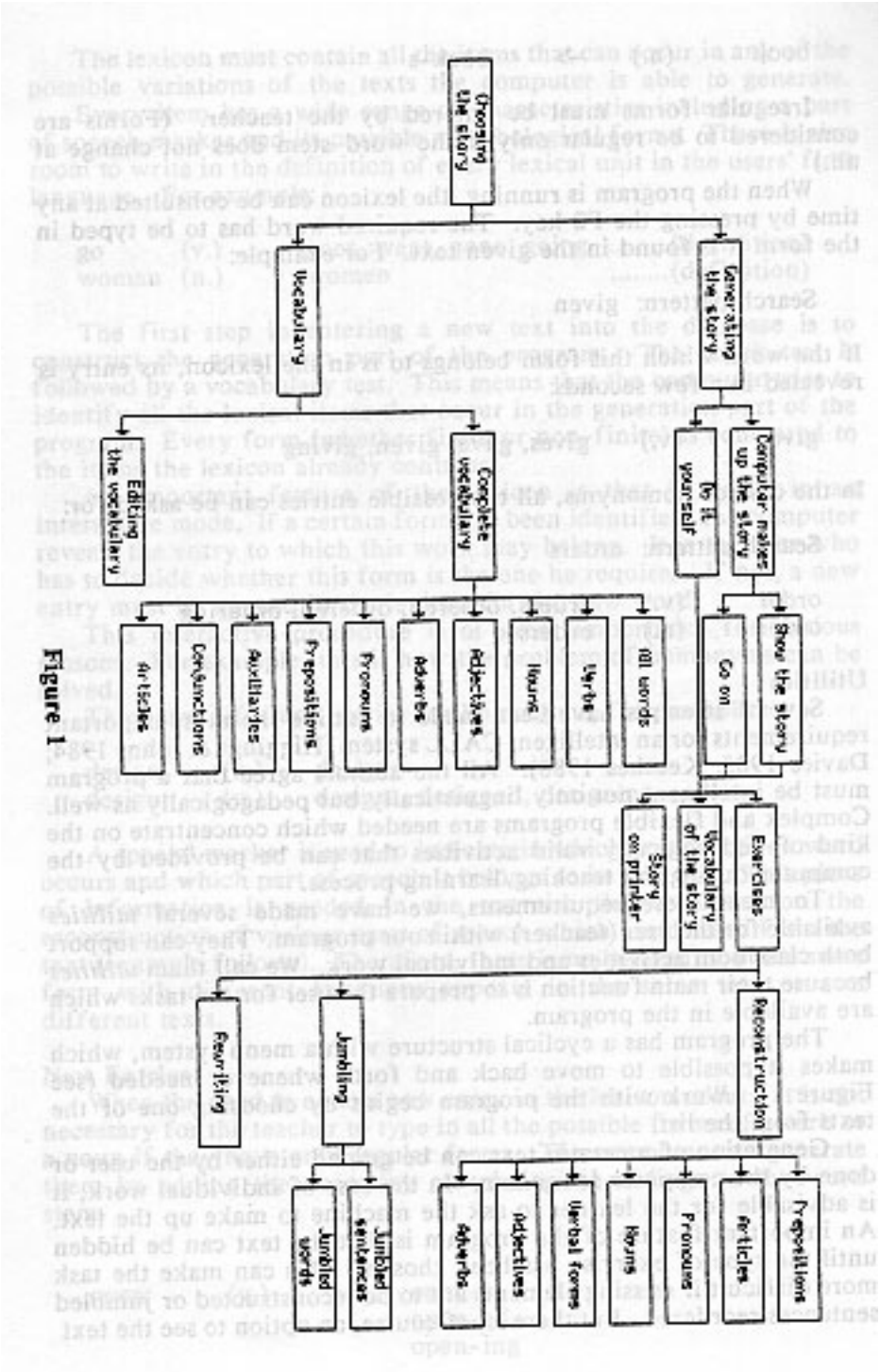
Choosing the story
Vocabulary
Generating the story
Editing the vocabulary
Complete vocabulary
Do it yourself
Computer makes up the story
Articles
Conjunctions
Auxiliaries
Prepositions
Pronouns
Adverbs
Adjectives
Nouns
Verbs
All words
Go on
Show the story
Story on printer
Vocabulary of the story
Exercises
Reconstruction
Rewriting
Jumbling
Jumbled words
Jumbled sentences
Adverbs
Adjectives
Verbal forms
Nouns
Pronouns
Articles
Prepositions

Figure 1

before choosing from the exercises. If the teacher wants the students to concentrate on certain lexical items or grammatical forms, it is possible to build up the text according to these needs.

When generation is guided by the user, the computer offers several alternative versions for each sentence. If, for example, the teacher wants the students to practice using prepositions, he will make up a text which contains as many prepositions as possible.

Schemas that stand for each sentence are still the major means of handling meaning within AI. If we use the combinatorial principles, the same statement can be expressed by various sentence types. For example:

Two women were advancing slowly across the plain.
going
making their way

There were two women advancing slowly across the plain.
going
making their way

The teacher who is to guide generation is allowed to choose from among the various sentence types which might contain different lexical elements to express the same piece of information.

The generated text can either be displayed on the screen or printed. This means that the program may also be useful as a story-generator. If a text is needed, it is available on printer in as many variations and copies as the teacher requires. Meaningful communicative activities can follow without the computer, based on the variations of the text at hand.

Vocabulary Utility

The other type of utility is connected with the lexicon. Recently there has been a tendency in language teaching to emphasize the importance of a systematic approach to vocabulary teaching (Swan 1985:7-8). To help this process, three types of vocabulary lists are available which can either be displayed on screen or printed. The first one contains all the lexical items that can occur in any of the variations of the texts which have been saved on disk so far. Words can be asked for either in the form of a simple list or classified according to their part of speech marker.

The second word list consists of the lexical units of each separate text. When a story is chosen for elaboration, the students can become acquainted with all the items that can occur in any variation of the text. This makes it easier for the teacher to plan the work with the program.

The third type of vocabulary list contains the words of the particular text that has been generated by the computer.

Exercises

Three different types of exercises have been built into the program so far: reconstruction, jumbling, and rewriting. The first two are based on the well-known *storyboard* operations of Higgins and Johns (1984). But in the reconstruction part there is an important innovation. Because the program is able to distinguish parts of speech, the computer can be ordered to omit prepositions, articles, pronouns, nouns, verbal forms, adjectives, and adverbs from the text.

The program can distinguish between verbal forms and modal auxiliaries. By *verbal forms* we mean all the possible morphological forms of lexical verbs and primary auxiliaries (do, have, be). Modal auxiliaries are the following: *can, could, may, might, shall, should, will, would, must, ought to, used to, need*, and *dare* (Quirk & Greenbaum 1973:26-37). When compound verbal phrases occur, the program makes the following distinction:

He would have gone there.
| | |
mod.aux. v.f. v.f.

Having been given a map, John could find his way.
| | | | | |
v.f. v.f. v.f. mod.aux. v.f.

A part of speech item may consist of more than a single word. This is especially common in the case of complex prepositions, such as *in spite of*, *out of*, etc.

If a word may belong to different parts of speech, this is noted in the lexicon, where there are different entries:

He has a fast car.
|
adjective (a.)

He always drives fast.
|
adverb (ad.)

As mentioned previously, the text in which the missing forms are to be reconstructed can be either revealed to the student or kept hidden until the operation starts.

An important feature of this exercise is that synonyms are also accepted. For example:

There was a wild wind blowing
fierce
harsh

The man gave them something to eat.
offered

They stayed in the house.
remained

The second type of exercise is unscrambling. Either jumbled sentences or jumbled words can be requested. In the case of jumbled words, a sentence is chosen at random from the text by the computer. The words have to be reordered according to the context. If there is more than one possible word order, the computer accepts any correct answer.

Reconstruction of Information

Rewriting is a new feature. This is the central part of the program, in which it is not the original text that has to be built up again, but the piece of information it conveys.

The best-known and most popular CALL programs reported so far require activities which consist of the reconstruction of a completely blanked-out text, various cloze exercises, unscrambling jumbled words and sentences, guessing missing elements, etc. All of them base the changes on the same basic idea and have something in common: it is always the original text that has to be built up again, word by word.

While acknowledging the enormous impetus these types of programs have given to the development of CALL over the past few years, Phillips suggests that there are certain difficulties and limitations inherent in the model (Phillips 1987:278). In his opinion, the most fundamental difficulty centers on a crucial distinction between real and realistic uses of language. The real use of language is when it is used to achieve some outcome in the real world. These programs are only realistic rather than real, since there is an intrinsic artificiality in the activities (Phillips 1987:278-279).

We can agree only partly with Phillips' concern. It is indeed true that this type of program has certain limitations, but not because it concentrates only on realistic rather than real uses of language. We are convinced that programs which are able to handle real uses of language can be expected to be developed only if we have a device which is successful in modelling the ability of humans to use language. Artificial intelligence research has made serious efforts to improve the ability of machines to handle natural human language but we are still very far from having a truly intelligent expert system.

Therefore, our expectations must be limited to the present state of the art. At this stage of development, only the realistic uses of language can be modelled in some form within a CALL program. Notwithstanding this, we must do our best to approach as closely as possible the real nature of language.

Let us use the example of text processing. When a learner encounters a text, he tries to understand it. When the message conveyed by the piece of language has been understood, the learner can be expected to reproduce it. The question arises: which task is more natural, to ask the student to reproduce the text word by word or the information conveyed by the text? There is no doubt that it is difficult and unnatural to limit the creative language competence of a human being to the "slavish" reproduction of a language chunk written or constructed by another human being. One of the most important features of language use is creativity. This should be limited as little as possible when language learning is in question. (We do not want to deny, of course, that there are sometimes cases when the reproduction of a text or a dialogue can be a useful exercise in the learning process.)

According to our survey the most frequent objection of student to *storyboard* type of activities is that "only one word can be used in one place." This fact prompted us to develop a new type of exercise which tries to reduce this limitation as much as possible.

In the case of the rewriting exercise, the student first has to understand the text generated by the computer. Unknown words can be asked for from the lexicon whenever necessary.

The text is subsequently deleted and several keywords are presented in its stead. It is very important to note that these keywords are also generated at random from the vocabulary list so they may differ from the ones in the original text. (See Figure 2) The user's task is to produce a variation of the text. This can be done by using three different strategies:

1) trying to reconstruct the original text word by word,

2) using the keywords offered by the computer,

3) trying to find lexical items, structures, and phrases other than those in the original text or among the keywords in order to express the same idea.

One of the greatest advantages of the program is that these strategies can be applied simultaneously. The computer is ready to accept almost any correct entry if it is relevant to the information and both grammatically and semantically well-formed. (See Figure 2) Scanning goes sentence by sentence but the text that has already been entered can be edited at any time. This means that even those entries which have already been accepted as correct can be replaced by new ones whenever the user finds it necessary. This may give the exercise an exploratory character (Higgins-Johns 1984:71-73). The learner's competence can be measured against the machine's. The program uses a limited syntactic parser which starts to work on the input sentence and compares it with the models it can generate. If it finds a mistake, which means that a certain word cannot be found in its lexicon, or the input has not been generated according to the models and rules the parser can cope with, it sends out a message on the screen, marking the word which is ambiguous. If the learner and the computer cannot "understand each

other" for some reason, help can be asked for by pressing the cursor down key.

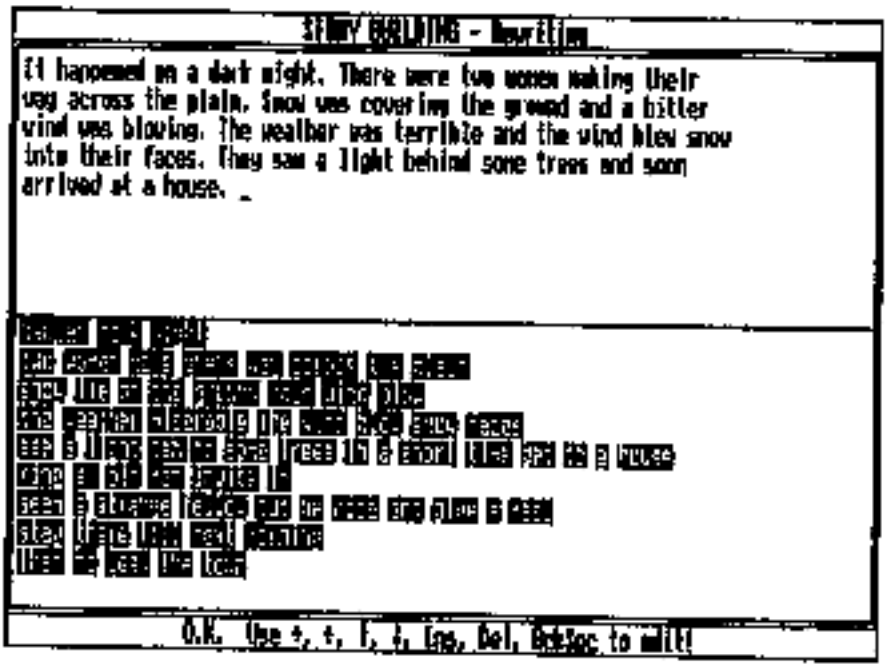

Figure 2

Keywords make an essential part of this exercise because they can guide learners as much or as little as they need. There is no compulsion to ask for them. (We find it very important that the computer should not force the user to follow a certain line or strategy.) For example:

Keywords: night; very; dark

Possible Sentence Formations:

The night was very dark.
The night was pitch-dark.
It happened on a dark night.
It got dark.
It was getting dark.
It was dark.
It was very dark.
It was pitch-dark. (and so on)

As we see, only the piece of information remains the same. The forms that carry this idea can change within a rather wide range. Even if a certain word occurs among the keywords, this does not necessarily mean that it should be included

in the sentence formation. But these non-finite words placed one after the other convey a certain message, and this is what has to be formed by the users. Much depends on their inventiveness and creativity. They are expected to puzzle out what forms can convey the necessary information. Exploring the machine's competence is an exciting task even for those who are more advanced in the language. If words unknown to the learner occur among the keywords, the lexicon can be consulted.

Summary

The development of STOBI is an attempt to combine within the confines of a program the three most important requirements of recent CALL program design: concentration on meaning rather than on form, the use of authoring techniques, and the adjusting of the structure of the program to the needs of teachers and learners.

No single one of these features takes special prominence over the other: they are mutually supportive. Linguistic, technical and pedagogical problems have been solved, bearing in mind the same theoretical standpoint--one in which more weight is given to an integrated approach to the system of language, information processing and a learner-centered methodology.

It is obvious that the consideration of the three requirements at the same time will have meant making certain concessions on linguistic questions. Let us illustrate this with some examples.

We had to simplify the structure of the text generator and the lexicon to a great extent so as to enable the average teacher to enter original material.

Another concession is that part-of-speech recognition works not by using various operations but by different markers that are entered when a new entry is opened in the lexicon.

Both concessions have been made to give the program a sound pedagogical basis, which is all important in a CALL program.

We should like to emphasize that the program is still being developed. One version is ready; it works and has been used by teachers in different classroom circumstances.

Further development entails the following:

a) Making the authoring part even easier to use (technically),

b) extending the scope of the exercises,

c) giving an option to ask for more or fewer keywords on the screen.

These are basically technical considerations now, because the structure of the program has been prepared for the required changes.

References

Bailin, Alan. 1988. Artificial Intelligence and Computer Assisted Language Instruction: A Perspective. *CALICO Journal* Vol. 5, No. 3. 25-51.

Davies, Graham. 1985. *Using Computers in Language Learning: A Teacher's Guide*. (With a section on the use of the computer in English Language Teaching by John Higgins.) London: Centre for Information on Language Teaching and Research, Information Guide No. 22. iv-159.

Farrington, Brian. 1988. AI: "grandeur" or "servitude"? In Cameron, K. (ed.) *Handbook for the CALL Enthusiast*. London: Ellis Horwood. 130 pp.
Higgins, John & Johns, Tim. 1984. *Computers in Language Learning*. New York: Addison-Wesley.
Kecskés, István. 1986. Complex, Cyclical, Generative Programs to Teach Grammar. In: Kecskés, István & Papp, Ferenc (eds.) 1986. *Linguistics and Methodology in CALL*. Budapest: SZAMALK. 37-53.
____. 1988. Computer Programs to Develop Both Accuracy and Fluency. *SYSTEM* Vol. 16. No. 1. 29-35.
Kecskés, István & Papp, Ferenc (eds.) 1986. *Linguistics and Methodology in CALL*. Budapest: SZAMALK.
Phillips, Martin. 1987. Potential Paradigms and Possible Problems for CALL. *SYSTEM* Vol. 15, No. 3. 275-287.
Quirk, Randolph & Greenbaum, Sidney. 1973. *A Concise Grammar of Contemporary English*. New York: Harcourt Brace Jovanovich.
Swan, Michael. 1985. A Critical Look at the Communicative Approach (1). *ELT Journal* Vol. 39/1. 2-12.
Underwood, John. 1984. *Linguistics, Computers and the Language Teacher: A Communicative Approach*. Rowley, Massachusetts: Newbury House.
____. 1987. When the Music Stops: A Review of Using Computers in Language Learning. *CALICO Journal* Vol. 4. No. 3. 35-45.

Authors' Biodata

István Kecskés, Associate Professor in the Department of Linguistics at Kossuth University, Debrecen, Hungary, is the head of a research group which develops CALL programs in English, Russian, German, French and Hungarian as foreign languages within the confines of a project. He graduated from Kossuth University as a teacher of English and Russian and received his Ph.D. in comparative linguistics from Kossuth University and his Candidate Degree in applied linguistics from the Hungarian Academy of Sciences, Budapest. He is the author of several text-books, CALL programs in English, Russian, and Hungarian, a manual on CALL, and a book on Applied Linguistics and Language Teaching.

László Agócs is the Head of the Department of Educational Technology at the University Medical School, Debrecen, Hungary. He graduated from Kossuth University as a teacher of mathematics and physics, and received his Ph.D. in educational technology from Kossuth University. He is involved in several national projects on CAI. He is the author of two books on programming and educational technology and the co-author with Dr. Kecskés of various CALL programs.

Authors' Addresses

Dr. István Kecskés
Kossuth University
Debrecen, Egyetem tér 1.
H-4010, HUNGARY

Dr. László Agócs
University Medical School
Debrecen, Nagyerdei Krt. 98.
H-4012, HUNGARY