Computer-Assisted Language Learning

Proceedings of the seventh
Twente Workshop on Language Technology

L. Appelo and F.M.G. de Jong (eds.)
PREFACE

TWLT is an acronym of Twente Workshop(s) on Language Technology. These workshops on natural language theory and technology are organised by Project Parlevink (sometimes with the help of others), a language theory and technology project conducted at the Department of Computer Science of the University of Twente, Enschede, The Netherlands. Each workshop has proceedings containing the papers that were presented. For the contents of these proceedings consult the last pages of this volume.

Previous workshops.
TWLT4, Pragmatics in Language Technology. 23 September, 1992.
TWLT5, Natural Language Interfaces. 3 and 4 June, 1993.
TWLT6, Natural Language Parsing, 16 and 17 December, 1993.

TWLT7 was devoted to computer-assisted language learning (CALL). It took place in the Vrijhof at the campus of the University of Twente in Enschede, The Netherlands. Just as with the previous workshop programs there were presentations by a select group of international researchers and other experts. In addition to the lectures there were several demonstrations of systems. The general aim was to offer a platform for the presentation of new developments in the field of CALL and for the exchange of ideas between people from the various disciplines that play a role. Among the issues addressed were: the use of natural language processing modules, user-orientation, multimedia aspects the role of advanced information technology, and integration of systems into the curriculum.

A workshop is the concerted action of many people. It goes without saying that we are grateful to the authors and the organisations they represent for their efforts. But in addition we would like to mention here the people whose work has been less visible during the workshop proper, but whose contribution was evidently of crucial importance. In a very initial stage Gerard Kempen was our advisor. Charlotte Bijron, Alice Hoogvliet-Haverkate and the people of convention bureau BASICS took care of the administrative tasks. Without Jos Buis and the laboratory support of the departments of computer science and educational technology there would have been no demonstrations. Dennis Bijwaard assisted us in making these proceedings fit for printing. Marion Keiren we owe thanks for introducing us to the language education network in the Netherlands. Finally we also wish to thank the participants for being there and for contributing to the discussions.

TWLT8, the next workshop in the series, will take place on December 1 and 2, 1994. Its topic will be spoken language understanding. We hope it will match the success of this and the previous workshops.

June 8, 1994

Lisette Appelo
Franciska de Jong
CONTENTS

Introductory Paper:

Computer-Assisted Language Learning: Prolegomena  
L.Appelo, F.M.G. de Jong (IPO / University of Twente)

Workshop Papers:

Eurolinguist test: An adaptive testing system  
M. van Bodegom (Eurolinguist Residential Language Institute, Nijmegen)

Discatex CD-ROM XA  
B. Cartigny (Escape, Tilburg)

Using a Corpus for Teaching Turkish Morphology  
H.Altay Guvenir, K. Oflazer (Bilkent University, Ankara)

Viewpoint Abstraction: a Key to Conversational Learning  
H. Hamburger (George Mason University, Washington)

Learning English with It's English  
J. Jaspers, G. Kanselaar, W. Kok (University of Utrecht)

Towards an integrated system for spelling, grammar and writing instruction  
G. Kempen, A. Dijkstra (University of Leiden)

Contextual vocabulary learning with CAVOL  
F. Kronenberg, A. Krueger, P. Ludewig (University of Osnabrueck)

Teachers, Students and IT: how to get teachers to integrate IT ...  
S. Lobbe (Rotterdam Polytechnic Informatica Centrum)

APPEAL: Interactive language learning in a multimedia environment  
J. Rous. L. Appelo (IPO, Eindhoven)

Developing a multimedia course for learning Dutch as a second language  
B. Salverda (SLO, Enschede)

Error analysis and explanation in knowledge based language tutoring  
C. B. Schwind (Universite de Marseille)

TELL into the mainstream curriculum  
J. Thompson (CTI, University of Hull)

Language in action, or learning a language by whatching how it works  
M. Zock (Limsi, Paris)

Workshop Demos:

The Rosetta Stone  
M. van Bodegom (Eurolinguist Language House, Nijmegen)

A speech-melody trainer for foreign-language intonation  
D.J. Hermes, G.W.G. Spaai (IPO, Eindhoven)

PROMISE: Steps towards communicative English teaching in an interactive ...  
R.S. John (Universität Osnabrück)
A vocabulary workbench for language learners and teachers
M. Klijn-Wuisman, M. Jagtman (Delft University of Technology)
Verbarium and Substantarium: Two CALL programs for Latin
J. van de Plassche, E. Bos, E. Jongen-Janner (SOS, Nijmegen)
Interactive Video in language and culture
S.M. Sanne (Universitetet i Bergen)
A computer BBS in language instruction
T.C. Stewart (University of South Carolina)
Sponsors and Support

We gratefully acknowledge help from:

University of Twente, Enschede

IPO, Eindhoven

Print-basisvorming, Hoevelaken

Leferink, Haaksbergen
Computer-Assisted Language Learning: Prolegomena*

Lisette Appelo
Institute for Perception Research,
P.O. Box 513,
5600 MB Eindhoven,
The Netherlands
appelo@prl.philips.nl

Franciska de Jong
Department of Computer Science
P.O. Box 217
7500 AE Enschede,
The Netherlands
fdejong@cs.utwente.nl

1 INTRODUCTION

The topic of the seventh Twente Workshop on Language Technology (TWLT7) is more application-oriented than any of its predecessors. As a consequence, aspects of design and development are not the sole themes of the workshop. User aspects co-determine the general perspective. There are several reasons for choosing this multiple perspective.

Our general interest and recent engagement in Computer-Assisted Language Learning (CALL) follows from our conviction that research in the field of Language Technology (LT) may profit from the experience with the building and use of real language processing systems. And for reasons to be discussed below, CALL is one of the scarce fields where language processing has really been put to use.

Another reason why we consider it worthwhile to devote a workshop to CALL is the fact that recent developments in Language Technology and the potential of advanced information technology (IT) may have a great impact on the field of Computer-Assisted Language Learning. Both on the features of programs and on the way they will be put to use.

For a start, let us state some predictions.

1. Information technology will undoubtedly influence the importance of CALL in people's education, because

   • class rooms will be more and more equipped with the electronic means required for the use of advanced courseware: personal computers, CD-ROM, CD-I, etc.
   • teachers involved in (language) learning will be more and more skilled enough to use what is available and to introduce it into their courses;
   • personal electronic equipment will evolve considerably in the next decades, both in the level of distribution and in the number of features integrated (sound, speech, video, screen manipulation, etc.);
   • the emergence of a fine-grained network of electronic highways will bring all kinds of services at home, including educational services for tele-learning and tele-training.
   • the more informal game-like software, already a big moneymaker in other areas, will penetrate the market of software supporting the acquisition and mastering of all kinds of skills, including knowledge of language.

2. Application of language technology will become more feasible, because of

   • recent developments in robust parsing methods
   • the emergence of powerful language and speech generation programs
   • promising developments in speech recognition
   • the availability of standardized lexicons and other linguistic resources

*We are grateful to Maaike Schootlemmer for comments on an earlier draft.
3. The multilingual society will grow, partly because of the still growing extension of media technology and the emerging language industry (cf. (Davis, 1993)). This means that there will be a stronger need for advanced language courses.

Now let us assume that all this will come true. What challenges does this scenario raise for researchers and developers in the field of CALL? What can the contribution of Language Technology be? How can those people whose first concern is the quality of language education play such a role that the growth of impact of IT on CALL will lead to something to be welcomed rather than a loss of teaching quality? One of TWLT7’s goal is to give some possible answers to these questions.

TWLT7 brings together researchers, developers and specialists with respect to teaching. If some result is to grow out of all the efforts that will be made and have been made already, the exchange of ideas between these people is of considerable importance. To offer a platform for this exchange is one of the other goals of TWLT7.

In the next section we will briefly point out some trends in CALL and indicate how the contributions to this workshop are related to them (Section 2). In section 3 we will discuss the relation between LT and CALL and make some concluding remarks in section 4.

2 TRENDS IN CALL

Computer-Assisted Language Learning evidently has its history. During the past decades several CALL systems were developed and ample user experience available from teachers as well as from learners. But both the system designers and the users had to cope with restrictions that prevented the successful introduction of CALL into language curricula. The problems will be outlined briefly in the next sections.

2.1 PROBLEMS FOR DESIGN

Early CALL systems used state-of-the-art computational linguistics. Therefore, most of the early programs are either (i) so-called drill-and-practice programs or computer-based training systems with predefined paths of use and static error messages (cf. (Swartz, 1992)), or (ii) dictionary and writing tools (e.g. spelling checkers, word processing facilities, etc.).

These tools can be divided into student-oriented and teacher-oriented tools. The student-oriented ones provide a student with help and/or explanations, while the teacher-oriented tools aim at helping a teacher preparing material to be used in classrooms (with or without computers). The early systems are helpful and should be part of every CALL environment, but the drill-and-practice programs do not comply with all the requirements for CALL one might think of. They often consist of tedious multiple choice exercises that are commonly considered to be only of use in specific parts of a language learning curriculum.

Undoubtedly, this multiple choice drill-and-practice approach and the preference for the development of tools was partly due to the limitations of state-of-the-art natural language processing techniques. These limitations simply did not allow the development of a complete CALL system with all the desired functions. For example, it was (and still is) not possible to develop a parser that covers an entire language and that is robust in the sense that it anticipates all possible incorrect input from users. Additionally, it has been very difficult to give adequate feedback to error, for which it is necessary to know why and how errors are made. Schwind (this volume) refers to this as ‘sensitive parsing’. This topic is also addressed in the papers by Kempen and Dijkstra and in the description of the system PROMISE by John.

Typical is the absence of tools or programs for teaching pronunciation and speech melody. Only a few attempts have been made
to develop learning aids in this area (see the description of the speech melody trainer by Hermes and Spaai). The unsolved technical problems in speech recognition probably explain this poverty. What is available in this area comes down to databases with training material for listen-and-repeat exercises without any feedback from the program. See the paper by Jaspers et al.

2.2 PROBLEMS FOR USERS

The situation for users of CALL systems was perhaps even worse. Initially teachers were reluctant to use CALL systems. After the first obstacle, the lack of adequate equipment, was more or less taken away, a lot of the available programs remained unused. This can partly be explained by the problems with the dissemination of appropriate program information, but the main problem seemed to be that teachers feared to be replaced by computers. Due to this fear, collaboration between researchers/developers of these programs and teacher/users could not get off the ground on large scale. On the other hand, students who had to work with the programs had to cope with very complex user-unfriendly user-interfaces and, as indicated above, with boring, non-motivating exercises. Especially the younger students were often very enthusiastic about the idea of working with computers, but the programs available did not live up to their expectations at all.

Summarizing, the following problems handicap the use of CALL systems:

- improper user-interface (not adaptive, not user-friendly etc.)
- tedious multiple choice drill-and-practice
- lack of adequate language processing methods
- gap between teachers and developers and between curriculum and programs

2.3 NEW DEVELOPMENTS

Recently, AI techniques have been added to CALL applications, resulting in what is often referred to as Intelligent Computer-Assisted Language Learning (ICALL) systems or Intelligent Tutoring Systems (ITSs) (cf. (Wenger, 1987), (Swartz and Yazdani, 1992)). These systems are characterized by a more adaptive and individualized instruction. They consist of:

- an expert module providing the domain knowledge (what to teach);
- a learner module including information about the learners knowledge of what is being taught (whom to teach);
- a tutor module containing strategies and instructional goals (how to teach);
- and a communication module (user-interface).

Most systems that are under development nowadays have this global structure. In addition the following developments and trends can be observed:

- integration of several parts

There is a trend to combine different tools and existing specialised programs into one system. Word processors become more intelligent; together with spelling and grammar checkers, dictionary look-up facilities etc., they can be connected to the training facilities parts of learning systems. The papers by Kempen & Dijkstra and Kronenberg et al. are examples of this development.

- new means for the storage and consultation of linguistic resources

A lot of systems use new means for storage and consultation such as CD-ROM, CD-I, etc. Examples are It's English described in Jaspers et al. (storage of pronunciation information), Discatext (Cartigny), WOORD (Klijn & Jagtman) and de interactive video programs Italcultura, RumboHispano and Ivana (Sanne).
• **use of multimedia**
The availability of so-called multimedia facilities, the integration of texts, speech, video, sound and graphics, brings about new creativity in all areas. For example, exercises and other course material can be made more realistic, more flexible and less predictable. In other words: more attractive and entertaining. (To stress this aspect the word *edutainment* has been introduced.) It is to be expected that some effort will be made to introduce aspects of the style of adventure games in computer-assisted learning systems. The growing market for educational multimedia products is an incentive for developers. This development might help to turn the boring image of the earlier systems into one that generates new motivation both for students and teachers. The use of multimedia is illustrated in many papers, for example the paper by Salverda.

• **focus on the flexibility of interaction and user-interface**
Student-centered CALL systems are often said to be interactive, but what does that mean? Is the user always in control? Or is there a mixed locus of control? Should the reactions by the system be adapted to the performance of the student and to which degree? These days attention is paid attention to such questions. More and more, flexibility in the interaction between system and user is required. The static menu-driven dialogues and multiple choice exercises will not do anymore. These topics are addressed in the papers by Hamburger and Rous & Appelo. The questions are, of course, related to the overall requirement of improvement of the user-interface, of both student- and teacher-oriented systems.

• **growing interest of teachers**
Teachers are getting interested in CALL. They are beginning to see the opportunities offered by CALL programs. As a result, they are getting involved in the development of CALL programs by bringing in their didactic knowledge and experience in course development. Curriculum-embedded programs are under development more frequently. See, for example, the papers by Salverda or Lobbe. Recent initiatives such as the TELL consortium (see the paper by Thompson) try to introduce and integrate computer-based materials throughout the higher education sector. The test system described in Van Bodegom's paper can be used in the intake procedure of language learners.

• **use of networks**
The use of networks brings about new applications for CALL systems. It is now possible to monitor students or let them talk to each other on a distance. Also the exchange of training material and experiences has become very straightforward. See the papers by Lobbe and the description by Stewart.

• **focus on student models**
The ITSs that use student models that are developed in terms of an expert module have become very popular. (Bouwhuis *et al.*, 1994), evaluating a series of NATO workshops on Advanced Education Technology (NATO AET series), discuss serious problems with ITS student models. They argue that student modelling should be conducted in terms of the cognitive model of the student and not in terms of the expert module. Explanations in terms of the expert module (often used in so-called glass-box approaches (e.g. (Rypa, 1992)) seem to be too difficult for students. ITSs are developed from an AI point of view rather than a cognitive point of view. New developments come from connectionist and situated-agents approaches. See the paper by Rous & Appelo.

• **conversational or communicative approach**
The classic drill-and-practice systems focussed on the training of form-in-isolation. A semantic component deal-
ing with the relation between linguistic form and the kind of situation referred to has been lacking. Recently more communicative or conversation-based methods have been introduced (see the papers by Hamburger, Zock, Kronenberg et al. and Jaspers et al.) Often some kind of microworld is developed in which a student can operate using the language to be learned. This new trend is related to the use of multimedia. Together these features can help to increase the students motivation.

- progress in natural language and speech processing

Considerable progress has been made in the area of natural language processing. Several grammar formalisms and more sophisticated methods for language generation developed and implemented independent of the requirements of CALL appear to fit the requirements for reuse in the context of CALL. (See, for example, the papers by Schwind, Zock and Rous & Appelo.) Expectedly this will lead to an increase in the amount of freedom for the user to choose the kind of input language. Speech processing techniques have improved highly. Speech recognition modules have progressed considerably over the past few years. Several application of small and large vocabulary systems are now feasible (e.g. telephone answering machines, automatic inquiry systems, or dictation systems). Speech synthesis is also experiencing fast development. Another increase in flexibility is to be expected from this.

Whether or not these trends will allow for the emergence of a more successful area for CALL is hard to predict at present. But certainly the capacity of CALL to incorporate at least some aspects of the more advanced techniques mentioned above should not be underestimated. In many respects the new technology is conceptually related to the means available in the pencil and paper tradition. Combining pictures, sound and text has always been part of language teaching methods. So the introduction of the potential of multimedia in CALL is not a too big move.

Anyhow, whether or not the results will turn out to be a success, the new trends at least seem to generate new inspiration for teachers and researchers.

In the next section we will go into the relation between CALL and Language Technology.

3 CALL AND LANGUAGE TECHNOLOGY

Why is CALL a suitable topic for a series of workshops on Language Technology such as TWLT? Why can the potential benefit of computational linguistics and artificial intelligence be demonstrated by CALL so very well?

Several issues should be addressed here:

- restricted fragments

As pointed out in the paper by Kronenberg et al., this suitability of CALL is first of all due to the fact that the restriction to specific language fragments and situations is quite natural within the context of language teaching.

Of course, the use of NLP modules with a restricted coverage may introduce new problems. For example, it raises the question whether input that is outside the covered fragment should be treated as erroneous or not. This uncertainty complicates the generation of adequate feedback messages. Furthermore, it is by no means clear whether users will be able to handle the limitations. Neither is it guaranteed that the restrictions can be fully over-seen beforehand. In this respect CALL applications are similar to other applications of natural language processing techniques. Supporting users with an editor which guides them through the restrictions ('controlled language') is one solution. But if the NLP component is not
meant as an analysis tool for user input, but as a generation tool in a multimedia environment, the limits of NLP can be overcome in a more principled way as is shown by Rous & Appelol.

- **role of semantics**
  In addition to fragment size, also the fact that the role of semantics can be kept restricted is of some importance. Of course the trend of conversational and communicative programs increases the importance of interpretational matters. In particular, the mapping of utterances on a domain or micro-world requires more sophisticated grammar formalisms than those needed for programs that deal with sentences in isolation and for which contextual matters fall outside of the scope. Still the amount of reasoning can be kept restricted, at least in comparison to the soundness of the logical component that is required to build a natural language interface for querying a database.

- **importance of language engineering**
  CALL research is often directed more by the requirements of the envisaged use than research in other areas of computational linguistics. Implementation of a system is therefore almost always an important goal for CALL-projects. As a consequence, matters of implementation, or rather language engineering, are more likely to demand for a solution in the context of CALI than in more theory-oriented lines of research. The integration of some model of language description into a working language processing environment requires the solution of a lot of non-trivial engineering matters. In fact, it is hard to come up with language processing techniques that can be implemented without sophisticated tools and design. This holds not only for programs that explore a conversational or communicative approach but also for CALL environments that are focussing on morphology and syntax matters (cf. the special purpose parser in Kempen & Dijkstra).

Adjustment of available NLP methodology is reported in many of the papers in this proceedings. For example, Jaspers et al., Kempen & Dijkstra, and Schwind.

A discussion of the role of CALL within the field of LT may contribute to the awareness that language engineering is not just a matter of implementing a suitable grammar formalism.

- **CALL-systems as spin-off**
  CALL may profit from the products of LT. On the one hand, this is demonstrated in a very straightforward way by CALL systems that use NLP-modules for exercises and drill-and-practice of mere form aspects. Examples of this kind are the systems described by Guvenir & Oflazer, Kempen & Dijkstra and Van de Plassehe et al. On the other hand, reuse of NLP-modules that are developed for other or general purposes appears to be feasible also in CALL environments which are designed to teach language aspects without neglect of context. The contributions by Rous & Appelo, John, and Schwind all report on this possibility.

4 **Concluding Remarks**

User-orientation appears to be the most characteristic feature of the more advanced systems presently under development. As it is the case with the design of usable natural language interfaces to information systems (cf. (Jong and Nijholt, 1993)) this implies that a considerable part of the job will consist of providing adequate models of the agents involved in man machine communication. Another consequence is that CALL is heading towards a stage where linguists can no longer pretend that adequate modelling of linguistic knowledge is the major effort to be made. Mere application of research in computational linguistics is by no means enough to build systems that meet the demands of the user parties. The real challenge for computational linguists is to enable the reuse of
linguistic resources within a flexible interface and to think of ways in which these resources can be put to use without infringing on the communicative needs of the language learners.

REFERENCES


EUROLINGUIST TEST: AN ADAPTIVE TESTING SYSTEM

Marjolein van Bodegom
Eurolinguist Residential Language Institute
Postbus 12
6500 AA Nijmegen
The Netherlands

ABSTRACT

An adaptive language testing system is described. The program, called EuroTest, is intended to facilitate and effectuate the intake procedure of language learners.

The student answers 45 questions, these questions are selected for him based on the initial entrance level and the way the student answers the questions. Answering these 45 questions takes about 20 minutes. The test results in a report that gives the teachers detailed information on what the student does and does not know. The test is user friendly and the results are instantly available to teachers. This article describes the principles behind the system.

This practical need resulted in the development of the EuroTest system. The idea was to develop a system that would give a detailed overview of the student's knowledge of the language. This required a printed form teachers could refer to while teaching the student. Another important aspect was the short time needed to complete this test (it turned out to be 20 - 30 min. maximum).

The EuroTest started out as a system consisting of two modes: a teacher mode and a student mode. While designing and working with the system the importance of adding an evaluation mode to the teacher mode became clear.

THE TEACHER MODE

Before the teacher could start working with the computer program, much hard work had to be done. This work consisted of dividing a given language into a number of meaningful categories and dividing these categories into 5 levels. In this way a matrix was formed. The five levels are the same ones used in Great Britain. The teacher now had to choose/design clear and relevant items. Experience has taught us that the test items need revision during the first few months they are in use. The English test which was the first test we designed was revised 5 times. During these revisions we made use of the evaluation mode. Based on our experience with the English test, we developed the French, German, Spanish and Italian tests. The Italian test is in its third revision stage. The items of the Dutch test are developed by CITO, a well-known Dutch testing bureau.

As an example we will take a look at a matrix consisting of 2 categories:

\[ NCAT \times NLEV = 2 \times 5 = 10 \text{ CELLS} \]
Each of the cells must now be filled with 9 questions in groups of three. The teacher can use several types of questions: Multiple-Choice with 2, 3 or 4 alternatives or open-end questions with up to 10 answer possibilities and a feature that takes care of small typo's.

The questions are given in groups of three because the program adjusts its level of questioning after such a group. If the student gets everything correct s/he moves up a level. If the student makes a mistake s/he stays at the same level and so forth.

<table>
<thead>
<tr>
<th>Level</th>
<th>Category 1</th>
<th>Category 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Q#1 Q#4 Q#7</td>
<td>Q#1 Q#4 Q#7</td>
</tr>
<tr>
<td></td>
<td>Q#2 Q#5 Q#8</td>
<td>Q#2 Q#5 Q#8</td>
</tr>
<tr>
<td></td>
<td>Q#3 Q#6 Q#9</td>
<td>Q#3 Q#6 Q#9</td>
</tr>
<tr>
<td>4</td>
<td>Q#1 Q#4 Q#7</td>
<td>SAME</td>
</tr>
<tr>
<td></td>
<td>Q#2 Q#5 Q#8</td>
<td>SAME</td>
</tr>
<tr>
<td></td>
<td>Q#3 Q#6 Q#9</td>
<td>SAME</td>
</tr>
<tr>
<td>3</td>
<td>Q#1 Q#4 Q#7</td>
<td>SAME</td>
</tr>
<tr>
<td></td>
<td>Q#2 Q#5 Q#8</td>
<td>SAME</td>
</tr>
<tr>
<td></td>
<td>Q#3 Q#6 Q#9</td>
<td>SAME</td>
</tr>
<tr>
<td>2</td>
<td>Q#1 Q#4 Q#7</td>
<td>SAME</td>
</tr>
<tr>
<td></td>
<td>Q#2 Q#5 Q#8</td>
<td>SAME</td>
</tr>
<tr>
<td></td>
<td>Q#3 Q#6 Q#9</td>
<td>SAME</td>
</tr>
<tr>
<td>1</td>
<td>Q#1 Q#4 Q#7</td>
<td>SAME</td>
</tr>
<tr>
<td></td>
<td>Q#2 Q#5 Q#8</td>
<td>SAME</td>
</tr>
<tr>
<td></td>
<td>Q#3 Q#6 Q#9</td>
<td>SAME</td>
</tr>
</tbody>
</table>

After the teacher has filled in the matrix it should look e.g. like this:

The five levels are fixed, but the number of categories can vary. In our language tests we decided on 5 categories.

Once the paperwork is done, the teacher can start creating the EuroTest program: Since teachers are not programmers and shouldn't need to be, the interface of the teacher mode is user friendly. Teachers that have never worked with a computer before can start typing in their questions after 10 minutes of explanation. The only thing they have to do is use the tab key and type in the correct entries in the correct places.

After the teacher is finished with a category s/he can check if the whole category is filled (the program keeps track of this). It is also possible to make a print out to check for typo's etc.

**STUDENT MODE**

The student mode is the part the student sees while taking the test. This is the interface. The interface in my opinion is the most important part of the courseware. If the interface is not clear and simple to use, your beautiful program will not have the wanted effect.

While taking the test the student doesn't see the upgrading and downgrading, this is done in the background. S/he remains concentrated on the questions and answers without any interference. At the end, after answering the 45 questions, the student gets the chance to use the review mode to go back, look at the mistakes made and see the correct answers.
Eurolinguist

English

Intake

Fig 1: The start screen

Question 1

...... you ever ........ to the casino?

☐ A  Was ...... √
☐ B  Were ......
☐ C  Have been
☐ D  Were been

Fig 2: An example of a multiple choice question
The test gives the students their average score and the time it took them to complete the test. It also gives an interpretation of the level. This explanation was added after a lot of questions from the students, such as: "What does level 3.2 mean?" or "What can I do with level 2.5?". The interpretations were given by our teachers based on their experience. We will have to wait for the ILTA to come up with internationally accepted criteria and descriptions for these levels.

The teacher can print the teacher's report the moment a student is finished or at a later time. The teacher's report gives a detailed overview of what the student has done. The report gives a score per each category. The five categorized scores of an English test could look like this:

1. Present perfect, Reported speech + Questions 3.0
2. Past + Past perfect, Conditional 2.5
3. Future, Word order, Tag-questions 3.1
4. Passive, Some/any, Modals 1.8
5. Prepositions, Phrasal verbs 3.8

This report shows that the student needs to work on categories 2 and 4. Category five shows that the student had no problems with prepositions and therefore needs no special attention in that area.

The teacher can see which path the student took through each category. This path gives extra information. The report concludes with a listing per category of all the student's wrong answers. This makes it possible to see at a glance whether the student made the same mistakes repeatedly, or if the student wrote something completely wrong or maybe just made a typo.

In short the report gives the teacher all the information needed to make an individual learning path (study plan) for the student in a private lesson situation or in a class. The report enables the teacher to make study groups as homogeneous as possible.

THE EVALUATION MODE

The program has its own evaluation mechanism. Every item has a counter that registers if the item has been answered correctly or incorrectly. These counters give an indication about the validity of an item after it has been in use for some time.
CONCLUSION

The EuroTest system has become an essential tool we would not want to do without. It saves the teachers and students hours of work. It takes the student only +/- 20 minutes to complete the test and no correction time is required. The students react positively towards the test. They like the ease of use, many questions are multiple choice and only a click of the mouse is required. The students also like to get their results immediately, so that they can review their errors at once.

The teachers are pleased with the reports since they give them immediate insight into the student’s knowledge level. They combine what they learn from the report with what they hear during their first teaching hour or in a preparatory telephone conversation with the student, and can immediately start working on the weak points.

Teachers that use the test outside the institute have used it for forming groups of the same level. They report that the groups they formed based on the EuroTest results are much more homogeneous than the groups they formed in past semesters.

Besides working in an educational setting we have recently placed the test in companies that use the test as a Human Resource Management tool. These companies use the test to assess new employees and to measure the results of language courses taken outside the company.
Discatext is a software-programme developed for French, German and English as an aid in teaching individual pupils how to enlarge their skills in studying texts in these languages. For each language, there are 3 parts consisting of a Presentator and 9 or 10 texts of a certain level.

There is an opening test on words, and at the end of the programme, the pupil has the chance to test whether his or her vocabulary has improved. The texts available on Discatext range from O-level to Polytechnic-level.

Discatext can be used in various ways. It can be used to train pupils for reading-exams, it can be used as a re-teaching programme, and it can be used to train and test idioms, and - not in the least - it is a good start for a classroom discussion, as it can help the pupil to prepare him/herself for such a discussion.

CD-ROM-XA enables the user to hear every text and every help-sentence parallel to the screen-presentation. This audio help is spoken by a native speaker. It is considered to be a big step forward in the support of language-training. Memorizing the material studied is made a lot easier when the user hears and sees the material at the same time. Moreover, the learning process will lead to fewer mistakes as far as the pronunciation of the words concerned. Finally, the learning process is intensified because the user not only sees the material but hears it as well. It is as if the teacher stands next to him/her.

Structure of Discatext

The Discatext-programme does not have fixed study-routes that should be followed by every pupil. It can be compared to a sort of market in which a lot of different goods can be "bought" at various stalls. The pupil is asked to find his/her way, depending on his/her needs and purpose.

There is information on word-, sentence-, and paragraph- and text-level. Moreover there are exercises and tests on words and the final test on the text as a whole. This test is based on the Dutch system of final exams for secondary schools (CITO).

If a pupil wants to acquire a basic understanding of the text, it will be sufficient for him/her to work at word-level only. If a student wants more information about sentence structures he/she can switch to sentence-level. In the same way he/she can concentrate on paragraphs at paragraph-level or on the structure of the text at text-level.

If he/she intends to use the programme for vocabulary-training, he/she will stay at word-level for longer, and use the exercises on words. In fact the programme has been designed to achieve a number of different aims. The programme does not direct the pupil towards a certain target, but the pupil's choice determines the route he/she will follow through the text. This may be done after consulting his/her teacher, or at the specific instructions of the teacher.

NOTE: When working with the programme, the pupil is able to change levels whenever he/she chooses, allowing him/her to find his/her own strategy. This, however, is not possible when doing the final test. This is done to ensure that the pupil can only use his/her acquired understanding of the text.
Contents

Discatext offers 30 texts on 3 levels for each language: from lower intermediate to advanced level. The lower level texts are available now, the other levels will appear in 1994/1995.

<table>
<thead>
<tr>
<th></th>
<th>FRENCH</th>
<th>GERMAN</th>
<th>ENGLISH</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOWER</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTERMEDIATE</td>
<td>1994</td>
<td>1994</td>
<td>1994</td>
</tr>
<tr>
<td>INTERMEDIATE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADVANCED</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Within each level the texts have been arranged in ascending order of complexity, so that the user can expand his/her knowledge and command of the foreign language step by step.

What does Discatext offer?

Discatext is an open programme without any fixed, obligatory routes set out for the student to follow. Rather, the programme may be compared to a market where a large amount of information can be selected according to the user's particular needs. Help is available, 18 types of exercises and 14 types of tests, on 4 language levels.

- words: explanations, exercises and tests (always within a context) for 1200 words;
- sentences: explanation of 450 complicated sentences;
- paragraphs: summaries and exercises for 60 paragraphs;
- text structure: explanation, exercises and tests concerning theme, summary and structural outline of 9 texts.

The user of Discatext is entirely free to choose (possibly guided by his/her teacher) a study route through the material in accordance with his/her current knowledge and method of studying.

Who is Discatext meant for?

The programme has been developed for people speaking English, French, German, Italian, Spanish or Dutch, who have a basic knowledge of English, French or German.
USING A CORPUS FOR TEACHING TURKISH MORPHOLOGY

H. Altay Günemin and Kemal Ofazer
Department of Computer Engineering and Information Science,
Bilkent University, 06533 Ankara
{guvenir,ko}@cs.bilkent.edu.tr

Abstract
This paper reports on the preliminary phase of our ongoing research towards developing an intelligent tutoring environment for Turkish grammar. One of the components of this environment is a corpus search tool which, among other aspects of the language, will be used to present the learner sample sentences along with their morphological analyses. Following a brief introduction to the Turkish language and its morphology, the paper describes the morphological analysis and ambiguity resolution used to construct the corpus used in the search tool. Finally, implementation issues and details involving the user interface of the tool are discussed.

1 INTRODUCTION

Language instruction delivered on computers is not new. Computer-assisted language learning has been used in foreign language instruction for some time with varying success. One of the important characteristics of a successful computer assisted language learning system is its ability to give example sentences about the use of grammatical features of the language [12]. The aim of this research is to design and develop a corpus search tool as a part of an intelligent computer assisted Turkish language tutoring environment, which will focus on issues of Turkish morphology, syntax and semantics. The main purpose of the search tool is to enable the user to see actual usage of words along with the morphological structure of Turkish words. When completed, the CATT environment will comprise several components, e.g., an intelligent drill and practice tutor [3], an online dictionary, and a corpus search tool.

The corpus search tool searches and displays sentences, from a given corpus, that contain words satisfying the features selected by the learner. In order to do this, all the words in the corpus are passed through a morphological analyzer once. Since each word is analyzed independently of the other words in the sentence, its morphological analysis may lead to ambiguities. These ambiguities are resolved using another interactive tool called ztag. The following step is the extraction of all the morphological features of the words into a structure called feature indexes. The corpus search tool, using the tagged corpus and the feature indexes, lists all the sentences containing the words matching the features selected by the learner. The tool also displays the morphological analysis of a word selected by the user to show how that word matched the features set. The block diagram of the corpus search tool is given in Fig. 1.

The next section describes the corpus used in the teaching environment. The third section presents a brief introduction to the morphology of the Turkish language. Morphological analysis of words and the resolution of ambiguities are described in the fourth section. Section 4 presents the organization and the user interface of the corpus search tool. The final section concludes with our plans for future improvements on the tool.

2 CORPUS

For pedagogical issues, the corpus that will be used in a teaching environment needs to be selected with extra care. The sentences should be simple, yet descriptive, concise and instructive. The usage should be accepted by the experts on the language and its grammar. The sentences must not have any ambiguities.

Considering all the issues mentioned above, we decided to use, as the corpus for our tool, the example sentences describing the use of the entries in a dictionary for the Turkish language.
4 MORPHOLOGICAL ANALYSIS AND AMBIGUITY RESOLUTION

Our morphological analysis subsystem is based on the two-level morphological specification of Turkish [7] based on the PC-KIMMO system [1]. This morphological analyzer is fairly comprehensive and is based on a root word lexicon of about 24,000 root words. We can illustrate some of the phonetic phenomena of Turkish by a few examples:

| Lexical: | narz+hA | N(table)+1PS-POSS |
| Surface: | narz00m | narzam |
| Lexical: | ev+êhêz+yA | N(house)+1PL-POSS+DAT |
| Surface: | eviniziz+0e | eviniz |
| Lexical: | ayak+mlN | N(foot)+GEN |
| Surface: | ayaq002am | ayaqin |

Here we see in the first example that a high vowel at the beginning of a suffix (denoted by H representing {i, i, u, i}), drops when the stem it is being affixed to ends with a vowel. In the second example, we see that H's are resolved as i in harmony with the last vowel in the stem ev, and the low, unrounded vowel (denoted by A in the last suffix, representing {a, e}) is resolved as a in harmony with the last vowel in the stem eviniz while the consonant y at the begin-

Turkish is an agglutinative language with word structures formed by productive affixations of derivational and inflectional suffixes to root words. A simple example of a Turkish word formation is:

kesilemedi

which can be broken down into morphemes as follows:

kes +il +eme +di
cut +PASS +NEG-CAP +PAST +3SG
stop

This verb can be translated into English as “it could not be cut/stopped.” For the details of Turkish grammar and word formations rules one can refer to a number of books [6, 13].
are ambiguities of such sort, the agglutinative nature of the language usually helps resolution of such ambiguities due to morphotactical restrictions. On the other hand, this very nature introduces another kind of ambiguity, where a lexical form can be morphologically interpreted in many ways. For example, the word evin, can be broken down as:

```
evin
POS  English
1. N(ev)+2SG-POSS  N (your) house
2. N(ev)+GEN         N of the house
3. N(evin)           N wheat germ
```

If, however, the local context is considered it may be possible to resolve the ambiguity as in:

```
... senin         evin
PN(you)+GEN      N(ev)+2SG-POSS
(your house)
...
... evin         kapsi
N(ev)+GEN        N(doors)+3SG-POSS
( door of        the house)
```

We have developed a POS tagger for Turkish text based on our two-level specification of Turkish morphology augmented with a multi-word and idiomatic construct recognizer, and most importantly a morphological disambiguator based on local neighborhood constraints, heuristics and limited amount of statistical information [8]. The tagger also has additional functionality for statistics compilation and fine tuning of the morphological analyzer, such as logging erroneous morphological parses, commonly used roots, etc. Preliminary results indicate that the tagger can tag about 98% of the texts accurately with very minimal user intervention, by using about 200 usage and contextual constraints, and heuristics.

For morphological processing of our corpus we used our tagging tool tag so that each word in the corpus had one and the correct morphological structure and part-of-speech. Morphological analysis and ambiguity resolution in our experimental corpus, which contains 10984 words in 1600 sentences, takes about 100 minutes of cpu time, as PC-KIMMO is a very slow morphological analyzer (about 2 words/second on Sun SparcStations). The feature extraction on this corpus takes 7 minutes of cpu time.

2 Currently, the speed of the tagger is limited by essentially that of the morphological analyzer, but we have ported the morphological analyzer to the XEROX TWOL system developed by Karstunen and Reesley [6]. This system which can analyze Turkish word forms at about 500 forms/sec on Sun SparcStation.
Figure 4. The corpus searcher.
5 THE CORPUS SEARCH TOOL

As we mentioned earlier, Turkish is an agglutinative language where words are formed by a sequence of morphemes that get attached to a root like “bead on a string.” Hence, one of the more difficult issues in learning the grammar of the Turkish language as a foreign language, is the order of the morphemes in a word. The corpus search tool, called corpus searcher, is designed to help the learner by displaying the order of the morphemes corresponding to a selected set of features. The words are shown as used in a sentence which is selected from the corpus. Illustrating the usage of words helps the learner to visualize the meaning of the whole sentence and the role of the morphemes in the words satisfying the features selected by himself.

The use of the corpus searcher is very simple. The user sets the values of some features of interest, then starts the search. All the sentences which contain a word satisfying the morphological features set by the learner are displayed in a window.3

The words matching the features are shown in bold font. If the learner clicks on any of these sentences, the morphological analysis of the matching word in that sentence is shown in a different window, along with its all morphological features.

As an example, suppose that the learner sets the value of the agreement feature to 3rd singular, the aspect to past, and the voice to passive. Out of 1600 sentences in our current corpus 44 contain a word satisfying these conditions. Suppose further that the learner clicks on the sentence

Musluğun akınına bir türlü kesilemedi.

which means “The leaking of the faucet could not be stopped, despite all efforts.” The display of the corpus searcher would then be as shown in Fig. 4. The word “kesilemedi” is shown in bold since it is the word that satisfies the morphological features that are set.

Currently, the corpus searcher enables the learner to specify any subset of the following set of features: agreement, aspect, case, category, possessive, tense, tense, voice, suffix, and root. The learner is free to set any subset, although the choice of some features imply the values of some others. For example, if the learner sets the value

of case to be dative, the value of the category is implied to be noun. Therefore, any further selection on, for instance, the tense feature would result in the null set of sentences.

The program runs on Sun SparcStations. It is implemented in Lucid Common Lisp, with lispview (Xview) interface.

6 CONCLUSIONS AND FUTURE WORK

We have presented our initial implementation of a corpus search tool for teaching aspects of Turkish morphology. The system enables a learner to set certain morphological features and then searches a corpus of example sentences for words matching the specified features, and if necessary, also displays the morphological structure of the word. This system is planned to be a part of a much larger system, CATT, that is being designed for computer aided tutoring of the Turkish language.

In the near future, we intend to augment the morphological analysis display with an English translation of the matching word. This is in general rather non-trivial as the English translation for a single Turkish word may be a complete sentence. We also plan to enlarge the set of features and group them under a paradigm menu, and incorporate derivational suffices so that the learner may also have a chance to see the semantic changes introduced by derivational suffices. We also plan to have a much better user interaction model whereby the system gives clear messages if the learner sets the features in a conflicting manner that can not be realized in Turkish morphology.

7 ACKNOWLEDGMENT

This work is being done as a part of the TULANGUAGE project that is being supported by NATO Science for Stability program III.

REFERENCES


VIEWPOINT ABSTRACTION: A KEY TO CONVERSATIONAL LEARNING

Henry Hamburger
George Mason University
Fairfax, VA USA

Abstract: Situation viewpoints or views represent the range of what the Fluent-2 system can say about actions, independently of any particular action. A view and an action are inputs to the view processor, the first step in natural language generation. The system supports foreign language learning by two-medium conversation. The two media are natural language and graphics, each connected to a microworld situation built out of everyday objects, classes, actions and plans. The system lets a learner practice unfamiliar aspects of the new language in the same kind of supportive setting - with visual information and situational continuity - that characterizes successful language learning in a simple conversation with an attentive native speaker. A second goal of the system is to provide a testbed for intelligent integration of the two media. The tutorial component selects views and makes other decisions about the discourse and the microworld, to fulfill its responsibility for coherent and educational two-medium conversation.

1 Conversational Language Learning

After introducing the Fluent-2 conversational language learning system and its pedagogical rationale in the first section, we turn to situation viewpoints. This key abstraction is presented in perspective in section 2 and then in some detail in the final section.

1.1 Sketch of the Fluent-2 System

Fluent-2 is both a working system and a work in progress. Taking the user's angle, Figure 1 shows the graphics of a typical microworld, Kitchen World. This scene is consistent with an interaction in which the learner has taken the initiative and picked up the cup, and in which the tutor now describes that action or its result. The scene can also arise from a command by the tutor, after which the tutor might give the next command or comment on the responsiveness of the learner's action.

Figure 1. A Fluent-2 Microworld

Getting to this state took a sequence of coordinated steps by the tutor and/or the student: turning on the water, picking up the cup, taking it to the water, where it filled (with filling shown as an animation), and so on. At this point the tutor can say, for example, "Now pour the water into the pot." Pouring, the ticking of the clock, and the fire on the stove are other animations.
Figure 2 - System Overview

NLU and GU handle input (U for 'understanding') in natural language and graphics. NLG and GG produce output (G for 'generate') in the two media, respectively. D diagnoses language input; SMM is the student modeling module; and VP is the View Processor. For NLG detail, see Figure 6.

The system overview in Figure 2 shows how the major components fit together. The graphics understander (GU in Figure 2) lets the student control the hand, to make it carry out actions in the scene on the computer screen. In the Kitchen World of Figure 1, these include picking up and putting down the cup and the other liftable objects, opening and closing the cabinet doors, and turning the water and the stove, burner on and off. The graphics generator (GG) shows the consequences, including animations, of actions by both the tutor and the student. Correspondingly, the NLU and NLG modules carry out natural language understanding and generation. A single cycle through Figure 2 includes the student saying or doing something, which goes to NLU or GU to the microworld, causing changes in the situation. Student inputs and their consequences then lead to student model updates and tutorial decisions about what to do and say via GG and NLG, updating the microworld en route. All of this is coordinated by the system executive, on the basis of decisions by the tutor or student.

Each microworld is an internal representation of not only the objects visible on the computer screen but also structures for the actions and plans that can be carried out, as well as certain pedagogical information. Despite the deliberate simplicity and ordinarness of the situations, the internal structures are complex, particularly for actions, which lie at the crossroads of the communication process, coordinating the content of messages among the three representational forms: graphics, natural language and the internal representation. Plans, which are highly flexible in ways described later, consist of subplans and actions, like the plan for boiling water, which includes a subplan for filling a pot, and which is itself a subplan of cooking potatoes. Finally, each microworld is linked, flexibly and indirectly, to other microworlds via the language learning subject matter that it provides and requires.

1.2 Pedagogical Justification

In our conversational immersion approach, the learner is continually using the new language, not analyzing it, not consciously computing grammatically correct sentences, and not falling back into an already known language. Supporting arguments for this approach are marshalled in Hamburger (1993), where specific connections are made between research findings and the projected capabilities of the system.

Immersive methods have a long history. For one thing, gradual immersion occurs and works in pure form for the child learning a first language, though that is only suggestive. More directly relevant to adults is the success with immersive foreign language classroom methods. The use of visible objects that these methods recommend is directly adaptable to our partially animated graphics on the computer screen.
To know a natural language is not just to produce grammatical sequences of words, but to produce output that expresses what you mean, and not just to parse or recognize grammatical input, but to understand it. Indeed conscious application of grammatical rules can be counter-productive (cf. McLaughlin et al., 1983). A learner must associate language with conceptual material. It follows that an experience or a system must make it possible for the learner to associate new words and phrases with ideas already in the context or picked up from visual clues. This strategy along with the use of simple subject matter keeps down cognitive load, giving language learning a chance (cf., Sweller, 1988).

2 Views in Perspective

Flexibility is the theme of this introduction to situation viewpoints or views (section 2.1) and their relationship to the rest of the system (2.2), especially actions (2.3). Actions merit this attention since views specify how to act upon them to give a representation of what to say.

2.1 Introduction to Views

The view is crucial to the flexibility of language generation in Fluent-2. Views will make it possible to say dozens of different things in reference to the same action, depending on the situation and the state of the discourse. Figure 3 shows just a few of the possible responses to the action of picking up the full cup in Figure 1. These examples can be generated now, but their linkage to the microworld is not yet in the flexible, declarative form described here. The description of the view in the second column of the figure is not actually a view, which is a formal object; see Figures 5 and 7 below.

<table>
<thead>
<tr>
<th>View</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>student's most recent action</td>
<td>You picked up the cup.</td>
</tr>
<tr>
<td>achieved goal of the action</td>
<td>The cup is in your hand.</td>
</tr>
<tr>
<td>do the next part of the current plan</td>
<td>Now pour the water into the pot.</td>
</tr>
<tr>
<td>compare action to preceding command</td>
<td>I asked you to pick up the pot, not the cup.</td>
</tr>
</tbody>
</table>

Figure 3. Language Output.

Examples of the language variety that views permit for the same action in Figure 1.

A view can call for describing an action the student has already taken, generating a command or suggestion for a possible action, processing an answer to a question, or other kinds of language moves, depending on the interaction type. For example, in Figure 3, the interaction type calls for description in 1-2 and involves a command in 3-4. The language content is action-based, but not slavishly so. That is, it is always related to a particular action, but (i) that action may have already been carried out, or mentioned, or neither; (ii) the action's arguments may be specific objects or class constraints; and (iii) one can discuss - besides the action itself - the states, plans and other actions associated with it. Views support all this flexibility.

The result will be a wide range of plausible natural language, tightly integrated with a rich graphical interaction, in the context of everyday scenes. Fluent-2's twin media, linguistic and graphical, are independent in that each is driven not by the other, but by tutorial decisions and microworld situations. Yet the two are tightly integrated in that they occur in close succession, referring to the same objects and actions in the same ongoing situation and conversation. As a foreign language learning environment, such a system promises a crucial form of foreign language experience that is at a new level of learner involvement. Alternatively, by deploying the learner's own (native) language in the language processing component, the same techniques lead to a capability for discussion and explanation in intelligent tutoring systems, on subject matter other than foreign language.

More generally, views can improve two-way, two-medium communication for a range of computational systems. Their flexibility is necessary across languages because of differences in how cultures prefer to express things (Delin et al., 1993) and because of differences in how languages make it possible to express things (Felshin, 1993). A selection of viewpoints is also needed within
languages, both for coherence (Meteer, forthcoming) and for effective rhetoric (Hovy, 1988a).

2.2 Flexible Conversational Tutoring

Flexibility is essential to our approach, both to mirror the generativity and wide range of natural language and to let the tutor adjust the level of challenge to the apparent ability of the current student. Six sources of this flexibility are listed below. In contrast, a typical foreign language end-of-chapter exercise in a textbook sets into concrete all six of these choice areas. Moreover, only a handful of computer-based language learning systems free up more than one or two of them. The best intelligent tutoring systems on non-language topics are flexible on the equivalent of microworlds, plan rules and tutorial strategy.

- microworlds
- plan rules
- situation viewpoints
- interaction types
- languages
- tutorial strategy

The first four items in this list are, in effect, the tutor’s tools. But Fluent-2 can also function as a learning environment (as opposed to an intelligent tutoring system), in which the learner is allowed, through a graphical interface, to make or influence these decisions. Pedagogically, this capability promotes active involvement and also lets the learner direct the conversation to areas of perceived need. The tutor must be able to take over again when the learner relinquishes control.

Microworlds. For communication in general, microworlds can provide a focusing device and a variety of levels of abstraction. For foreign language tutoring, they provide a variety of situations. A simplified sub-microworld can be a valuable introductory device in either kind of system, in one case to simplify the topic, in the other to restrict the relevant language.

Plans. Plan rules represent the coherent real-world activities that are possible in the microworld. They consist of subplans and ultimately the action rules that govern the primitive actions in the ongoing microworld situation. The flexibility of plans is that

(i) they have variables, so they can apply to any objects of the appropriate classes; (ii) subplans can have a non-rigid order; and (iii) other parts of a plan allow it to be resumed by the tutor in mid-course, even after arbitrary actions by the student.

Interaction Types. Flexibility also arises from the variety of choices of an interaction type, which structures the discourse by distinguishing among command-act, question-answer, act-describe and other move sequences, and also specifying who - tutor or student - does the asking, commanding, etc.

Views. The view lets the tutor talk about an action itself, its results or its role in a plan, as well as permitting many ways to talk about an object. Together, the view and the interaction type provide the basis for the wide variety of language constructions needed in a foreign language learning environment. Sections 2.1 and 3 elaborate.

Language. Another major source of flexibility is of course the multilingual, natural language processing (NLP) software. The NLP system was built by MIT’s Athena Language Learning Project which adapted it for use in Fluent-2. It includes a relatively comprehensive yet efficient parser, with unified, multi-level error handling, as well as a natural language generator, both interacting with a semantic representation that is language-independent in some important ways.

Tutorial Strategy. The tutor must coordinate all these choices in a manner consistent with the integrity of the conversation and the educational needs of the current student. We have defined tutorial schemas to make the tutor’s job manageable without relinquishing the independence and flexibility of the tutor itself. (See Hamburger, in press).

2.3 Representing Actions for Views

This section treats actions and their representation, to provide the basis for understanding views, treated in depth in section 3. Actions operate on microworld objects, and their applicability hinges on the permanent and temporary properties of those objects, as well as their class memberships and relationships among the objects. Objects, actions and plans all are implemented as programming objects and have a structure that supports their triple role for language, graphics and internal representation.
HEADER: (pick-up (Sh hand) (Sobj physical-object)  
       (Sfrom physical-object) (Smw microworld))

GOALS:  ((oav $h grasp $obj))
PRECONDS: ((oav $h grasp nothing) ... )
TRIGGERS: ((clench $h) (visit $h $sobj))
G-RESULTS: ((g-attach $h $sobj $smw))
K-RESULTS: ((modify-oav $h grasp $sobj)  
       (delete-among $from things-on-top $sobj) ... )

Figure 4: Action Rule, Pick-up

The Header includes the action’s name and parameter list. The tutor can use an action if its Preconds are met and its Goals are not. The Triggers indicate how the student activates it graphically. In either case - whether it is the student or the tutor that takes the initiative - the last two slots indicate how to update the graphics and the underlying microworld.

More precisely, both actions and plans are implemented as parametrized rules with constrained variables, as in Figure 4. Applying a rule requires, at least, finding actual arguments of the right class. For example, the first argument of Pick-Up in Figure 4 must be a hand. Action rules are bi-directional: either the student or the tutor can activate them, depending on the type of interaction. The student does this in a graphically realistic manner, for example by dragging a hand to the faucet and clicking the mouse. Since actions are organized into plans, they can be called upon in a coherent and purposeful sequence, when initiative falls to the tutor.

Certain slots in action rules play crucial roles in view processing. The Header slot, which contains the rule name or predicate as well as argument-constraint pairs, is used in a straightforward view that results in a simple description of the action. Equally useful for view processing is the K-Results (K for knowledge, as opposed to graphics) slot, containing object-attribute-value triples for updating the internal situation as a result of the action. State views can select among these results to report various changes. There are slots for preconditions and goals, and these too can underlie views. It is the goal slot that gives plan execution the flexibility, noted earlier, to skip over unnecessary actions and plans, since they are the ones whose goals are already achieved. Plans are like actions, but have a Subplan slot and no K-Results slot. Subplans are associated by and, or and sequence operators, allowing the tutor to make plausible variations in its own selection of actions and also to accept variety in student action sequences. Two additional slots, in action rules only, are for information passed to and from the graphics module and are not used for views.

The arguments of an action rule can appear not only in the header but in other slots as well, and that they have scope over the whole rule, so that when an argument is bound in any slot it is bound throughout the rule. A copy of an action rule with all the arguments bound is called an instantiated action, and is a crucial input, along with a view, to the view processor, as will be seen in the next section. Letting binding originate in any slot allows information to flow in various directions among the student, the tutor and the microworld, thereby helping to support the two-way, two-medium conversation.

3 Anatomy of Views

The view plays the principal role in determining what to say about an action, so in that sense, the tutor’s selection of a view is the crucial first step in natural language generation (NLG). A view is an abstraction of what to say and how to say it, expressed as a structure. It specifies how to operate on an action or a possible action in a situation to produce a language-independent conceptual structure corresponding to a statement, command or question about that action or about its results, purpose, participants or other aspects. The partial example in Figure 5 indicates the overall structure of a view, in particular its three levels: context, event and object. The event level is central in that it corresponds roughly to the proposition expressed in the main (or only) clause of a sentence. The other aspect to notice at this point is the View-Type, within the event level. The view type here is ‘action’, yielding a view that expresses the action itself, without reference to the plan or the resulting state, making this an action view. We will come back to other aspects of this view a bit later.
NAME: current-action

CONTEXT:
When: recent-past ...

EVENT:
View-Type: action
What-Action: student-did ...

OBJECT: ...

Figure 5. Part of a View: Current-Action

A stripped-down version of a straightforward action view, for stating what the student did.

The view processor interprets a view in combination with a particular instantiated action, thereby instantiating the view. The instantiated view (the IV) is a language-independent intermediate representation which is in turn processed further to form the semantic structure of a sentence and ultimately the sentence itself. In this sense an instantiated view corresponds to a sentence in the current language. The declarative approach described here allows in principle for the possibility of using the same view structures for both understanding and generation, though in fact we have worked only on generation.

To see the key role of views, suppose that the student has just made something happen and that the system's role, according to the current interaction type, is to make a relevant comment. The most straightforward way to formulate a comment would be to say what the student did, using a representation of the microworld action that was just carried out, consisting of an operator with its operands. This would result in the likes of sentence #1 in Figure 3. Just such a representation is available in the Header slot (see Figure 4) of the action rule just triggered by the student and it can be transformed to a semantic structure that is an appropriate input for the natural language generation module, which can output the resulting sentence. We do use such a technique, but rather than acting deterministically on only the one action-to-semantics connection just described, we interpose a wide-ranging choice of different approaches to constructing something to say. These are the views, and they play a key role in providing the linguistic variety that is essential to our conversational approach. The straightforward description of a student's action is still possible, but it is just one of a host of possible views; in particular, it is the view in Figure 5.

Figure 6. View Processing in context.

Organization of modules relevant to language generation, showing the central role of view processing. The IV-SS converts the instantiated view (IV) from knowledge representation form to a linguistic semantic structure (SS). (The IAT is the interaction type.)

The central role of view processing is emphasized in Figure 6. After the tutor has selected a view, the view processor uses it with other information to create an intermediate structure, which in turn is passed to later phases of language generation (in Figure 6). We anticipate that properly conceived
views can serve as a language independent specification for organizing action-based conversation. A more generic and powerful version of views than that presented in Figure 5 is possible and desirable. The action just carried out by the student is not the only one that the system has occasion to describe. In view terminology, 'What-action' can be what the tutor did, as opposed to the student, and, moreover, the action need not be one that has actually occurred, but may have only been talked about or just constructed for generation.

These distinctions depend on the interaction type, which we now introduce. Besides playing a key role within the view, interaction types also complement views by organizing the basic conversational move structure. An interaction is a short sequence of specified kinds of linguistic and spatial turns by the tutor and student. Choosing an interaction type determines whether it is the tutor or the student that momentarily takes the initiative. A pedagogically useful interaction type for language learning has at least one linguistic move (is not just graphical actions).

Either the tutor or the student can start an interaction, with one of four move types: action, command, question or statement. Following each with its anticipated response yields the eight simplest interaction types, named so as to reflect the tutor's role. Thus the tutor asks a question in Quizmaster; it gives a command that the student may act on in Commander; and these roles are reversed in Servant. Tourguide is an interaction type with three moves: an action by the tutor, a description of that action, and acknowledgement by the student. Tourguide can provide initial exposure to a new microworld. A variation of it, AnteTourguide (or anticipatory Tourguide), allows the description to preclude the action, with consequences for tense, aspect and temporal adverbs, depending on the language in use. Finally in Movecaster (broadcaster of moves), the student can make any possible move, and the tutor then comments.

In these terms, the Current-Action view can be elaborated to Figure 7, which shows Movecaster associated with Student-Did, the student's action, whereas Commander calls for an action - Tutor-Thought - not yet carried out by anyone. What-Action takes four possible values: Student-Did or Tutor-Did for the most recent action executed by the student or tutor; and Tutor-Did or Tutor-Thought for an action constructed by the tutor as the basis of something already said or about to be said.

NAME: current-action

CONTEXT:
When: (case interaction-type
  ((movecaster tourguide) 'recent-past)
  (antetourguide 'near-future)...) ...

EVENT:
View-Type: action
What-Action: (case interaction-type
  (movecaster 'student-did)
  (tourguide 'tutor-did)
  (antetourguide commander)
  'tutor-thought))

OBJECT: ...

Figure 7. More of Current-Action View

This version of the Current-Action view shows the role of interaction types in selecting the action and the temporal viewpoint. The various cases are expressed in Lisp syntax.

Besides determining the action and time aspects, the interaction type must be able to require its second move to be responsive to the first. Thus some kinds of questions call for a sentence in answer, others a phrase or just "yes" or "no". Similarly, actions are expected to be responsive to commands, and it is reasonable for a tutor to comment about responsiveness to a command, or lack of it. For this purpose, we have defined an object that integrates views and interaction types: the view-constrained interaction type.

1. You picked up the pot.
2. The pot is in your hand.
3. Now fill the pot.
4. I asked you to pick up the cup, not the pot.
5. The water is not (yet) on.
6. What is (still) on the counter?
7. Now that you have the pot, fill it.
8. Get the pot filled.

Figure 8. Varied linguistic responses
The language consequences for some of these formalisms can be seen in terms of Figure 8, which shows eight different ways to respond to the same action, with a variety of views and interaction types. Thus item #6 in the figure is a question, while #3, #4 and #7 show different aspects of a command-act interaction and the others are descriptions. Views differ in what actions they refer to, with #1 referring backward, #3 forward, and #4 referring to two actions, one of which was created earlier in formulating a command that was never performed. Among state views, the most straightforward is to comment on the new value of an object's attribute, as in #2, but it is usually also quite possible to comment on the cessation of the corresponding previous value (Figure 9, caption) or, as in #5, the goal state of the next action in the plan, to be achieved in the near future. Such negative statements would be especially useful with an underlying system able to detect misconceptions. State views are even compatible with commands, as #8 shows. Finally, if a new property value is shared by a similar object, one may say, for example, that now there are three cups on the table or that both cabinet doors are open.

Underlying the command in #7 is a plan view, in this case the notion of transition to the next action in the current plan. Plans can also refer to such things as the completion of a plan or subplan and the transition from one subplan to the next. Plans exist in the microworlds so that the successive actions will make sense, not only those chosen by the tutor to carry out itself, but also those the tutor tells the student to do, as in #3. The resulting situational continuity supports a language beginner by keeping it clear what is being talked about. For a more advanced student, plan views provide their own form of variety, including two-clause sentences like #7, in which the first clause involves a state view, the second an action view and the whole sentence comes from a plan view, the transition from a just completed subplan to the next action whose goal is not already satisfied.

State views need two slots at the event-level that are not in event views; see Figure 9, implemented using a subclass of the view class. Since an action may result in more than one change in the values of object attributes, state views have an Aspect to specify how to select one of the changes. The selection method shown in Figure 9, '(position 1)' simplistically takes the first one in the list of updates (but see below). The Pre-Post slot tells whether to use the updated value or the prior one, as in Figure 9.

| VIEW-TYPE:  | state         |
| WHAT-ACTION: | last-action   |
| ASPECT:      | (position 1)  |
| PRE-POST:    | pre           |

**Figure 9. Part of a State View**

The event level of the state view containing the sentence, "The cup is no longer in your hand"

Arbitrarily taking the first element in the update list to comment on makes sense only if results are in order of importance. More principled criteria are preferences among semantic roles of the arguments, among the attributes whose values are changed or among those values. For example, changes in the patient, agent, instrument or destination could be placed in priority order. On attributes, one might prefer reports on changes in (relative) location to reports on changes in content. Values could be used indirectly, say by preferring to comment on movable rather than stationary objects, though movability itself is not the topic since it does not change. Which of these methods of choice is to be used might depend on the topic of conversation; for example, one might wish to have a discussion of the sizes of things or of moving things.

The specification of these other ways to select a fact to comment on appears in the Aspect of the event level. Entries may take the form (<factor>) and (<factor> <value>), for example the name of an attribute, possibly accompanied by a preferred value. The earlier suggestion about discussing the sizes of things is indicated by the list (size), and talk that emphasizes moving things is achieved by (movable true). Other permissible examples of this form are (agent), (object1), etc., from which the middle level of generation extracts fine-grained semantic roles in context, using the lexicon of the current language. Putting the simple list (goal) as the Aspect means that the triple comes from the Goal slot of the current instantiated action, and is therefore typically the most important update. This enables us to construct an eminently sensible view that lets the tutor point out that the goal of one subplan has been fulfilled, so it's time to start on the next subplan.
Plan views not only can contribute to the variety needed for language learning, but seem especially suited to the structuring of instructions and explanations in applications other than language learning. Plan views should express (i) descent from a plan to its initial subplan, (ii) transition from one subplan to the next, within the same plan, (iii) the subplans that have been completed so far, (iv) those that remain to be done and (v) a violation of plan-based expectations. Other plan views are straightforward variations of action views. One of these would be used upon completion of a plan, to yield an IV expressing that the plan has been completely carried out. A similar view is appropriate just before embarking on a plan.

Whereas a view tells where to get information, the instantiated view (IV) holds the information itself, which the view processor has for the most part extracted from the instantiated action. For an action view, this is principally the arguments, taken from the action header and placed in appropriate IV slots. Under the guidance of the object level of the view, the view processor associates each argument with the correct slot and puts in the contents. Designed for this purpose is the IV-O, or object level of an IV. Each IV slot can be filled by (i) an IV-O, (ii) a microworld object, (iii) a class, which is a language-independent meaning corresponding to a common noun, (iv) a list of items of the three foregoing kinds, or (v) another IV. The latter yields a subordinate clause, whereas each of the others underlies a noun phrase.

Object-level views determine how to express a particular microworld object (or set of related objects) to convey its relationship to other aspects of a conversation. With a black and a grey cup, for example, after moving the black one, the grey one can be referred to as "the other one," "the grey cup," "the second cup" or even "the cup that is still on the table." In each noun phrase the head noun corresponds by default to the class of the object, unless "one" is included in the specification for that object (giving, in English, the likes of "the red one"). The decision whether to include modifiers (adjectives, relative clauses, and prepositional phrases) may in some cases be expressed by code that includes a method that selects whatever properties are needed to distinguish an entity from others of its class.

The object level may also have information that affects decisions about determiners and possibly quantifiers or pronouns. The choice of determiner can not be specified in isolation by the view, since it must take into account the recent mentions of, and actions on, an entity, for example, "Pick up a (indefinite) box" and then, "Good! You picked it (definite pronoun) up."

<table>
<thead>
<tr>
<th>Subslot</th>
<th>Possible Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRECISION</td>
<td>Top, Parent, Direct, Distinct</td>
</tr>
<tr>
<td>REFERENCE</td>
<td>Other, Pronoun, Demonstrative, Nil</td>
</tr>
<tr>
<td>PROMINENCE</td>
<td>High, Wh, Nil</td>
</tr>
</tbody>
</table>

Figure 10. Possible values at Object level

Object-level subslots and their permitted values appear in Figure 10. First comes the degree of Precision with which the object is to be described. It can indicate whether the class for describing the object should be its Direct class (e.g., girl, teaspoon), its Parent class (e.g., child, spoon) or the highest class (Top) permitted by the type constraint for the particular argument of the action rule (e.g., person, thing). Another option is Distinct, the highest level distinguishing the item from everything else in the current situation. If the item is not alone in the class named, the output needs a modifier or else an indefinite determiner.

If the Reference subslot in a view has the value Other, the item is to be described in terms of other items in its class, e.g., "the other X" or "the rest of the Xs", as opposed to the default case, a description of an object by its own properties. The value Pronoun is only a preference, subject to prior reference and other conditions. The graphical operations of presentation and highlighting of objects also affect the computation of reference. The Prominence subslot specifies whether its object should be made prominent (High) or not (Nil). In English prominence can be achieved by topicalization or clefting. The Wh value indicates questioning of that position.
Implementation: The implementation is entirely in MCL2 Common Lisp with CLOS (the object system), and is running on a Quadra-800 with 20MB of main memory. Inferencing in microworld situations relies on Norvig's Prolog-in-Lisp, which we have integrated to CLOS by making it efficiently find attribute values to use as logical facts. The ProMotion package has helped make animations relatively easy to create.

Acknowledgement: The work reported here has been supported by grant IRI-9020711 from the U.S. National Science Foundation.

References:


LEARNING ENGLISH WITH IT’S ENGLISH

J.G.M. Jaspers, G. Kanselaar, W.A.M. Kok
Department of Educational Sciences, Utrecht University
Mailing address: Prof. dr. Gellof Kanselaar
Department of Educational Sciences
Heidelbergaan 2, 3584 CS Utrecht, The Netherlands
Tel: 31.30.534893, Fax: 31.30.532352
Email: kanselaar@fsw.ru.nl

ABSTRACT:
This paper describes a new Computer Assisted Instruction program for foreign language (English) teaching based on a communicative approach.

Due to the greater mobility and the closer ties in the European community, foreign languages have become an even more important subject in the school curriculum. Based on the results of a study in 1989 [3], a new Computer Assisted Instruction program for foreign language teaching (English) has been developed. Main features of this program are: the communicative approach, a 70,000 word dictionary, multimedia and a syntactic parser.

BACKGROUND IN THE DEVELOPMENT OF IT’S ENGLISH

To a large extent foreign language teaching is directed towards communicative goals. Socio-cultural developments (mass media, holidays abroad etc.), scientific developments in teaching methodology, the use of new media, linguistics and psycholinguistics have led to a more functional approach to foreign language teaching.

The traditional grammar-translation method, in which vocabulary was learned mostly by memorizing L2/L1 language pairs and in which grammar was taught as a system of rules, did not directly lead to fluent oral proficiency.

In developing the CAI programme we chose the communicative approach to foreign language teaching. In this approach language is viewed as an instrument of human behaviour. The behavioral “situation” is carefully selected in teaching. The linguistic context, the situation, speakers’ roles and types of texts are all taken into careful consideration in the communicative approach. Linguistic function is seen as more important than linguistic form. The structuring of the learning materials is first and foremost determined by communicative needs.

When designing the learning materials for the CAI programme we used Neuner’s typology as a starting point for the construction of exercises. The three types of exercises in the programme, including receptive, reproductive (cloze texts) and productive (free sentence input) exercises can be placed on this continuum.

BASIC AIMS AND ASSUMPTIONS

The aim of the project is to create a computer-assisted learning environment for communicative foreign language teaching [1]. This form of foreign language teaching is oriented towards an active and passive command of a foreign language in everyday situations. It avoids translation as the only or the most important means of conveying meaning. Vocabulary is acquired on the basis of context-specific sentences and meanings in the foreign language, to ensure the development of a language system that is separate from the learner’s native language. In addition to the syntactic and morphological characteristics of words, their pronunciation is of great importance, as is sentence intonation. The sound of a new word is a basic element in assigning meaning to that word. In order to effect this a great number of words and sentences were stored on CD-ROM. The aim of communicative language teaching is mainly achieved by having students use the non-native language productively. In order to achieve random language-input a sentence parser was developed for the program. This makes use of the grammatical information contained in the CD-ROM dictionary.
The combination of dictionary, phonetic word-representations and parser makes it possible to create different receptive and productive exercises in this interactive learning environment.

The software program IT'S ENGLISH is primarily aimed at teaching vocabulary. However, as an intelligent tutorial system are working on making it usable in the teaching of writing. In the following sections the program itself will be described.

A short history of It's English

During a previous study [3] a program called "PRESTIGE" written in SD-PROLOG and running on MSDOS was created. This program allowed, among other things, students to enter their own text. This limited dictionary (± 2000 words) proved too small. Student often used words not contained in this small dictionary.

To overcome the problem of the limited dictionary we used a 70,000 word dictionary that was made available to us. The size of this dictionary however and the addition of sound capabilities to the program made it inevitable to use another storage medium. At the time the CDROM-XA standard was the most adequate choice. This resulted in a CDROM-XA disk which held both the dictionary databases and a great number of soundfiles. This version of the program was tested in four groups of students in secondary education [7].

The MSDOS version of the program had two major drawbacks: the limited amount of memory available to the program and the limited capabilities of the 80*25 character screen. In order to remove memory limitations and improve on the user-interface a new version of the program is being developed. The new version runs under Microsoft-Windows version 3.1 in enhanced mode. This version was implemented in Prolog-2 for Windows, with some extensions in TurboPascal for windows. This transition removed the memory problem and improved the user-interface.

The CDROM-XA standard for sound however is not supported by the windows multimedia environment. Conversion of the soundfiles to the standard multimedia format of Windows (.WAV-files) is currently in progress.

DESCRIPTION OF THE 'IT'S ENGLISH' PROGRAM.

A computer program that connects with the above basic aims and assumptions should enable the user to retrieve semantic definitions and context sentences in English, as well as synonyms, antonyms, pronunciation and the syntactic and morphological properties of words. In addition such a program should be capable of evaluating the user's use of language to enable him or her to apply the vocabulary in (re)productive exercises. To achieve all this the program consists of several components:

1. An automatic sentence parser which checks random language input for syntactic, orthographic or inflectional etc. correctness.

2. A CD-ROM database, which contains definitions and context sentences in everyday English, as well as the grammatical properties and synonyms and antonyms of over 70,000 words.

3. A CD-ROM database with the digitized pronunciation of over 5000 words, 1200 context sentences for the presentation of phonetic word representations and English intonation.

4. An exercise generator, which generates exercises for passive, active and reproductive language use.

5. A feedback-generator, which uses the domain-input of the parser to reveal and remedy learner errors in (re)productive exercises.

The Collins Cobuild English Language Dictionary [2] was chosen as a basis for the lexical CD-ROM database. This dictionary meets the following requirements. It was compiled for 'non-native learners of English'. Each word is accompanied by a semantic definition in simple English, and a number of authentic, explanatory citations. In addition, each word meaning is accompanied by systematic and moderately detailed grammatical information, for use by the parser. This direct connection between grammatical information and the semantic section makes the meaning of every word that is recognized by the parser instantly available for retrieval, as well as a number of context sentences and other relevant information such as synonyms, antonyms and superordinate
terms. One inherent restriction in the application is that the parser’s “power”, that is, the number of English structures that can be automatically parsed, is limited by the grammatical information available in COBUILD. An important advantage of using such an extensive dictionary is the availability of information on nearly all English words. In previous studies [3] we found that in productive exercises with limited dictionaries (2000 words) pupils often use words that are not in the selected subset of English.

CD-ROM AS A STORAGE MEDIUM.

Until recently it had not been possible in computer-aided education to devote much attention to foreign-language pronunciation, in view of the large amounts of external memory involved. To a large extent the arrival of CD-ROM has removed this restriction. In this project CD-ROM is used as a storage medium both for digitized pronunciation and the lexical database based on Collins Cobuild Dictionary. The choice of the Extended Architecture (XA) standard for the CD-ROM in our project is exclusively related to the audio-component.

The most important considerations in the choice of the XA-standard for the audio-component were the following:

1. The sound-quality had to be optimal. In particular the absence in speech of the higher sound-frequencies (3500 - 11000 Hz) makes it very difficult to distinguish the pronunciations of certain words in English, whereas more or less subtle sound differences in fact signal differences in meaning. This is especially important in speech from female speakers, because their speech is naturally richer in higher frequencies.

2. In the acquisition of a good foreign-language pronunciation it is of importance for learners that the phonetic representations of words should be presented not only in isolation but also in sentential contexts. The learner should experience the pronunciation of words as being connected with and influenced by their contextual position as well as by the intonation of the sentence.

PROGRAM EXERCISES.

The exercise generator.

The generator is capable of generating different types of exercises on the basis of texts stored in the database. Exercises are been developed for the program which range from reading exercises to free writing. These exercises follow a route from receptive via reproductive to productive language use, and may be completed in a teacher- or learner-determined sequence. Three types of exercises have been developed so far in the making of the program. These are prototypical of the type of application that is possible with such programs. In developing these exercises we have followed Neuner [4], who distinguishes three main phases in an exercise sequence that leads to communicative language use. As a result of the newly-developed parser, adding exercises in more elaborate versions of the program is a relatively simple matter.
A. Reading texts (receptive use).

A fragment of a story is selected by the student or teacher from the text database. Learners are free to retrieve a definition of each text-word in English, or the context sentence that accompanies each word. As soon as a text has appeared on the screen, the user can click on any word on the screen to search for the word in the lexical database. Definitions and contexts are retrieved and presented. If the selected word occurs in the sound-database that sound-fragment will be presented to the student. The first meaning paragraph from the dictionary will be shown. Appropriate synonyms, antonyms and superordinate terms will be displayed. The user can select the type(s) of information by enabling options in the box in the upper left corner of the screen. The NEXT and PREVIOUS buttons enable the user to browse through the available information (see figure 1). It will also be possible for the teacher to mark a word in advance to indicate that the learner should call up information about that word later. In this process word-specific feedback can be provided by the teacher.

B. Texts with blanks (reproductive use)

In the creation of texts with blanks one or more properties of a word, all of which are recorded in the parser's property-list, may serve as a criterion for the omission of that word from the exercise, or for its inclusion in it. For example, texts could be created without any prepositions or adjectives. The decision to leave out all nouns, verbs, prepositions or adjectives can be taken by the teacher or the learner. Sentence constituents such as verb phrases can also be left out. This type of filling-in exercise is clearly superior to the cloze-texts of current prevalence, in which only every n-th word may be omitted. Since the database can be put into operation in the analysis of the learners' answers, feedback can be provided which is aimed at various kinds of errors. This makes IT'S ENGLISH different from traditional programs, in which every erroneous answer, plus the accompanying feedback,
Figure 2: A gapfilling exercise

has to be anticipated and dealt with. The type of feedback that is provided by IT'S ENGLISH for filling-in exercises can be of the following kind: the typed-in answer is synonymous with a superordinate concept; the word class is correct but the actual word is not; information about possible 'typing errors' which is based on an algorithm; information about conjugation and other grammatical characteristics. It is also possible to provide help by giving a context sentence provided by the dictionary from which the target word has also been omitted (see figure 2)

C. Random text-input (productive use).

This type of exercise allows the learner to type in a story. The parser checks each sentence for grammatical correctness. Parsing of natural language is quite a complicated process. The current version of the parser knows over 400 rules to connect words to form grammatical sentence-constituents. The current version of the parser is able to analyse some quite complicated sentences. Most parsers are build around a relatively small dictionary where the information is tailored to the needs of the parser. Our parser is completely dependent on the information from the Cobuild dictionary. One consequence is the programs inability to obtain gender information from the dictionary.

For a parser to be able to pinpoint an error in a sentence we will have to adapt the parsing mechanism to generate a hypothesis for possible errors. This will require extensive modifications to the parsing mechanism and research into the most common errors of students. When an error occurs a feedback message will be generated and the learner is given an opportunity to correct the error. The final story is stored in the text database, and can thus serve as a basis for new exercises. This enables learners, among other things, to create fill-in exercises for each other. All the help options based on the dictionary that are available for the receptive exercises are similarly available for this type of productive exercise. The structure of a sentence can be made visible in tree-form on the screen. This possibility of random language input has one drawback: it makes great demands on the speed and capacity of the computer. The dictionary generates all possible grammatical uses of each word in a sentence. This gives rise to a combinatorial explosion of possible ambiguities. This slows down the parsing process. We are looking into ways of optimizing the parsing process to cope with this problem. However, in our opinion the program is uniquely valuable at present precisely because of its
capacity for processing random input, with graphical feedback about the grammaticality of input-sentences (see figure 3).

In the teaching of writing various aspects of writing practice can be dealt with, such as planning and organization, formulation and revision. The program offers help in formulation at sentence level. A particularly powerful aid is the ability to call up context sentences for a particular word in order to examine its use in frequently occuring contexts.

**Learner registration.**

The program stores information concerning the words that each learner has studied up to a given moment, the help that has been requested and the answers to the exercises that have been given. These data can be scrutinized by the teacher.

**Exercise development.**

The system applications allow learners as well as teachers to enter random texts for the development of exercises. For example, a teacher could in principle enter a text for later use by the learner to develop exercises. There are two entry-checks. The first one tests whether the word with the desired grammatical properties is present in the computer lexicon. The second check tests whether the parser is capable of parsing the input-sentence and whether the sentence is grammatical. Both checks are required for the creation of exercises.

Stories can also be entered after the parser has been cancelled. In such cases the only type of operation to be carried out is the connection of each word to its first dictionary meaning. The advantages of this option are that the input-process is faster and that texts of any degree of difficulty can be entered. This kind of text can only be used for receptive exercises.

The generative potential of the parser and the great amount of information in the lexical database make it possible for the program to adjust quickly to different teaching methods by allowing for the addition of method-specific texts. The menu-driven environment and the extensive context-sensitive help-options make the program very user-friendly for both teachers and learners. The unambiguous screen layout makes it relatively simple to enter...
commands. It should be borne in mind, however, that the program’s full potential can only be realized with suitable machines (i.e. computers with 80486 processors, 50 Mhz, 8 Mb RAM, hard disk, Windows 3.1 and a soundcard).

**Multimedia**

The Windows environment supports a wide variety of ways to present information. Text and graphics are the most common types used in applications.

With the addition of a soundcard sound can be easily recorded and played back. This way listening comprehension tests could be created.

A relatively new addition is the ability to display digital video in a windows on the screen. The performance in frame-rate and the size of the playback-windows is currently quite limited. Advances in image compression will soon overcome these problems. Use of digital video in language teaching opens other powerful ways of creating interesting exercises.

**References**


TOWARD AN INTEGRATED SYSTEM FOR GRAMMAR, SPELLING AND WRITING INSTRUCTION

Gerard Kempen & Alice Dijkstra
Experimental and Theoretical Psychology Unit, University of Leiden
P.O. Box 9555, 2300 RB Leiden, The Netherlands
tel. +31 (71) 273834, {kempen, dijkstra}@rufts.LeidenUniv.nl

ABSTRACT
Orthographic accuracy is an important goal of writing instruction in primary and secondary education. If the language has a rich morphology, this goal presupposes a great deal of grammatical insight on the part of the student. However, the transfer of training from grammar instruction to writing and spelling is often very low, causing poor mastery of syntax related aspects of orthography. In this paper we argue for a written language curriculum that is strongly intertwined with initial grammar instruction. We outline the design of an integrated system of software modules that supports a unified grammar, writing and spelling curriculum. The design is largely based on existing modules, developed within our group, for processing (parsing and generating) Dutch words and sentences, and for teaching and exercising grammatical concepts.

1. SYNTAX SENSITIVE ASPECTS OF DUTCH ORTHOGRAPHY

One of the central goals of writing instruction in Dutch and Flemish primary and secondary education is orthographic accuracy. This is due to a notoriously problematic aspect of Dutch spelling. Like in French, and in contrast with English and German, the spelling rules of Dutch systematically produce homophonic but heterographic inflections. For example, the verb verkleeden (Eng. to dress up) has three conjugation forms that sound the same but are spelled differently: in addition to verkleeden which serves as infinitive and present tense plural form, there are the past tense singular verkleede, the past tense plural verkleedden, and the inflected past participle verklede (I disregard some further functions of these forms). Another example is the spelling of third person present tense singular verbs ending in the phoneme /t/. Went, wend and wendt are pronounced identically but went is derived from the infinitive wennen (to get used to) whereas wend and wendt are forms of wenden (to turn). Wend serves as imperative, as first person singular and, in case of inversion (i.e., subject following the finite verb), as second person singular. Wendi is third person singular, and second person singular in clauses without inversion (subject preceding the finite verb). Problematic cases like these are not restricted to rare verbs; e.g., the /t/-spelling rules force writers to choose between word and words — highly frequent forms of the passive auxiliary and copula verb worden (to be, to become).1

In practice one often tries to solve such spelling problems by invoking ad hoc heuristics such as the following: "If you want to know how to spell word(t) in Ik word (I am), Je wordt (You are), Word ik (Am I) and Word je(Are you), then try a verb like werken (to work) where you do hear the inflectional ending /t/. Since you do not hear a /t/ in Ik werk (I work), you write Ik word without a -t ending. Similarly, one writes Je wordt and Word je in analogy with Je werkt and Werk je." Such heuristics are indeed helpful but do not always yield the correct solution. For instance, je is ambiguous between personal and possessive pronoun (you versus your). So, the writer has to realise that in Word(t) je moeder geholpen? (Is your mother being helped?) je belongs to je moeder, and that Werk tje moeder rather than Werk je is the correct analogy. Of course, teachers have come up with a trick to determine whether je is a personal or possessive pronoun: "If one can change je to jij without making the sentence ungrammatical, it is a personal pronoun; it is a possessive pronoun if you can change it to jouw." However, this heuristic is not foolproof either. Although in Word je eigen baas! (Be your own boss!) one can substitute jouw for je, this does not imply wordt as the correct spelling: word is used here as imperative, and je eigen baas is predicate rather than subject NP.

---

1 Actually, by applying grapheme-to-phoneme conversion to over 250,000 inflected forms originating from Van Dale’s Dictionary of Modern Dutch (1991), we have found about 2150 sets of two or more homophonically but heterographic words. The largest set contains 6 members: weid, weidt, wijd, wijd, wijdt and wijt. The number of differently spelled words in these sets amounts to about 4400.
2. INITIAL GRAMMAR INSTRUCTION

Other examples of similar import are easy to find (e.g., see Kempen, 1993). They inevitably lead to the conclusion that in order to spell homophonic but heterographic inflection forms correctly one needs to perform at least a partial syntactic analysis of the sentence. This in turn calls for a prominent position of grammar instruction in primary and secondary school curricula. However, in spite of all the attention that is given to syntactic analysis of sentences in Dutch schools, the results of this training are very poor (e.g. see Prevaes 1994). To make things even worse, whatever parsing skills are acquired during these lessons hardly transfer to writing skills (Schaurs 1990). The causes for this regrettable situation are legion and eliminating them is anything but easy. Factors interfering with successful grammar instruction certainly include the following:

(1) the low level of motivation in students (and not seldom in teachers) due to the perception of grammar as a subject of little practical utility
(2) the poor quality of traditional paper-and-pencil parsing exercises (insufficient time for practising, late and fragmented feedback, inflexible and uninteresting sentence materials), and
(3) the lack of a transparent representation of syntactic structure.


3. A UNIFIED APPROACH TO TEACHING WRITING AND PARSING SKILLS

To sum up, writing instruction is hampered by the lack of grammatical insights in students and by insufficient transfer of such insights to writing tasks, and grammar instruction seems to lack compelling arguments in support of its immediate practical usefulness.

This formulation of the two problems reveals that they could be important assets to each other's solution. Teaching writing and parsing skills as one integrated subject matter will enhance the transfer of grammatical insights to writing on the one hand, and provides a tangible argument for the practical value of grammar on the other. In what follows we will outline a unified approach to writing and grammar instruction based on various software tools created by members of our research group. As is generally recognised, present-day microcomputers offer excellent opportunities for skill acquisition. We will put emphasis on training in applying syntax sensitive rules of Dutch orthography and on parsing exercises.

4. CONTROLLED WRITING AND PARSING CONTROLLED TEXTS

It would be utterly unrealistic to propose a CALL system consisting of the following components:

(1) a full-fledged 'grammar checker' that accurately diagnoses the grammatical quality of texts about arbitrary topics typed into a word-processor file by, for example, fifth- or sixth-graders
(2) an 'intelligent tutoring system' that, based on the checker's diagnoses, infers lacunae in the students' grammatical knowledge ('student modelling') as well as customised remedial grammar curricula, and
(3) a varied collection of instruction modules and exercises for inclusion in the curricula.

In view of prevailing hardware and software limitations we have set ourselves a modest target which we believe could still make for considerable improvement.

First and foremost, the writing tasks to be performed by the students will not be unconstrained creative writing assignments. In order to keep error diagnosis at an acceptable level of accuracy, we will use 'controlled writing' tasks, e.g. of the 'sentence combining' variety. Paper-and-pencil versions of this task that have been applied successfully in the United States (Dauker, Kerek & Morenberg 1985, 1986). The basic idea is as follows. Students are presented with a sequence of short clauses that each express an atomic proposition. Together the propositions make up a little story or article. By transforming the short clauses and combining them into longer sentences, the students should produce a coherent and fluent piece of text. A simpler alternative — but even farther removed from unconstrained creative writing — would be to display on the screen a varied selection of words (citation forms) whose conjugation or declension in sentential contexts involves difficulties of the kind discussed in Section 1. The students are then instructed to make up arbitrary but grammatical sentences using the given word materials.

Secondly, we do not propose to have such 'controlled texts' analyzed by a grammar checker. Instead we will use a stripped and adjusted version of the syntax sensitive spelling checker for Dutch that was designed and implemented by Vosse (1992, 1993, forthcoming). The central component of this system is a Generalized LR(1) shift-reduce parser operating on the basis of an augmented context-free grammar. It was specifically designed for the detection and cor-
rection of syntax dependent spelling errors as described in Section 1. These errors usually surface as mismatches between morphosyntactic features within a constituent. For instance, the misspelling *ik wordt* can be analyzed as missing agreement between subject (first person) and finite verb (second or third person). Vosse created a robust feature unification mechanism that, rather than giving up in case of unification failure, attempts to pin-point the missing agreement(s) and to find the 'cheapest' way of restoring correctness. The implied changes to one or more words of the input sentence are then displayed as a suggested correction. For details see Vosse's forthcoming dissertation.

A PC version of the parser with the original vocabulary of over 250,000 entries and over 500 grammar rules has been tested extensively on various types of texts. Its performance in terms of accuracy of agreement error detection and of speed is certainly unsatisfactory for unconstrained texts. But we estimate that a stripped version with a smaller vocabulary and a simplified grammar tuned to the sentence materials in the controlled writing exercises will be able to reach an acceptable level of performance.

The third restriction we wish to impose on the system has been inspired by our experiences with an earlier 'educational word-processor' (Vosse 1989). Parser performance degrades drastically when punctuation marks are missing or incorrect so that accurate segmentation of the text into sentences is impossible. Since punctuation by fifth- and sixth-graders typically leaves much to be desired, it will be necessary to run a text segmentation routine, e.g. the one in Vosse's system, before activating the parser. Also, we need a user interface module that helps students to polish up punctuation before sending the text to the parser.

5. TRAINING SYNTAX SENSITIVE SPELLING RULES

In the framework of a CALL program it would obviously make no sense to have the system repair all misspellings automatically. Instead, misspellings and their diagnosis by the parser should activate a teaching module capable of

(1) explaining the nature of the error in terms of pertinent morphosyntactic rules
(2) offering opportunities for training the rules that apparently are beyond the student's mastery, and
(3) eliciting from the student a correction of the misspelling, and checking its quality.

During each of these steps further gaps in the student's knowledge or skill may come to light, e.g. lack of understanding the grammatical concepts used in error explanations, or insufficient rule application skills. Since each such gap may cause activation of another module, potentially elaborate hierarchies of calls to explanation and teaching modules may come about. In order to avoid the risk of students getting lost in a forest of low-level details and forgetting about the top-level writing task, we will need to devise a more flexible control structure. We leave this didactic problem as an important system design issue and, in the rest of this paper, concentrate on the nature of the teaching and exercise modules instead.

6. EXPLAINING ERRORS AND GRAMMAR RULES

The system must be able to handle two different cases. A spelling error may lead to a non-existent word, or to an existing word that causes the sentence to be syntactically ungrammatical, e.g. because of an agreement error. In the first case the student may need help to determine the correct spelling of the intended word (cf. Section 8). In the second case the student may first need guidance in locating the misspelled word that causes the agreement error. For example, the parser may inform the student that an agreement error has been found between the subject and the finite verb. To understand the meaning, and — more importantly — to know the course of action to be taken to remedy this error, the student should be able to obtain more specific information about the type of error, and information pertaining to the grammar rule(s) involved.

A hypertext-like database system, as was used in a system supporting Dutch writers of English (Dijkstra & Sijtsma 1992; De Ruijter & Dijkstra 1993), can help the student find the appropriate information. In order for the student not to get lost in hyperspace, the grammatical information in this database should be very well organized. The Dutch curriculum for initial grammar instruction developed by Jongen-Janner, Reimann-Pijls & Kempen (1991) seems ideally suited for this purpose. It introduces and explains virtually the complete grammatical terminology needed to describe the grammatical structure of non-compounded sentences (almost 100 terms). The operational definitions of these terms adhere to the format of a genetic graph; that is, every new concept is defined in terms of already known concepts. The 'dependency links' thus created are easily converted to hypertext links.

An error diagnosis produced by the parser can direct the student to the appropriate information in the hypertext database. If the parser is incapable of determining the exact nature of the error, the student himself may be asked to build a representation of the
syntactic structure of the intended sentence. By mouse manipulation he could indicate the words and constituents in the sentence and associate with them the names of parts of speech, word groups and grammatical functions. (Large parts of the user interface of TAALTRIS, the computer assisted grammar exercises described below in Section 7, are suited to this purpose.) On the basis of this structure the parser will then be able to determine the appropriate error diagnosis.

Once the student has located and understood the error he has made, he should be urged to correct it. If he does not know how, a spelling tutor can be invoked to help him out (cf. Section 8).

In order to provide the student with a better insight into the grammatical organization of sentences, their structure can be displayed two-dimensionally, e.g. in the form of trees, of directed acyclic graphs (DAGs), or of 'recursive temples' designed by Dienesveld & Kempen (1993). To this purpose we have developed various graphical tools (Boe & Hensgens 1989).

7. ATTRACTIVE COMPUTER ASSISTED GRAMMAR EXERCISES

Grammar instruction cannot restrict itself to conveying grammatical insights to the students. An equally important goal is skill acquisition: students should reach a sufficiently high level of proficiency in applying the operational definitions of the grammatical concepts, e.g. in identifying parts-of-speech (POS, i.e. word categories) and syntactic constituents of various types, in recognizing their morpho-syntactic properties, and in constructing utterances that instantiate such concepts. Any grammar curriculum should therefore provide extensive and attractive materials to exercise these skills.

In an attempt to maximize the students' extrinsic motivation to engage in grammar exercises, we have designed an adaptation of the well-know computer game of TETRIS (Dienesveld & Kempen 1993; we baptized it TAALTRIS: Du. raad = Eng. language).

The computer screen displays a box with a row of bricks on the bottom. A word is printed on every brick; the words together make up a sentence. The top of the screen shows another set of bricks with part-of-speech names of printed on them. The game's objective is to pair up the words with their word classes by directing the POS bricks to the word bricks as fast as possible. When the student does not succeed in completing a sentence in the time span allotted, a new sentence appears on top of the previous one. The student then proceeds by assigning parts-of-speech to the words of this new sentence. If he succeeds in time, the second sentence disappears and the student may now complete POS assignment of the earlier sentence. In case of bad performance, the box will gradually get filled up with word bricks, causing the student to lose. The game, implemented by Peter Dienesveld, contains various other gadgets and trimmings similar to those used in popular computer games. Currently we work with two versions: in addition to the one for POS assignment, we have another one where the word bricks have been replaced by word group bricks, and the POS bricks by bricks with names of grammatical functions printed on them (subject, direct object, finite verb, adverbial modifier, etc.). The goal of this version is to acquire skill in identifying the grammatical functions of major phrases at clause level.

In an ongoing evaluation experiment with 60 sixth-graders in two primary schools we have observed that students find the game highly attractive. More importantly, their proficiency in POS assignment and in identifying grammatical functions seems to improve markedly.

The sentence materials selected for the TAALTRIS grammar game were picked from children's books or made up by teachers. To increase the involvement and motivation of the students, they can be encouraged to turn the sentences they generated during controlled writing exercises (Section 4), into exercise material for fellow students. For this purpose, some of the TAALTRIS components will be re-usable. However, the parts of speech and grammatical function assignments by the students must now be checked by the parser.

8. A SPELLING TUTOR FOR DUTCH

To determine the inflectionally and orthographically correct form of intended words, a spelling tutor will be included. For the category of Dutch verbs, such a tutor has already been developed: HET SPELRAAM (Kempen & Janner 1990). It operates on the basis of morpho-syntactic knowledge built into the program and is equipped with error diagnostic tools based on that knowledge (see Kempen 1992 and Bos 1994 for details).

HET SPELRAAM includes a spelling algorithm consisting of two stages. The first, so-called morpho-syntactic stage generates a formula specifying the prefix and/or suffix to be attached to the verb's stem for the particular conjugation. An example of such a formula is (ge+)stem+dt for past participles, equivalent to the English stem+ed formula. The substring (ge+) indicates an optional prefix ge, while dt indicates a choice between the homophonic endings -d and -t. These choices have to be made in the second, so-called orthographic stage (see below).
In order to find the appropriate formula, the student traverses a decision tree by answering a series of questions concerning morphological features of the desired verb form and the syntactic structure of the sentence. Some examples are the following: Is the required verb form finite or non-finite? Is the tense present or past? Does the sentence contain subject-verb inversion? Is the number of the subject singular or plural? When the student answers these questions correctly, he arrives at the appropriate formula. If not, he should be given appropriate feedback and, if necessary, be guided towards the relevant information in the grammar database described in Section 6.

In the second stage of the spelling algorithm, the formula obtained in the first stage must be applied to the verb's citation form (the infinitive). This involves a maximum of three steps, each requiring some string manipulations according to morphological and/or phonological rules involved. First, the verb's stem must be determined. Second, the prefix and/or suffix(es) specified in the formula must be added to the verb's stem. And some additional string manipulations must be performed in order to conform to certain phonological principles, e.g. changing the stem's final consonant v into f. A third and final step is only needed in case of inflected participles, which call for an additional -e ending and possibly some further alterations to the end result of the second step. In this stage too, HET SPELRAAM checks the actions performed by the student and provides appropriate feedback and help if necessary.

For the system proposed in this paper, HET SPELRAAM should be extended to include algorithms for inflecting and spelling adjectives and nouns. Moreover, the new version should communicate with the parser so as to be able to check the student's responses during the morpho-syntactic stage.

9. SYSTEM ARCHITECTURE, STUDENT INTERFACE, AND STUDENT MODELING

The software components we have proposed in the previous Sections should all be embedded and integrated in a larger system. The central module will be a simple text processing system from which the various components are called. Furthermore, the system will have to provide for transfer of information between the modules.

A relatively simple but very useful student modeling tool may be based on automatic registration of errors made by individual students or by homogeneous groups of students. Such information will help the parser and spelling tutor to find correct error diagnoses, and will guide the system in selecting proper grammatical feedback and remedial grammar exercises.

* * *

In this paper we have emphasized the role that various individual modules can play in the proposed computer assisted system for written language and grammar instruction. Integrating these modules and adapting the existing versions to their new environment will not be an easy task. Nevertheless we are convinced that building the system will be rewarding in view of the expected increase in learning speed and teaching efficiency.

ACKNOWLEDGEMENT

We are indebted to Theo Vosse, Elena Jongen-Janner and Peter Diesveld for their invaluable contributions to the design and implementation of current versions of the modules described in this paper.

REFERENCES


CONTEXTUAL VOCABULARY LEARNING WITH CAVOL

Friedrich Kronenberg, Anja Krüger, Petra Ludewig

Universität Osnabrück
Institut für Semantische Informationsverarbeitung
D 49069 Osnabrück, Germany
email: ludewig@dosuni1.rz.uni-osnabrueck.de

ABSTRACT

Computer Assisted Language Learning (CALL) is an ideal application to demonstrate the potential benefit of Computational Linguistics (CL) and Artificial Intelligence (AI). The restriction to well-defined language fragments and situations is an essential principle of language teaching and, thus, can be - at least to a large extent - justified by a claim inherent in the application itself. The problems nevertheless arising with respect to this realization of CL&AI concepts may give valuable hints for future basic research.

The CAVOL system for Computer Assisted VOCabulary Learning illustrates that CALL systems based on CL&AI technology are superior to conventional programmes for language learning: On the one hand classical exercises (e.g. multiple choice) can be optimized with regard to their flexibility and economy. On the other hand new exercise types allowing extensive learner input (e.g. dialogic exercises) become possible. Furthermore, the encoded linguistic and world knowledge (e.g. the lexicon) can be reused within different contexts (e.g. lexically supported editor).

1 INTRODUCTION

To exaggerate, one might say that classical programmes for Computer Assisted Instruction (CAI) have the wide scope of functions books can offer along with optimized cross references and fast feedback (cf. Puppe 1992). Their deficiencies originate primarily from the fact that they do not have at their disposal a model of the knowledge to be taught and, as a result, cannot adequately treat learner input which has not been explicitly anticipated before. But a complete anticipation of correct answers is not always easy and the anticipation of incorrect answers usually even knows no bounds. For this reason multiple choice and similar exercises enjoy great popularity (for further details see Gust et al. 1994). This state of affairs comes into serious conflict with the generativity of natural languages and the resulting educational aim of linguistic creativity.

Against this background many language teachers assess computers as anti-communicative and, consequently, irrelevant to language learning. Thus, not only do they forget that conventional language learning also comprises elements of stupid swotting. But they also disregard the fact that adventure games (Wiegand 1991) and simulations may have stimulating effects on the communicative behaviour of students. With respect to its potential capacity the computer is the interactive medium per se, even though, unfortunately, most actual adventure games and simulations do not have sufficient linguistic knowledge at their disposal yet.

Having access to a model of the knowledge to be taught and the student to be instructed as well as didactic principles Intelligent Tutoring Systems (ITS) are able to behave more flexible than conventional educational programmes. But still, the vast majority of the knowledge based CALL systems have experimental character and focus on the aspect of intelligent, i.e. knowledge based, treatment of linguistic mistakes (cf. Krüger-Thielmann 1992). Actually, the transition from experimental to commercial knowledge based systems should be less difficult for CALL than for any other language application, e.g. computational translation, because in contrast to other language applications language teaching itself postulates a deliberate restriction to special lexical and grammatical fragments and well-defined situations. For this reason CALL is the ideal application to demonstrate the workability of CL&AI concepts. But, to guarantee the acceptance of these systems a considerable cooperation between computer linguists on the one hand and language teachers on the other hand will be needed.1

1 At this point we would like to thank all members of the study group MGU (MedienGestützter SprachUnterricht) supported by the Lower Saxony Ministry of Education for their endless willingness to discuss the conception of our CAVOL system.
With regard to the components fundamental to Intelligent Tutoring Systems CAVOL focuses on the model of the knowledge to be taught. Didactic principles and learner modelling are realized only in a very rudimentary form. Furthermore, CAVOL concentrates on lexical aspects of foreign language learning, because vocabulary learning has wrongly got a bad name in current language didactics (Arendt, 1992). Vocabulary learning should be rehabilitated by redefining and extending the term of vocabulary learning. One step in this direction is taken by contextualizing vocabulary learning. The mastery of words of the target language comprises efficient access structures (lexical associations) and their correct use in situational contexts.

Thus, CAVOL is already in itself an interesting application system. But, since every complex programme for language learning needs a lexical component CAVOL should also be integrable into complex CALL systems. For this reason the architecture of CAVOL is designed as modular as possible (section 2).

The efficiency of CAVOL is based on a highly structured lexicon (section 3). Contents based lexicon access especially profits from the multimedia representation of lexical semantics. The coordination of graphics and formal situation descriptions (Gust & Krüger, 1994) permits to exploit graphics to an extent that is distinctly above the average.

To demonstrate the benefit of CL&AI technology exemplary exercises were implemented (section 4). A multiple choice exercise on semantic relations illustrates to which extent conventional exercise types gain flexibility and lose developmental expenditure. The exercise for vocabulary learning in situations uses classificational knowledge and benefits from a sortal hierarchy. A graphically supported dialogic exercise on prepositions outlines perspectives for new exercise types. Furthermore, a lexically supported editor shows that once the linguistic knowledge has been encoded for an exercise it also can also be reused within contexts beyond language learning in its strict sense. This writing mode can also be understood as a further exercise.

2 ARCHITECTURE

In this section we shortly describe the overall architecture of the highly modular CAVOL system (Figure 1).

Since parts of CAVOL are integrated into a CALL system named PROMISE (cf. John, this volume) the CAVOL system can exploit PROMISE's components. By using the PROMISE parser and its dialogue handling component dialogic exercises (like "Completing a Sketch") with "free" learner input (according to specific situation) become possible.

All exercises are controlled by a didactics component (section 4) which guides the learner through a sequence of exercises and controls the exercise organization.

A main focus of the CAVOL architecture is the organization of the lexicon (section 3). The lexical information is distributed over several lexicon modules (according to teaching units).

Via "Read Access" and "Write Access" the user can ask for lexical information or add and change lexical entries. A filter (section 3) makes sure that any information which is of purely technical interest or which would directly reveal the task's solution is hidden when accessing the lexicon within an exercise.

Both, the CAVOL (for the complete program documentation see Gust et al., 1993) and the PROMISE system, are implemented in a PROLOG derivative named G_LOG (cf. Gust, 1992) which has been developed by Helmar Gust at the University of Osnabrueck. Compared to standard prolog the window-oriented G_LOG development system offers object- and graphic-oriented extensions.
3 THE LEXICON

CAVOL heavily relies on its lexicon. In every exercise the learner can ask for lexical information. With the help of a special interface for lexical maintenance teachers, experienced learners or system developers can extend the lexicon or adapt existing entries for specific demands or situations. Since all exercises are generated on the basis of the lexical data, extended or changed lexicons will lead to new or adapted exercises. The lexical databases can also be accessed in the CAVOL editor to assist the learner in writing foreign language texts.

In this section we will describe all possible ways to access the CAVOL lexicon.

STRUCTURE OF THE LEXICON

The lexical information is distributed over several lexicon modules. Each module stands for one unit. The complete vocabulary to be acquired by the learner becomes available when all lexicon modules are loaded. This distributed lexicon approach was chosen to clearly distinguish the vocabulary that should be learned in one unit.

In some contexts, e.g. working with the CAVOL editor, where the relevant vocabulary is not easily identifiable, the complete lexical database is loaded.

Since all lexical information is organized according to units, the data for one lexeme can be distributed among different units. Thus, specific aspects of an entry can be easily attached to specific units. Working in a unit the vocabulary of all preceding units is still accessible while information attached to subsequent units is not usable in the current exercise (but can still be looked up in the lexicon).

For the learner a lexical entry appears as a grouping of attribute/value pairs. In Figure 2 we present the entry for the Italian lexeme lingua. If required, attribute/value pairs can be attached to specific readings of a lexeme: in our example both sense entries of the Italian lexeme lingua have a separate definition feature. Information, e.g. wortart (engl.: part of speech) and genus (engl.: gender), that does not refer to specific readings applies to the complete entry.

Figure 2 shows the entry for lingua as it can be seen by the learner. Internally the entries are represented in a "distributed" structure, i.e. there is a separate clause in the lexical knowledge base for every attribute/value pair of a lexeme. The entry in Figure 2 is built up by collecting and converting all features into the output format.
READ ACCESS

The learner has access to CAVOL's lexical information from all components of the system. Entries can be looked up by using several search strategies. Electronic dictionaries may offer more ways to access vocabulary than printed dictionaries do. In CAVOL a lexical query can consist of

- a single lexeme
- one or more attribute/value pairs
- (if two or more languages are activated) a lexeme or attribute/value pair query in one chosen language
- a mouse click on an object in a graphically represented situation

Up to three matches are directly shown in the window named "Beispiel" (engl.: example). If a query leads to more than three matches, e.g. when looking up certain attribute/value pairs, the corresponding lexemes are listed in the window "Wortliste" (engl.: word list). In this window the student can select the lexemes that he/she wants to see in detail.

As an example for graphic- and situation-based access to lexemes we take a look at the English lexeme *pedestrian crossing*. The student, who does not know this lexeme, can load a situation in which the object "pedestrian crossing" occurs. By clicking on the corresponding object in the picture the lexeme(s) (here: *pedestrian crossing* and *zebra crossing*) belonging to this object are shown with all their lexical information.

The learner gets access to an appropriate situation by looking up a known word which can be found in the same situational surroundings as the word looked for. An example: A student who is looking for the lexeme *pedestrian crossing* or a traffic scene and who knows that the word *traffic light* appears in the same situational surroundings can look up the entry for *traffic light*. In this entry he/she finds a pointer to the traffic situation (in German: VERKEHR). By clicking on this pointer a picture with a typical traffic situation appears (Figure 3). In this picture the denoted object (in this case the "traffic light") is marked by inversion. The student can now click on other depicted objects, like the pedestrian crossing, and thus access lexical entries.
LEXICAL FILTER

When the learner accesses the lexicon within an exercise, we have to make sure that any information which would directly reveal the solution is not displayed. This function is realized by a filter mechanism with defaults for hiding all technical information which is exclusively needed for the internal language processing (e.g. sort) and which should only be visible when working in lexical maintenance mode. The filter can be adapted individually for every exercise: In the exercise "Sense related words" this filter hides all information concerning synonyms, hypernyms etc. because the learner should learn words interconnected by these relations and not answer a question by copying the information from the lexicon.

WRITE ACCESS

CAVOL is designed as a partially extendable system, i.e. the user may write and change lexicon entries when working in lexicon maintenance mode. This mode is primarily addressed to system developers and teachers.

The following operations can be performed on lexical data:
- add new entries,
- add information to existing entries,
- change information in existing entries,
- delete entries.

By working in lexicon maintenance mode the teacher who uses CAVOL in teaching can adjust existing units for his specific demands and add new units.

CAVOL controls the names of features which are used when writing new or changing existing entries, so that typing errors in feature names are prevented. In a separate window named "Bausteine" (engl.: constituents) the user can select from a group of relevant attributes. In addition to these sample features the user can freely introduce new names in the lexicon. When an entry is integrated into the lexicon all feature names which are unknown to the system are reported (Figure 4). The user is questioned whether a "new" feature name should be introduced in the lexicon or not (e.g. because of a typing error in the attribute name which the user can correct immediately).
4 EXERCISES

Up to now, four exemplary exercises have been implemented to demonstrate the feasibility and the advantages of knowledge-based systems for vocabulary learning. A "didactical" component takes care of maximal standardization of exercise surfaces and courses. It offers predicates for

- handling the tutorial feedback,
- grading students,
- managing the items to be tested, and
- administrating the leaflet which will be shown at the end of an exercise.

Prompting comments and the "mirror" reflecting the actual grading of the student (cf. Figure 6), try to exploit hints of experienced language teachers and parents according to which motivation preserving elements are imperative to practical CALL systems.

4.1 SENSE RELATED WORDS

The classical multiple-choice exercise which can be found in many commercial teaching systems today is based on explicitly anticipated questions and answers. In contrast CAVOL generates tasks dynamically on the basis of the highly structured lexicon. Thus, by extending the lexicon there are automatically more words available to generate new tasks.

In the exercise 'Sense Related Words' a student is asked to identify the word which stands in the required relation to a given item and which is "hidden" in a list of alternatives. Every alternative is, in some way, sense related to the task word. Possible relations are hyperonymy, hyponymy, taxonomy and synonymy one of which is chosen at random at the beginning of each task generation.

In Figure 5 the task word is 'car' and the chosen semantic relation to be tested 'hyperonymy' (germ. 'Oberbegriff'). The right word to be selected would be 'vehicle' which is randomly mixed with other words which are in some way sense related to 'car'.

The tutorial feedback is in so far intelligent as it uses lexical information about semantic relations to evaluate the student's answers and to generate the feedback. On the basis of this information an error explanation can be given, cf. Figure 6.

---

2Comments and faces are adopted from a vocabulary trainer Helmar Gust implemented for his sons.
As usual, the lexicon can also be used as a tool to look up those words of the exercise which are unknown to the student. Within this exercise the information about hyperonyms etc. are not displayed, because they would reveal the solution.

4.2 WORDS IN SITUATIONS

In this exercise words are learned within the framework of a situation. The situational context is illustrated in a graphic representation in which particular objects are marked by inversion (cf. Figure 7). The student has to name the inverted objects.

---

3We would like to thank Andre Menzel and Veit Reuer for having been so kind to draw the traffic scenes.
The depicted objects are related to the lexicon via object types (sorts). Each lexical item which has as a semantic description the corresponding sort (e.g. 'bike' and 'bicycle' point to the object type 'FAHRRAD') represents a possible correct answer.

If the student gives as an answer a genus term of the inverted object (e.g. 'vehicle' instead of 'bike') this can be classified as an almost correct answer by having recourse to the sortal hierarchy. In this case and if there are any alternatives to the given expression the correct solutions are named in the tutorial feedback of the system, cf. Figure 7.

4.3 COMPLETING AN ACCIDENT SKETCH

The dialogical exercise on prepositions is embedded in an exercise sequence that deals with a traffic situation (cf. John, in this volume). First the successive phases of the accident are shown. Then, as a witness, the student has to report the accident by telephone to a police station. Now the policeman arrives and tries to draw a sketch of the situation (cf. Figure 8). The student has to describe the position of the objects and persons so that the policeman can complete the sketch showing the site of the accident.

At the beginning, a half-completed sketch is displayed on the screen. First the student is asked to describe his position in the sketch. When the student indicates the position of objects and persons using spatial prepositions the system interprets his input by inserting the symbol of the named object into the sketch (bird's eye view; cf. Figure 9).

Thus, not only the use of prepositions is trained in this exercise, but also the ability to describe a situation from another perspective.

The spatial component calculates the coordinates of the object to be inserted on the basis of the system's knowledge of the modelled situation. Situational knowledge comprises information about the size and prototypical place of the reference objects and the objects which are to be localized. If a local description is too vague or ambiguous the system (policeman) asks another question (cf. Figure 9).

This makes it clear that the preposition exercise exceeds the narrow compound of traditional vocabulary learning. We are dealing now with a dialogical exercise which does not require only lexical and taxonomical knowledge as the preceding exercises did, but also parsing, dialogue control and world modelling, i.e. further essential components of complex CL & AI systems. These components are to be supplied by PROMISE (cf. John, this volume).

---

4It is planned to implement a procedure for detecting lexical interference errors (e.g. 'left from' instead of 'on the left of', cf. Figure 9.
4.4 LEXICALLY SUPPORTED EDITOR

If the student wants to write a text in the target language he or she may use the lexically supported editor of the CAVOL system which has access to the vocabulary to be trained with CAVOL. Thus, the transition from CALL via computer assisted writing to computer assisted bureau communication is fluid.

This writing mode can also be understood as a further exercise. Here, we comply with the educational requirement not only to teach knowledge, but also to impart techniques to gain access to knowledge (Fickert 1992): With CAVOL flexible and even content based access strategies for searching lexical knowledge (cf. section 3) - as computers may support them - can be trained.

The CAVOL-Editor was implemented to demonstrate that the lexical knowledge of the CAVOL system can be re-used in different applications from those it was initially designed for, e.g. to help a user write foreign language texts.
5 OUTLOOK

The performance of CAVOL will improve considerably if the meaning of verbs, adjectives and abstract nouns can also be represented graphically, if necessary with the help of rudimentary animation. Furthermore, the CAVOL lexicon will be extended by an interface to an extensive background lexicon (cf. Hötker et al. forthcoming) which will improve the treatment of words unknown to the core lexicon (cf. Gust & Ludewig 1989, Ludewig 1991, 1993), the individual lexicon extensions carried out by the system user as well as the lexical support of the editor. For example, when using the lexically supported editor, the student shall be able to look up dictionary entries and check word forms orthographically by marking single word forms. Extended search strategies will be used to find the stems corresponding to inflected word forms and to handle lexemes consisting of more than one word.

CAVOL is one of the core elements of the PROMISE system (cf. John, this volume). In this respect the PROMISE system can be understood as

- a measure to evaluate CAVOL with respect to its integrability,
- a tool-box supplying components - e.g. lexical access tools, parser, grammar, and dialogue interpreter - for expanding vocabulary exercises, as well as
- a frame for sketching the place of vocabulary learning within a complex learning process.

6 REFERENCES


Teachers, Students and IT:
How to get teachers to integrate IT into the (language) curriculum

Sylvia Lobbe
Rotterdam Polytechnic Informatica Centrum
s.a.lobbe@ic.hro.nl

Introduction

In 1989 we knew what we wanted. In the first Informatica Beleidsplan of the Rotterdam Polytechnic our aims were clearly stated:

We were going to buy a certain number of computers, set up local networks managed by local computer centres; we would have one central computer centre to develop, manage and promote network services for the whole Polytechnic; we were going to buy software both for administrative purposes and for teaching; a number of projects would get us started in the field of computer-aided teaching.

Information technology (IT) in 1992

Four years later the situation is as follows. Each faculty has one or more local networks and a computer centre. Administrative personnel have their own computers if they need them for their work. Other personnel have hardly any computers at their disposal; most computers are in computer labs. Teachers with some expertise in IT teach general computer courses, mainly in-service training, and tell students how to use IT in methodology courses. Management does not feel committed to stimulating the use of IT in teaching: there are enough machines and there is software, teachers should do the rest. But the organization as a whole is not equipped to support teachers and students.

From 1993 onwards

In 1993 a new organizational structure is introduced: each faculty forms a small number of units called 'clusters' out of the large number of departments. A cluster has one head, its own budget for training teachers and is responsible for its own personnel. At the Faculty of Teacher Training a plan is made to formalize the use of IT. In an appendix there is a description of an overall course I have made for the integration of IT into teaching. In principle this can be the basis for training any of the clusters. Per cluster a selection will be made and the contents will be geared to the teachers taking part. The course consists of five mini courses around the following themes:

1. IT in general: types of programs, evaluating software, network facilities, cd-rom.
2. IT in the classroom: (im)possibilities, learning styles, role of the student vs the role of the teacher, checking the student's progress.
3. IT in the educational faculty: the role of the computer centre, management, copyrights, network vs stand-alone, applying for software, experiences elsewhere.
4. Hands-on course: integrating CAL/IT in the curriculum
5. Hands-on course: use of CAL/IT in the classroom

From these five general course descriptions an introductory programme is selected by a group of more advanced teachers, one per subject. Per subject at least two teachers take the course. The introductory course takes eight sessions of three to four hours. The hands-on courses consist of plenary sessions and individual training.

There are prerequisites. Teachers must have access to the local network and the network services of the Polytechnic. The local computer centre should do its utmost to ensure that programs are on the network and actually work properly.

This plan is never officially adopted by faculty management. Still a pilot project is set up.

Cluster 1

The first cluster to engage in this project is the cluster of history, social studies, economic studies and geography. This cluster is selected by the chair person of the faculty, because her own subject, geography, would be included. The cluster is in the making and eventually only one period of 8 weeks can be used for this training.
course, one session of 3 hours per week. The topics chosen are:

- network facilities (e-mail and the navigating service Gopher), CALL programs & evaluation, CD-rom, the role of faculty management and the computer center, computer-assisted learning elsewhere.

Because it takes a long time for the project to start, I only participate in the initial stage of the preparations and teach just the classes on network services. During the course it becomes clear that the teachers will not get access to the network in their offices. The chair person does not show up for any of the sessions. Enthusiasm withers away. Although the participants are fairly satisfied with the course, they are thoroughly unhappy with faculty management. No follow-up activities are set up.

**Cluster 2**

At the end of the academic year I go to the next cluster: the teachers of French, German and English. Their management team is positive from the moment I suggest IT training. Immediately arrangements are made. Time for meetings and sessions is timetabled. The procedure that was tried out with the first cluster is followed again. A team of relatively advanced teachers selects items from the five general courses and gears them towards the languages. These items are:

- CALL programs, network services, applications (WordPerfect, Transword), concordances, presentation programs, CD-rom, CALL elsewhere. Follow-up activities will be formulated as we go along.

Before the course can start, certain services and programs must be available on the local network. It takes weeks to get the computer centre to cooperate. Each week the programs we want to use have to be checked till the moment the session starts. In some cases we use disks instead.

Three teachers of French, 1 teacher of German and four teachers of English take part in the course. After a hesitant start they become more and more confident. They find that they are able to use e-mail and navigation services with little effort. The Times CD-rom convinces them that IT can and should be used in language teaching. In other words: their attitude has changed. A visit to another Polytechnic shows them CALL programs are used there and how this is organised. The team is eager to start using IT in their own teaching.

The cluster manager is told that a number of CD-roms have to be bought (the Oxford Dictionary and Le Robert Electronique), the Word Perfect language modules and some CALL programs developed at the Polytechnic we visited. On top of that we need one network computer in each office.

Another block of eight weeks goes by. The Oxford CD-rom has just arrived (it is a Windows version, we have not got Windows), Le Robert is on its way. The manager has finally heard that we will get the network computers before the summer holidays. The CALL programs have been ordered. We have started to make a few basic exercises in CALIS\(^1\). We are still determined to use IT in our courses in the next academic year.

**Conclusion**

Another cluster is waiting. This time it is the group of teacher trainers for primary education. This cluster is eager but only has a few stand-alone computers. There is no local network and no link with the network of the Polytechnic.

Ten years ago few teachers were interested in IT. Today word processing, using CD-roms and network facilities like e-mail have become common. Teachers are discovering the potential of IT. It is up to the school, the managers in particular, to recognize this development and create an environment that stimulates it. This fall a seminar will be organised to show faculty directors and cluster managers what IT has to offer, what it takes to organize the use of IT in teaching properly and what IT could mean for the Rotterdam Polytechnic.

\(^{1}\) The Informatica Beleidsplan is the document describing the plans of the Polytechnic in the field of information technology.

\(^{2}\) The Rotterdam Polytechnic has four faculties: the Faculty of Teacher Training, the Faculty of Fine Arts and Architecture, the Faculty of Health and Human Services and the Faculty of Technology.

\(^{3}\) CALIS, an authoring system for languages.
APPEAL: Interactive Language Learning in a Multimedia Environment

Joep Rous
Institute for Perception Research,
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
rous@prl.philips.nl

Lisette Appelo
Institute for Perception Research,
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
appelo@prl.philips.nl

Abstract

In this paper the APPEAL system is introduced. This system provides a multimedia environment for learning. Although we will discuss only a version of the system where the learning domain is language, it can be used for other domains as well. The most important characteristics of the system are the multi-agent approach to teacher modelling and the natural language processing method that is used for the on-line generation of exercises. The integration of both approaches in one system guarantees adaptiveness, flexibility and robustness. The degree to which the system adapts itself to the student and the way it manages the dialogue with the student are emerging properties of the interaction between the agents of the system and the student. The adaptive power is further increased through the utilisation of a natural language generation component which enables the system to generate tailor-made exercises on demand both with respect to type and content. All exercises are presented in a multimedia context.

1 Introduction

A major problem with traditional computer aided instruction (CAI) systems was their inability to adapt to the performance and behaviour of the student. To overcome this problem ‘Intelligent Tutoring Systems’ (ITS)(Wenger (1987)) equipped with a learner module were developed. In most of these systems the learner module includes a model which contains the information that specifies what is being learned. In general, this model is specified in terms of the knowledge representation used in the expert module. The expert module is the central component in an ITS and it provides the domain intelligence of the system.

However, enhancing tutoring systems with a learner module did not bring about the quality of adaptiveness that is needed in learning environments. Masthoff and Van Hoe (1994) analyse why current ITS’s will fail to solve the adaptiveness problem. Firstly, they argue that it is not obvious that the representation of the expert module corresponds to the cognitive representations of the student. Empirical evidence shows that knowledge structures do not provide the most useful models for teaching (McArthur et al. (1988), Roschelle (1990)). Secondly, most ITS’s don’t take into account the learning needs of the individual student. Jamison and Chapelle (1988) report that the effectiveness of a CAI system should be considered in the light of the type of student using the system. The locus of control is an example of an important aspect which should be tuned to the needs of the individual student. A lot of ITS’s suffer from either a very rigid plan-based tutoring strategy where the system has complete control over the interactions or, in so-called discovery-based systems, from the total absence of such control. In both cases the capabilities and preferences of the individual student are ignored. Other examples are the type and contents of exercises that are presented. Depending on their cognitive style students might respond better to explorative exercises than to multiple-choice exer-
cises. One rarely encounters ITS’s, however, that are able to offer different exercise types depending on the student’s performance.

There is a trend in ITS research to enhance the system with multimedia material (Buford 1994). An important reason for doing so is that with the aid of multimedia, learning material can be presented in a realistic context which might have a positive effect on the learning process. Additionally, there might be individual differences between preferred sensory channels for learning. If a system is capable of exploiting different modalities it will, in theory, be able to tune the usage of these modalities to the preferences of the student. In current systems, however, one rarely encounters such subtle use of multimedia. This observation is in line with the above discussion on adaptiveness and individualisation of ITSs.

In the APPEAL (A Pleasant Personal Environment for Adaptive Learning) project at the Institute for Perception Research (IPO)\(^1\), we try to develop methods, strategies and/or tools that can be used in ITS’s to increase their degree of adaptiveness and their quality of interaction. We also use the system as a platform for experiments with multimedia, e.g. video, speech, graphics, non-speech audio etc. The first results of this research have been implemented in the APPEAL prototype system for learning Dutch as a second language. In its present form the system can be classified as an Intelligent Computer-Assisted Language Learning/Instruction (ICALL/ICALI) system. The APPEAL system is primarily intended for usage in situations in which a teacher is not always available and in which the system is the only ‘conversation partner’ for the student. It will be obvious that a system in such a setting has to conform to even higher adaptiveness standards.

In this paper we will describe the APPEAL system and we will especially focus on two topics which are responsible for the high degree of adaptiveness within our system:

- modelling the tutor module (to be called TE: Teaching Expert) with situated agents.
- using state-of-the-art Natural Language Processing (NLP) techniques for the generation of adequate exercises.

In Section 2, we start off with an introduction to the architecture of the APPEAL system. In Section 3 and 4 we will describe in more detail the two most important components of the system. In Section 3 we will sketch the multi-agent approach to teacher modelling. In Section 4 we will focus on the NLP framework and the way it is exploited in APPEAL for the generation of language data that can be used in exercises. In Section 5 we will show how these language data are integrated with other media into an actual exercise. Finally, in Section 6 we give an impression of the overall user interface of the system. In Section 7 we will conclude with some remarks.

2 The APPEAL System

The APPEAL system is a modular, distributed system. Its architecture is shown in Figure 1.

From this figure it will be clear that the central component in the APPEAL system is not the expert module containing the domain knowledge as in traditional ICALL systems, but rather the Teaching Expert which is responsible for the complete dialogue with the student. The next sections will explain which choices in the design of the APPEAL system are responsible for this deviation in architecture.

The **instruction database** contains all the course material. It consists of a network of several types of nodes. For example, **theme** nodes contain instructive material and **skill** nodes contain specifications of learning objectives for a particular (sub)skill. The structure provides information about the

\(^1\)This institute is a cooperative venture between Philips Research Laboratories and the Eindhoven University of Technology.
relative level of difficulty of the contents of these nodes. The student history contains a history of the student's performance. The teaching expert is responsible for the instructive dialogue with the student, i.e. instructions and feedback. This component is supposed to be independent of the chosen learning domain. The domain model contains information about the domain (the Dutch language in the implemented version of the system). The domain expert generates exercises and examples on the basis of information provided by the teaching expert and the domain model. It also evaluates the student's answers. This component represents the domain-specific aspect of teaching. It also has knowledge of possible error types specific to this domain. The interface manager provides the interface between system and student. It translates, for example, the instructions and feedback provided by the teaching expert into the language chosen by the student at the beginning of a session.

3 THE TEACHING EXPERT

In classical ITSs the student's performance is interpreted within the representational domain model of the expert. This implies that the adaptive behaviour of the system crucially depends on the adequacy of this model for the particular student using the ITS. The adaptiveness of a classical ITS is determined by the extent to which the set of all possible behaviours of a student match the representational framework of the expert module. There are a number of problems associated with this approach. As we mentioned in the introduction, McArthur and Roschelle give empirical evidence that expert knowledge structures do not provide the most useful models for teaching. Furthermore, the student's performance is rarely consistent and it is in fact impossible to predict the full range of possible student behaviours.

In our opinion it is very difficult to improve the adaptiveness aspects within the classical ITS framework. Hence we chose an approach which is fundamentally different: it is data-driven or situation-driven rather than representation-driven and it relies on the student's behaviour itself rather than on the hypothesised knowledge state of the student.

In the APPEAL system the Teaching Expert is the module which is largely responsible for the adaptiveness of the system. It has been implemented by means of agents. These agents are situated agents as in the work of Wavish and Connah (1990), Connah (1993), Agre and Chapman (1987) and Chapman (1991), and are inspired by the work of Brooks (1991), Steele (1989) and Suchman (1987). Agents are autonomous entities that function continuously in an environment in which other processes take place and other agents exist (Shoham (1993)). Van Hoe and Maathoff (1993) explain that the following characteristics are relevant for an agent-based approach:

1. situatedness. An agent's most important resource in determining what to do is its immediate situation, rather than internal plans of action.

2. interactivity and emergent functionality. The activity pattern emerges from the dynamical interaction between an agent and its environment, rather than being explic-
itly designed as a property of an agent or the environment.

3. autonomy. On a micro-level, autonomy implies that the activities of the individual agents do not require constant external guidance. On a macro-level, there is no central authority to control the overall interactions between the agents.

4. cognitive economy. Agents do not maintain a symbolic model of the world: the world is its own best representation. This implies that perception is central in the functioning of an agent, as it is the agent's only access to the concrete situation.

5. parallelism. Agents function concurrently with each other in a multi-agent environment.

6. predictability. The environment in which the agents function is partially unpredictable.

7. behaviour-based. Agents are described in terms of their behaviour in response to other behaviour in their environment rather than in terms of their knowledge.

In the Teaching Expert an agent is meant to model specific aspects of a teacher's behaviour like navigating through course material, practicing, giving instruction and feedback, trying different ways of presentation of problems and keeping a student history of the student's performance. The idea is that the complex behaviour of the teacher emerges from the interaction between these internal agents and the only external agent, the student. For a more detailed discussion on the specification of these agents, see Van Hoe and Masthoff (1993) and Masthoff and Van Hoe (1994).

4 THE DOMAIN EXPERT

Most of the time NLP techniques are used in ICALL systems to enable students to enter free input into the system (see, for example, Catt and Hirst (1990)). At the moment, however, the state of the art in computational linguistics is such that it is not yet feasible to construct NLP systems that cover a complete language. Full natural language understanding by computers or unrestricted conversation with computers is even further away. At first sight it doesn't seem necessary that an ICALL system cover a complete natural language on all levels of representation (lexical, morphological, syntactic and semantic) in order to function adequately. The most frequently used argument is that in ICALL systems very small domains can be used that could narrow down the number of possible words, syntactic constructions and meanings that a student can reasonably use in an utterance at a given moment. Especially in a learning environment the type of feedback given after the student enters an utterance seems to be crucial. However, even if the domain is small we cannot exclude the possibility that students will enter utterances which are not expected by the system. This can happen if they make an error or if they enter a correct utterance which is beyond the capabilities of the NLP component. If the student enters an utterance that is outside the language fragment covered by the system, it is impossible to discover whether the utterance is correct or erroneous and hence, a feedback message which is to the point is very difficult to give. We will not add the possibility of free natural language input until we have a better insight into the effect of the feedback problem on the learning performance of the student.

For the moment we use NLP for controlled input of natural language. With 'controlled input' we mean that the system controls explicitly the way in which a student can construct a sentence. Multiple choice is the best known form of controlled input. There are, however, more creative ways to control the input of the student. For example, the system shows the picture of a clock to the student, gives him/her a set of sentence units and asks him/her (after giving an example) to construct a sentence that indicates what
time it is. The system itself knows what the answer should be. So it can immediately intervene if the student selects the wrong units or puts correct units in a wrong place. By extending the set of sentence units with erroneous ones, i.e. units which contain, for example, tense or agreement errors, the system has complete control over the type of errors that a student can make.

In the APPEAL system we use the language generation part of the Rosetta Machine Translation System\(^2\) to generate the sentence units a student can choose from. The input for the generation process is a semantic derivation tree expressing the meaning of a sentence. The framework that is used in the Rosetta system is the Controlled M-Grammar (CMG) formalism. In the next section (Section 4.1) we will give an introduction to the CMG formalism. Section 4.2 gives an example to show how related sentences of Dutch can be generated. How this leads to the definition of templates will be shown in Section 4.3. Finally, in Section 4.4 we will explain how the templates can be used as the basis for the generation of training material.

4.1 THE CMG FRAMEWORK

A grammar in the CMG framework (Rosetta (in press)) comprises various components, of which the syntactic component is the most salient. In this paper, the term CMG grammar,\(^3\) or simply grammar, will refer to the syntactic part only. It should be noted, however, that the organisation of a CMG grammar is strongly influenced by the fact that it also has a (parallel) semantic component. An essential property of CMG is that one and the same grammar can be used for both analysing and generating sentences. If used as a generative device, a grammar transforms sequences of basic expressions into valid sentences. Derivation trees indicate how a particular sentence is derived. Figure 2a is an abstract derivation tree for ‘Rudy koopt koeken’ (lit. Rudy buys biscuits). It represents the derivation of this sentence from the basic expressions V(kopen) (buy), N(Rudy) and N(biscuit) (koek), and shows that the sentence is derived in the form of the structured expression (1), which is called an S-tree (syntactic tree).

\[(1) \quad S(NP(N(Rudy))) \ V(koempt) \ NP(N(koeken)))\]

The structure in (1) conveys that ‘Rudy koopt koeken’ has two constituent noun phrases (NP) which together with the verb ‘koemp’ form a sentence (S). This S-tree is a simplified one. In the real system, the constituent structure is much more complicated and all kinds of features are attached to constituents (see Rosetta (in press)). However, for the purposes of this presentation even the structural aspects of S-trees are not essential and, hence, S-trees will be abbreviated by the sequences of words that occur in them.

Figure 2a indicates that the grammar can derive ‘Rudy koopt koeken’, but it abstracts from the way the components of the grammar accomplish this. We will explain the structure of CMG grammars by reducing the level of abstraction in derivation trees for ‘Rudy koopt koeken’ step by step.

An important characteristic of a CMG grammar is its modular structure. It is divided into subgrammars, each of which has a well-defined linguistic task, e.g. the construction of a noun phrase or a prepositional phrase (Appelo et al. (1987), Appelo (1993) or Rosetta (in press)). In the case of our example sentence, two subgrammars (SG) play a role\(^4\); SG-NtoNP, which turns the nouns ‘Rudy’ and ‘koeken’ into noun phrases and SG-VtoS, which turns the verb ‘kopen’ and the two noun phrases into a sentence; see Figure 2b. In this simple case the noun phrases

---

\(^2\)The Rosetta Machine Translation System was developed at Philips Research Laboratories in the period 1985-1991.

\(^3\)In previous publications, a CMG grammar was called an M-Grammar.

\(^4\)We describe a grammar for Dutch with the corresponding Rosetta grammar in mind, but for explanatory reasons it is necessary to simplify things. For a more realistic account of the Rosetta grammars, see Rosetta (in press).
contain only one noun, but noun phrases can be quite complex: they can contain adjectives, relative clauses, etcetera. These can all be produced by subgrammar SG-NtoNP.

The operation of a subgrammar is a composition of operations of rule classes\(^5\), which are sets of linguistically related grammar rules. The operation of subgrammar SG-VtoS in the derivation of ‘Rudy kookt koeken’, for instance, first invokes the rule class (RC) for subcategorisation, RC-SUBCAT, to combine the verb with its arguments. Then it calls for the rule class for voice, RC-VOICE, to determine whether the sentence to be produced will be an active sentence or a passive one. Then it uses the rule class for tense, RC-TENSE, to determine whether the sentence will be in the present tense, or in the past tense, or in one of the other tenses of Dutch, and the rule class for mood, RC-MOOD, to determine whether the sentence is declarative, interrogative, or imperative, and also whether it is a main sentence or a subordinate one. Figure 3a is a derivation tree for ‘Rudy kookt koeken’ with subgrammar SG-VtoS unfolded to the level of rule classes. The two occurrences of subgrammar SG-NtoNP can be unfolded in a similar way. Not every application of a subgrammar uses the same rule classes in the same order. The admissible orderings are defined by a regular expression over rule classes, one for each subgrammar (see Rosetta (in press)).

The next level of abstraction is obtained by specifying for each rule class the rule (R) that is actually used; see Figure 3b. In this figure, R-trans is a rule for transitive verbs, R-active makes active sentences, R-present determines the present tense of the sentence, and R-maindecl turns the sentence into a declarative main sentence. A derivation tree at the level of grammar rules is called a D-tree. A D-tree need not be the most detailed level of specifying a derivation: rules can also have inner structure, in which case they are composed of subrules, much like rule classes consist of rules. The level of D-tree is special because a rule has a primitive meaning, which is shared by all of its subrules. The rules in a rule class have different, but related, meanings.

4.2 SENTENCE TRANSFORMATIONS

Consider the following examples (2) and (3):

(2) Rudy kookt koeken.
Rudy buys biscuits.

(3) Er worden door Rudy koeken gekocht.
There are by Rudy biscuits bought.

If we want to change the active voice of the sentence in (2) into a passive voice as in (3), we change the rule R-active in the derivation tree for (2) into R-passive, as illustrated in Figure 4. The sentence that will be generated

\(^5\)A subgrammar also uses transformation classes (see Appelo (1993) or Rosetta (in press)), which are ignored in this presentation.
Rudy koopt koeken
RC-MOOD

| Rudy koopt koeken
RC-TENSE
| Rudy koeken kopen
RC-VOICE
| Rudy koeken kopen
RC-SUBCAT

(a)

Rudy koopt koeken
RC-MOOD

| Rudy koopt koeken
RC-TENSE
| Rudy koeken kopen
RC-VOICE
| Rudy koeken kopen
RC-SUBCAT

(b)

Figure 3: Derivation tree with subgrammar SG-VtoS unfolded to the level of rule classes (a) and to the level of grammar rules (b).

R-maindecl
| R-present
| R-passive
| R-trans

kopen SG-NtoNP SG-NtoNP

Rudy koeken koek

Figure 4: Derivation tree for ‘Er worden door Rudy koeken gekocht.’

by executing the rules in the transformed D-tree is the sentence in (3).

In the same way we can transform sentences in the present tense into sentences in the past tense, main sentences into subordinate sentences, declarative sentences into yes-no questions, and make combinations of those transformations.

Table 1 gives an overview of the resulting sentences, starting with sentence (2), which is a main declarative active sentence in the present tense.

4.3 Templates

All sentences in Table 1 have the same derivation structure at the level of rule classes. That is, the tree of Figure 3a is the same for all of them, apart from the outputs of the rule classes. Stated differently, the D-trees for these sentences are instantiations of the D-tree template of Figure 5a where voice, tense and mood are variables over rules. The notion of templates can be generalised by allowing parts of a D-tree to be replaced by variables of other types. One may have variables that range over basic expressions, so that ‘Rudy koopt koeken’ may be changed into ‘Judith eet koeken’ (lit. Judith eats biscuits). More powerful things can be done if variables may have D-trees as values. For instance, the template in Figure 5b contains a variable np ranging over D-trees that produce
Table 1: Overview of the possible transformations starting from Rudy koopt koeken.

<table>
<thead>
<tr>
<th>transformation</th>
<th>resulting sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>past</td>
<td>Rudy kocht koeken</td>
</tr>
<tr>
<td>question</td>
<td>koopt Rudy koeken?</td>
</tr>
<tr>
<td>subordinate</td>
<td>dat Rudy koeken koopt</td>
</tr>
<tr>
<td>passive</td>
<td>er worden door Rudy koeken gekocht</td>
</tr>
<tr>
<td>past question</td>
<td>kocht Rudy koeken?</td>
</tr>
<tr>
<td>past subordinate</td>
<td>dat Rudy koeken kocht</td>
</tr>
<tr>
<td>past passive</td>
<td>er worden door Rudy koeken gekocht</td>
</tr>
<tr>
<td>question subordinate</td>
<td>of Rudy koeken kocht</td>
</tr>
<tr>
<td>question passive</td>
<td>worden er door Rudy koeken gekocht</td>
</tr>
<tr>
<td>subordinate passive</td>
<td>dat er door Rudy koeken gekocht worden</td>
</tr>
<tr>
<td>past question subord.</td>
<td>of Rudy koeken kocht</td>
</tr>
<tr>
<td>past question passive</td>
<td>weren er door Rudy koeken gekocht</td>
</tr>
<tr>
<td>past subord. passive</td>
<td>dat er door Rudy koeken gekocht worden</td>
</tr>
<tr>
<td>question subord. passive</td>
<td>of er door Rudy koeken gekocht worden</td>
</tr>
<tr>
<td>past question subord. pass.</td>
<td>of er door Rudy koeken gekocht worden</td>
</tr>
</tbody>
</table>

A noun phrase, which may of course be more complex than just a word. For instance, the noun phrase might be 'Lisette en René' and the sentence 'er worden door Lisette en René koeken gegeten' (lit. there were by Lisette and René biscuits eaten) might be produced. The D-trees that are to be substituted for D-tree-valued variables may be obtained by instantiating other D-tree templates.

4.4 Generation of Training Material

Templates as described above are used to create 'stimulus-response pairs' in the Domain Expert module of the APPEAL system, as illustrated in (5) in a transformation type of exercise. The template is used twice. First a stimulus is generated and then the response by changing the value of the required parameter. The response sentence is used for the presentation to the student (extended with wrong alternatives) by the Interface Manager and for checking the response of the student.
(5) task: Give the passive form:

stimulus: Rudy koop koeken.
response: Er worden door Rudy koeken gekocht.

stimulus: Koop Rudy koeken?
response: Worden er door Rudy koeken gekocht?

The choice of basic expressions can be adapted in various ways, e.g., to a text which was presented earlier, to the lexicon acquired by the student and to a certain level of difficulty. The different values of rule class variables can be used for the generation of exercises to train, for example, past tense forms of strong and weak verbs in Dutch. ‘Kopen, kocht, gekocht’ (buy, bought, bought) is a strong or irregular verb, for example, while ‘maken, maakte, gemaakt’ (make, made, made) is a weak or regular one. The transformation from declarative to subordinate sentence is useful for learning Dutch word order. In subordinate sentences the conjugated verb form is usually at the end while in main declarative sentences it is in second position. In addition, the position of the particle of particle verbs, such as nadenken, binnenkomen, zich voordoen (Eng. think about, enter, occur), can be practiced with such exercises. An example is the transformation of sentence (6a) containing the particle verb nadenken with the particle (PART) na, into the subordinate sentence (6b).

(6a) a (main, decl., active, present)

b (subordinate, question, passive, past)

This example also illustrates how difficult the right position of the word er is. It can even disappear, as in the transformation of sentence (7a) into its subordinate counterpart (7b).

(7a) a (main, decl., active, present)

b (subordinate, decl., active, present)

dat er twee zijn

That er in passive sentences with an indefinite subject, as in (3), is also very difficult for foreigners. To make students learn the inversion of subject and verb we can generate exercises to practice the transformation of main declarative sentences into questions. Sometimes, the verb form changes in this situation as illustrated in (8):
5 Embedding the Generated Training Material in a Multimedia Context

In this section we will sketch how the training material generated in the way described in Section 4.4 is extended with multimedia material. We will do that in an operational way by describing step by step which actions are taken by the Teaching Expert, the Domain Expert and the Interface Manager.

In APPEAL, the Teaching Expert can decide, on the basis of information in the instruction network and the student history, that for a certain student at a certain moment a particular exercise type associated with a specific learning objective is required. The type of the exercise may be chosen on the basis of the student’s performance with different types of exercises. Next, the Teaching Expert can instruct the Domain Expert to generate an exercise of that specific type and of a specific difficulty level corresponding to the learning objective in question.

As we have seen in the previous sections the body of an exercise consists of a set of related sentences. The first task for the Domain Expert is to find out which sentences have to be generated. For that purpose it uses a database, the so-called World Database, which enables the Domain Expert to associate the given learning objective with an appropriate natural language expression. Figure 6 gives an overview of the structure of this database. One of the object types in

The wrong alternatives are generated in a controlled way by special error transformations. In the future, these transformations will draw on data from investigations of typical errors of second-language students. These transformations concern morphological errors (tense, person, etc.), and also syntactic ones, such as the place of verbs, or lexical ones, such as the choice of the auxiliary verb hebben or zijn for the Dutch perfect tense. The most important advantage of this approach is that, given a lexicon containing classified entries and a grammar for a certain language, a very large number of new, but appropriate, exercises can be offered to the student.

The diagram shows that the database is a world. A world can be a
picture, an animation or a very short movie. A world object contains information about its type and its contents, i.e. where the actual graphical, video and/or audio data can be found. A number of expression objects is associated with each world. An expression object contains a template expression, i.e. a template together with values for its parameters. Parameter values can of course also be template expressions. A template expression serves as the input for the NLP component of the Domain Expert for the generation of one or, in the case of ambiguous expressions, more sentences. The template expressions have been defined in such a way that all sentences that can be generated from them are valid in the particular world they are related to.

Information about a template is stored in a template object. A template object contains information about its type, i.e. about its parameter types and its result type in the functional view (cf. example 4 in Section 4.3). Finally, a set of learning objectives is related to each expression object (and vice versa). The database has been designed in such a way that in the future a kind of inference module can infer for one particular world new meaningful expressions from a given set of valid expressions. At the moment, however, all expressions that are valid in a specific world have to be explicitly listed in the database.

The learning objective chosen by the Teaching Expert can now be used by the Domain Expert to select an appropriate expression in the World Database. The selected expression gives access to a corresponding world, e.g. a short full motion video, and to the relevant template. Depending on the required exercise type the NLP component of the Domain Expert can generate a number of variants of the sentence that correspond with the retrieved template (cf. Section 4). It also may request speech synthesis for each of the generated sentences using speech-synthesis modules developed at IPO, using PSOLA techniques (Charpentier and Moulines (1993)). These modules are tuned to the output of the CMG grammar. After the sentences have been mapped onto a specific exercise format, this format together with the location of the relevant video and audio data is sent to the Interface Manager which will present them to the student. The intermediate and final results are sent back to the Domain Expert for evaluation. The results of this evaluation are forwarded from the Domain Expert to the Teaching Expert.

6 The Multimedia Aspects of the APPEAL Interface

In the previous sections a number of modalities have already been introduced. In this section we will once more go through them in a systematical way.

An important design criterion for the user interface of the APPEAL system has been Ease of Use. That is the main reason why we have opted for a single button mouse as the only tactile input medium. For the visual modality, i.e. the screen interface with the student, we use the classroom metaphor with a teacher, a blackboard, a working desk and a library. The classroom metaphor was chosen for reasons of recognisability. By opting for an environment each student is familiar with we try to eliminate the part of the learning phase in which the student uses most of his/her time trying to find out how the interface works. The screen has been subdivided into four windows, or areas as we call them (since we try to avoid a student wrongly associating them with the windows used in window-based operating systems for PC's and workstations). These areas are static and cannot be manipulated, i.e. moved, iconised, resized etc.

Figure 7 gives an idea of the APPEAL screen. In the upper left corner the teacher area is located. In this area the student sees a teacher in full motion video. The teacher gives feedback and instructions, encourages and/or comforts the student in response to his/her actions and gives help and examples
when the student asks for them. The behaviour of the teacher is the result of the complex interaction of the agents in the Teaching Expert (cf. Section 2). It is important to notice that the video the student sees in this area has not been prerecorded in this form. It consists of a great number of short video segments which are concatenated in the order the Teaching Expert demands. In the upper right corner the presentation area is situated. The presentation area is to be used as a blackboard, television or slide screen. In this area texts, pictures and films will be presented. Neither upper corner is accessible by the student. In the two lower areas on the screen the actions of the student are interpreted by the system. In the lower right area, the working area, the student has to perform exercises. In the lower left area, the information area, the student can consult dictionaries and grammar tables and can choose other course material.

Apart from this visual interface, there is also an auditor/visual interface with the student. Most of the text that appears in the presentation, the working and the information areas is supported by speech. Since some of the natural language expressions are generated on the fly this means that the system has to be able to synthesise the speech of some expressions that are presented to the student. The system also contains a small speech recognition module that is used in a vowel-duration exercise in which the duration of the vowel in particular Dutch words is trained. When the student is engaged in this exercise he/she has to repeat a word which is pronounced by the system. Next, the resulting speech signal is recorded via a microphone and analysed and evaluated by the Domain Expert.

7 CONCLUDING REMARKS

In this paper we have described the APPEAL system and the ideas behind it. Our main research goal has been to design a system with a flexible interaction, i.e. a system

- with a mixed locus of control
- which is adaptive with respect to type and contents of exercises and feedback
- which can be a more natural dialogue partner for the user.

In the current prototype we try to achieve this by introducing
the multi-agent approach to teacher modelling. The usage of situated agents enables us to make the system's behaviour sensitive to the performance of the user.

- natural language processing techniques for the generation of exercises. The adaptiveness level of the system is further increased by offering the student exercises which are tuned to his/her individual situation. This is possible by generating the language material used in the exercises on the spot.

- a multimedia environment. Multimedia are used (i) to put the language material in a realistic context and (ii) to support the dialogue with the user.

Since the emphasis was on methods that would increase the adaptive behaviour of the system rather than on courseware, the current APPEAL implementation comprises only a very small part of an imaginative Dutch course. However, extensions of the current courseware are planned and can be implemented easily because of the modularity of the world database and the graph-structured instruction database.

At the moment we are doing experimental research in order to obtain optimal values for some of the parameters of the system, e.g. for the rules used by the agents, for the kind of language errors that can be expected and for the alternatives that are possible with respect to the screen interface. Furthermore, it is one of our goals to develop a methodology which enables us to evaluate the overall system in realistic settings. In the current prototype we exploit natural language processing techniques to generate sufficient variation in training material in a multimedia context. In the near future we want to investigate how a further integration of natural language and multimedia can be brought about with techniques developed in the area of computational linguistics and artificial intelligence.

As a final remark we want to state that, although the APPEAL system is merely a prototype and a lot of research has to be carried out, we are already convinced of the fact that a combination of a multi-agent approach with NLP techniques in a multimedia context results in a more interactive system for language learning.

ACKNOWLEDGEMENTS

We are indebted to René Leermakers, Rudy van Hoe and Judith Masthoff for their helpful suggestions.

REFERENCES


waveform processing techniques for text-
to-speech synthesis using diphones. In
Proceedings Eurospeech 89, 2, pages 13–19,
1993.

[Connah, 1993] D.M. Connah. The design
of interacting agents for use in interfaces.
In M.D. Brouwer-Janse and T.L. Harring-
ton, editors, Human-machine commu-
nication for educational systems design, pages

and J.P.M. Masthoff. A multi-agent ap-
proach of interactive learning environ-
ments. In AI-ED 93 Workshop ‘Collabo-
ratitive problem solving: theoretical frame-
works and innovative systems’ at the
World Conference on Artificial Intelli-
gence in Education, Edinburgh, Scotland,
1993.

and C. Chapelle. Using call effectively:
what do we need to know about students?

[Masthoff and Hoe, 1994] J.F.M. Mast-
hoff and R.R.G. van Hoe. APPEAL: a
multi-agent approach to interactive learn-
ing environments. Manuscript submitted
for publication, IPO manuscript no. 1007,
1994.

[McArthur et al., 1988] D. McArthur,
C. Statz, J. Hotta, O. Peter, and C. Bur-
dorf. Skill-oriented task sequencing in an
intelligent tutor for basic algebra. Instruc-

for conversations. In Annual Meeting of
the American Education Research Associ-
aton, Boston, 1990.

[Rosetta, in press] M.T. Rosetta. Composi-
tional Translation. Kluwer, Dordrecht, in
press.

programming. Artificial Intelligence,

[Steels, 1989] L. Steels. Cooperation be-
tween distributed agents through self-
organisation. Journal on Robotics and Au-

[Suchman, 1987] L.A. Suchman. Plans and
situated actions: The problem of human-
machine communication. Cambridge Uni-

[Wavish and Connah, 1990]
P.R. Wavish and D.M. Connah. Repre-
senting multi-agent worlds in able. Tech-
nical Note, Philips Research Laboratories.
1990.

[Wenger, 1987] E. Wenger. Artificial Inte-
lligence and Tutoring Systems. Morgann
DEVELOPING A MULTIMEDIA COURSE
FOR LEARNING DUTCH AS A SECOND LANGUAGE

Betske Salverda
SLO
Boulevard 1945 3
7511 AA Enschede, The Netherlands

ABSTRACT

Three groups - two Dutch institutes (the SLO, the National Institute for Curriculum Development, and the RBO Rijnmond, a Regional Centre for the Innovation of Education,) and one multimedia developer (Q Multimedia) - have joined together to develop a course for Dutch as a Second Language. This project, aimed at providing extensive language practice for the thousands of immigrants arriving in the Netherlands each year and studying in special, government-funded schools - will cover all language skills, beginning with zero entry level (the alphabet). This course will be based on an especially-created video whose function is to both introduce students to the language and to local culture. In this paper, I will discuss in general the development process used by the team and in particular, the general pedagogical and technological principles we have adopted for our multimedia products.

II. ABOUT THE COURSE

The SLO together with RBO and Q Multimedia are in the process of developing a multimedia course for the teaching of Dutch as a Second Language for the large immigrant population studying Dutch in the special schools as well as in the workplace and other adult centres. The first two phases of this development effort will produce a prototype (by October of this year) and a course starting at the zero entry level and covering approximately 3500 vocabulary items, all of Dutch grammar, essential functions and strategies and typical situations of everyday life in the Netherlands (the first segments will be released around September 1995). The central theme of the course will be presented by a video which will be filmed on location throughout the Netherlands. The video will be divided into episodes of approximately 5-7 minutes (like a "soap opera") and all activities and student interactions will be connected thematically to the video in each segment. Using a CD-ROM as the delivery medium, the course will give students practice in listening and speaking, reading and writing at their own level and speed, with records being kept that will provide both students and teachers with an accurate account of individual progress. The course also presents students with a set of "tools" which will help them in the learning process: a bi-lingual dictionary with the option of specific vocabulary practice; language explanations with the option of specific practice; context-sensitive help and detailed progress reports. The course is being developed in the Netherlands with SLO and RBO providing the content expertise and Q Multimedia planning and organizing the general design and technical production.

The creation of interactive multimedia educational courses can be divided into four main phases: Planning, Development, Evaluation and Production (see Chart 1 below). Of particular
interest is the planning stage where decisions about the general nature of the product are made, based on specific philosophical and educational viewpoints as well as the capability of the medium. These decisions were made by the designers of the product who have extensive experience in the development of multimedia language courses together with the chief content people who have extensive experience in teaching Dutch as a Second Language and the technical consultant who is an expert in multimedia development.

III. PLANNING THE PRODUCT

A. THE VISION

The first decisions taken concerned the vision of the product based on educational goals and principles and selected technological parameters. The following decisions were made:

LEARNING

1. Student-centred. The program sets up a student-centred (and not teacher-centred) environment that encourages learning (i.e., freedom of choice, variety of activities, problem solving, etc.).
2. Democratic environment. All decisions are made by the student; nothing is obligatory; there is no set order or forced sequence although suggestions are given and a recommended order is implied in the organisation of the material.
3. Different learning levels. The course provides a variety of opportunities to cover the material so slower students can have more exposure.
4. Self-paced. Students can go at their own pace.
5. Redundancy. To account for different learning styles, each topic is represented in different ways so students can select the way they wish to learn the subject.
6. Use of native language. To ensure that the program can be used easily without a great deal of teacher intervention, many of the options will be available in the learner’s native language.
7. Extensive tools. A group of help-like functions will be available to assist the learner in doing the activities and tasks: i.e. bi-lingual dictionary, language explanations, help, etc. Most of these options will be available in the user’s native language.
8. Student records. The program will keep track of student progress to provide progress reports to students and teachers and will make suggestions on the basis of student achievement.

CURRICULUM

1. Organisation. The curriculum is organised according to accepted curriculum standards and existing methods used in the Netherlands.
2. Remediation. Weaker students will have the possibility of additional practice as suggested by the computer when needed.
3. All language skills. All aspects of language (grammar, vocabulary, functions, language use) will be handled by the course.
4. All presentation modes. The course will allow students to practice speaking, listening, reading, and writing.
5. Authentic language. The course is based on a video which will represent authentic situations and use authentic language.
6. Situationally-based. The course is situationally based with the pedagogical elements being derived directly from the situations.
7. Appropriate for all cultural backgrounds.

USER INTERFACE

1. No assumptions about previous knowledge of computers or operating systems such as Windows.
2. Object-oriented interface - only icons.
4. No extensive menus.
5. Students should have free access to all options - choice not obligation.

TECHNOLOGY

1. Windows operating system
2. CD-ROM based, with extensive use of video and sound
3. Super VGA (256 colours)
4. Use of programming shell to facilitate production process
5. Maximum amount of video and sound
B. THE LOOK & FEEL
Computer-based multimedia is a visual medium and not a computerised book. Any multimedia course will be compared to TV and arcade games and not to computer programs. Therefore, the product should make a strong visual statement which must be determined very carefully. Wherever possible, the look of the screen should reflect the reality of the world or task being represented (in the form of a metaphor). Thus, if our goal is to give students survival skills, the course should present as realistically as possible those specific situations we are training them to cope with. For this course, it has been decided that there will be two main design elements: the video and extracts from the video (both visual and sound presenting the real world and authentic language) and the screen layout which will complement the metaphor of the video while providing access to different activities based on the video.

C. THE INTERNAL DESIGN
Preparation of the internal design of the program is the implementation of our general vision. It is at this point that we have had to translate our philosophy into action, while keeping in mind the look and feel of the product and technical limitations of the computer. Such specific decisions as student input types, student judging, feedback messages, branching, etc. need to be made along with detailed definition of exercise type and format. It is at this point that heated discussion takes place between the courseware designers and the curriculum specialists, trying to find a compromise that will make both sides happy. Every potential interaction must be specified in detail to eliminate potentially disastrous situations where the program is ready but one discovers that something has been left out.

D. THE TECHNICAL DESIGN
At this stage of the process, our general technical specifications are being determined. Programming will be done at a software house in the Netherlands including the authoring environment. They will act as technical subcontractors and have nothing to do with the design of the product - it is very important for the technical people (including programmers of course) to understand that they do not determine the design of the product. The entire development process is lead by the designer with the content expert. The programming staff only implements decisions - it does not make them.

E. THE STAFF
So far our staff consists of the decision makers who act as the creative input to the project: Project Leaders (1 from SLO, RBO and Q Multimedia); 3 curriculum specialists (2 from SLO, 1 from RBO); 2 designers (from Q Multimedia); technical expert (Q Multimedia); graphic artist (Q Multimedia) and a video concept specialist (subcontractor). This will be the staff working on the prototype along with the programmers (subcontractor). Once work on the entire project is begun, we will be hiring the implementers: teacher-writers, editors, data entry people, etc.

IV. CONCLUSIONS
Working on this project has forced us to not only think about the organisation of material but also to make critical philosophical decisions about language learning and presentation styles. This very productive process of examining our basic assumptions has also had an effect on the standards we have established for evaluating other multimedia products. We thoroughly believe that a successful interactive multimedia learning course must on the one hand, take full advantage of the visual elements of that technology and on the other hand, utilise the strength of the technology to provide a successful learning environment.

CHART 1

Planning Phase
- Determination of the goals of the course and the basic philosophy underlying the approach (the “vision”);
- Determination of the physical characteristics of the course (“look and feel”);
- Specification of the structure of the course (the “internal design”);
- Specification of the visual elements (the “graphic design”);
- Selection of technical environment (“technical design”);
- Determination of personnel needs (“team creation”);
RESULT: DESIGN DOCUMENT, WORK PLANS
Development Phase

- Preparation of general program
- Preparation of content
- Preparation of video and sound elements
- Preparation of graphics
- Integration of elements into one “program”
- Initial testing

RESULT: FIRST VERSION OF COURSE

Evaluation Phase

- Quality Control
- Beta Site Testing
- Revision on basis of feedback

PRODUCT: MASTER OF COURSE

Production Phase

- Preparation of accompanying materials (user’s manual, teacher’s manual, etc.)
- Preparation of packaging
- Pressing of CD-ROM

PRODUCT: COMMERCIAL PRODUCT
ERROR ANALYSIS AND EXPLANATION IN KNOWLEDGE BASED
LANGUAGE TUTORING

Camilla B. Schwind
CNRS, Laboratoire d'Informatique de Marseille
Faculté des Sciences de Luminy, Case 901
163 Avenue de Luminy
13288 Marseille, Cédex 9
FRANCE
e-Mail: schwind@gia.univ-mrs.fr

ABSTRACT

In this paper, we present a uniform framework for dealing with errors in natural language sentences within the context of automated second language teaching. The idea is to use a feature grammar and to analyse ill-formed sentences as being sentences where features have other values than those they should have. Using a feature grammar it is possible to describe various types of errors (agreement, syntactic and semantic errors) in a uniform framework, to define in a clear and transparent way what an error is and -this is very important for our application- to analyse errors as arising from a misunderstanding or ignorance of grammatical rules on the part of the students.

1. INTRODUCTION

The problem of the treatment of ill-formed input has frequently been addressed in the context of various types of natural language understanding (NLU) systems (Fink and Biermann 1986; AIJCL 1983). The treatment and even the definition of errors in a language tutoring (LT) context are very different from those in other applications, because the goal is different. In most applications, the goal is to understand a sentence despite any errors, i.e. to somehow analyse the sentence. In a language tutoring system, the goal is to understand what the student wanted to do, where he went wrong and what grammar rules he misunderstood or was unaware of. In this respect, error treatment is more difficult in the context of language tutoring than in other contexts, because errors do not need to be just ignored or sentences to be parsed despite any errors, but the errors and the reasons for them have to be understood. To formulate this concisely, we could say: we are attempting to perform sensitive parsing rather than robust parsing. This means that we aim to achieve a system which is very sensitive to all possible kinds of errors and which moreover tries to find the "whys" and the "hows" of errors.

In (Weischedel 1983), the authors define absolute and relative ill-formedness, the first relating to utterances not belonging to a typical listener's language competence and the second relating to utterances not belonging to the system's language competence. In our application, we are concerned only with absolute ill-formedness, because we assume that the system can analyse all well-formed sentences. Clearly, this is a very strong assumption (and generally speaking it is not very realistic). But it is possible to make it because in a language tutoring context we are always working on a sub-language which is entirely known by the system, the language which the student is learning. Hence we can always assume that the system knows all the grammar rules needed to analyse the student's natural language input. The existence of a sub-language is a basic (and necessary) assumption made by everyone involved in language teaching. When a student studies a foreign (second!) language he generally does it with a textbook containing a finite number of grammatical phenomena and a limited vocabulary. Moreover, most methods work with a (finite) limited set of typical situations on the background of which exercises are conceived. This corpus will constitute the sub-language which the student will try to acquire. Clearly, the language tutoring system to which the error analysis system is applied has to ensure that the student does only form sentences which can be analysed, i.e. does not form well-formed sentences outside the competence of the system. This requirement is fulfilled by formulating the exercises so as to suggest a restricted language to the student. The language tutoring system in which our error analysis system is applied is described in (Schwind 1986; 1990a). Perhaps we could formulate it in rather overrified
terms by saying that in a language tutoring system such as the one we have conceived, the language is not really natural language but a somewhat artificial language which is closely related to the natural language and is described in students' grammar books.

A typical example for the difference between this and other applications can be found in the agreement rules. Generally speaking, agreement rules are very often ignored in natural language understanding systems because the sentences can usually be understood despite any agreement errors. Weischedel has proposed in (Weischedel and Sondheimer 1983) to suppress the subject-verb agreement test when it fails. For every such error there are at least two possible corrections: either the verb or the subject does not have the correct number or person. We propose in this case to tell the student that he has made an error in the number or the person between the subject and the verb of the sentence. It is however conceivable to ignore such errors in a natural language understanding system because the sentence can be approximately understood, even when the verb and the subject do not agree. But in an language teaching environment, it is really important that the system be able to correctly apply agreement rules because this is one of the objectives of language learning.

It was proposed to classify errors into morphological, syntactic, semantic and pragmatic errors. Within this classification, we drew finer distinctions, and ended up with the following seven groups (Figure 1):

1. Spelling errors (these will not be treated further within this paper).

2. Morphological errors: these are errors in the construction of words. In German, indefinite articles, nouns, verbs and adjectives are constructed by appending to the root of the word a suffix indicating the number, case and gender in the case of adjectives, articles and nouns, and the person, time and number in the case of verbs. There exist several groups of nouns and verbs, which depend on the form of the suffixes and on whether any changes are made in the roots when suffixes are attached. Typically, students of German mix the groups by appending to a word in a certain group an ending belonging to another group. The treatment of this kind of error is not very complex, provided the student takes an existing ending and an existing root.

3. Agreement errors: In German, articles, adjectives and nouns in a noun group must agree in gender, number and case, and verbs must agree in person and number with the subject noun phrase of the sentence. The object complements of verbs take certain cases and so do prepositions. Agreement errors are errors on the syntactic level, but they do not concern the order of the words in a sentence, hence they can be corrected by changing the case, the number, the person or the gender of the noun phrases or parts of them.

4. Syntactic errors: We consider two types of syntactic errors, the first involving words which have been omitted or added, e.g. when an article or a preposition is missing or superfluous, and the second involving the permutation of words or syntactic groups. The latter error is very frequent in German because here the possible places of verbs in a sentence can differ from many other languages: for example, the verb can go to the very end of a sentence or to the very beginning. Some syntactic errors have to be partly analysed at the morphological level because they involve both word construction and word order: for
example, in German some verbs have a prefix which in certain cases has to be detached and placed at the end of the sentence.

(1) *Er kommt zurück (He comes back).

This is the correct formulation of the sentence. Students of German tend typically not to detach the prefix zurück and construct the ill-formed sentence

(2) *Er zurückkommt (He backcomes).

The word zurückkommt does not exist in German (although the infinitive is zurückkommen) and this error has to be recognized at the word level, because zurückkommt is an ill-formed word, although the underlying error is a syntactic error.

5. Semantic Errors: We have actually been working on one type of semantic errors, namely errors in the semantic verb cases which arise from a misunderstanding of the meaning of verbs and nouns and their semantic relationships, and are thus caused by lexical problems. For example, when a student forms the sentence

(3) *Das Heft arbeitet (The notebook works).

he has not understood that arbeiten requires an animate subject and that Heft is not animate, i.e. he has a lexical problem.

6. Logical errors are based on a misconception of logical properties of connectives.

(4) *Neither I nor my brother eat neither meat nor salad.

Sentence (4) cannot be translated into a logical formula because the connective A(\_\_\_\_\_\_) is not self-distributive, i.e. A(A(a∧b)c) is not equivalent to A(a∧A(b∧c)). It is therefore not possible to unambiguously assign a logical formula to (4).

7. Pragmatic errors are based on a misconception of the world, i.e. of the context set up by the vocabulary of the exercise. For example, the sentence "He took a beer in the restaurant in the sea" is wrong because of a misconception in the nonlinguistic data. It is not clear whether errors of this type should be treated by a language tutoring system, because they are due not to language misconceptions, but to extralinguistic causes. They might however possibly be due to misconceptions about lexical and semantic relations.

In this paper, we will deal only with agreement errors and syntactic and semantic errors. In our opinion, logical and pragmatic errors should not be treated within the context of language teaching because the reasons for which people make such errors are not to be found in their incompetence in the foreign language and hence cannot be explained in terms of a lack of grammatical competence. The second group, (morphological errors), is very important for language teaching but we have not dealt with these so far.

In the rest of this paper, we will first describe the theoretical background to error definition. In chapter 2, the concepts of feature grammar and unification are defined. Complex features have been used by many schools of linguistics (Kaplan and Bresnan 1983; Karttunen 1984). The whole process of syntactic analysis is governed by features and their values. Not only lexical elements are classified by features but so are the syntactic categories. For example, the category "sentence" is subclassified by the feature satzstellung, the values of which indicate whether the sentence has a normal word order or has the verb at the end or at the beginning, (this corresponds in German to different types of embedded phrases), and by the feature embedded with values + and - indicating whether a sentence or a noun phrase contains any embedded phrases. We have constructed a grammar using about 25 syntactic and 60 semantic features. This formalism will be used to provide formal definitions for agreement, syntactic, and semantic errors. To our knowledge, feature grammars have not yet been applied to the problem of analysing ill-formed sentences. (Catt 1990) employs a feature grammar formalism for the syntactic analysis of his system but errors are not analysed by means of this formalism. He does not provide a definition of errors based on the feature grammar formalism.

Catt’s system is based on a model of the linguistic competence of the second language learner in analysing errors. Rules for language transfer and rule overgeneralization, which have been identified as the chief sources of error in learner language, are systematically used for error diagnosis. This is surely a powerful model but important classes of errors cannot be recognized, namely all errors concerning phenomena which
occur only in the second language of the student but not in his native language, and which are not due to overgeneralization. For an example the whole problem of noun phrase agreement errors in German cannot be tackled (at least not completely) by transfer rules (concerning English or French) (or only in a very limited way) because English and French do not decline adjectives and nouns.

The same deficiency is suffering Schuster's system (Schuster 1986) who incorporated a grammar of the native language for error analysis of the second language.

We incorporated to some extent transfer rules namely in order to analyse typical syntactic errors for German and in the hope to handle better prepositional phrases, but we did not so in a systematic way, i.e. by using explicitely the grammar. We think however that using transfer rules is rather an ad hoc method, because errors are anticipated by the system. Moreover this cannot be the exclusive approach because not all kinds of errors are based on transfer.

Related research has also been carried out by (Menzel 1986; 1988; Pullman 1984; Weischedel et al. 1978). Menzel's system is much more limited than ours, because he handles only fragmentory utterances (without really performing syntactical analysis) whereas we attempt to analyse the correctness of complete sentences which are freely formed by students. Weischedel was the first to address this problem in the context of language tutoring. His treatment of syntactic and agreement errors is very similar to ours but less general and less thorough. Errors are only recognized, but not explained, and the crucial problem of ambiguity in error analysis is not dealt with.

(Jensen et al. 1983) present a text-critiquing system called CRITIQUE which provides error diagnosis for diction, spelling, grammar and style. Originally intended for use in an office system it has been proposed that CRITIQUE could be used as a CALL tool in teaching English as a second language (Richardson and Braden-Harder 1988). CRITIQUE is based on a feature grammar formalism but its error detection mechanism is not. Commonly occurring (anticipated) errors are encoded directly as phrase structure rules. Other errors, including violations of grammatical agreement rules, are recognized through "constraint relaxation": cooccurrence restrictions on constituents of a phrase are relaxed so as to permit the recognition of ill-formed structures. Clearly, this technique is not acceptable in a language tutoring system since here the goal is not to permit ill-formed structures but to analyse the ill-formedness as we have pointed out above.

In chapter 2, we introduce shortly the feature grammar formalism we have used for error definition.

In chapter 3, we will give formal definitions of agreement, syntactic and of some kind of semantic errors. In order to actually deal with errors we use also error rules based on transfer phenomena.

Our system is implemented in PROLOG and has been tested with various dictionaries and by various language student users (adults, students, children).

2. THEORETICAL BACKGROUND

Our treatment of errors is based on the analysis of errors as being not-unifiable sentence analysis elements.

2.1 Complex features symbols

Roughly speaking, a feature symbol is a set of pairs [feature, value] where value can be a set denoted by {...} or by disjunction or negation. A complex feature symbol (cfs) is a set of of feature symbols. For a more formal introduction, see (Schwind 1990; Kaspar and Rounds 1986; ...). We will represent feature symbols by columns and cfs by sets of columns.

Most current theories also allow for features that have complex values. Using disjunction and negation of values, sets of feature symbols can be written much more economically. For example, the German noun Kind can have three cases (nominative, dative and accusative), but with complex values, it is denoted by just one symbol which contains the value neg(genitive) for Case.

f(orf(v,w)) means that the value of f is the set \{v, w\}; f(neg(v)) means that it is \(V_f \setminus \{v\}\), where \(V_f\) is the set of possible values for f; f(f), which is noted f, means that it is \(V_f\), The set of cfs over a set of features F with value sets V is denoted by C (F, V) (V being the union of all \(V_f\)).
Example 1. The definite article *der* is described by the cfs

<table>
<thead>
<tr>
<th>Art-cat(def)</th>
<th>Art-cat(def)</th>
<th>Art-cat(def)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gender</td>
<td>Gender(fem)</td>
<td>Gender(masc)</td>
</tr>
<tr>
<td>Case(genitive)</td>
<td>Case(or(genitive, dative))</td>
<td>Case(nominative)</td>
</tr>
<tr>
<td>Number(plural)</td>
<td>Number(singular)</td>
<td>Number(singular)</td>
</tr>
</tbody>
</table>

The morphological features of the noun *Lehrer* (teacher) are described by the cfs

\[ n_1 = \begin{align*}
 & \text{Gender(masc)} \\
 & \text{Case(neg(genitive))} \\
 & \text{Number(singular)}
\end{align*} \]

\[ n_2 = \begin{align*}
 & \text{Gender(neut)} \\
 & \text{Case(neg(genitive))} \\
 & \text{Number(singular)}
\end{align*} \]

The morphological features of the noun *Kind* (child) are described by the cfs

\[ n_3 = \begin{align*}
 & \text{Gender(masc)} \\
 & \text{Case(neg(dative))} \\
 & \text{Number(plural)}
\end{align*} \]

The notation of cfs could further be contracted by setting aside common elements of the set, yielding for the teacher example the symbol

\[ \text{Gender(masc)} \lor (\text{Case(neg(genitive))}, \text{Number(singular)}) \]

But this notation would be neither clearer nor more efficient for the unification algorithm.

Some notations. In order to express the fact that two symbols have the same features and different values for some set of features \( G \), we write

\[ a \not\in_G b \iff a - b \text{ and } a \not\in b \text{ and } \forall f \in G a(f) \not= b(f). \]

In order to express that \( a \) and \( b \) agree on common features, we write

\[ a \cap b \iff \forall f \in d(a) \cap d(b), a(f) = b(f). \]

The formalism introduced so far provides a uniform tool for describing natural language categories and their morphological, syntactic and semantic features. Formally, there is no distinction between different kinds of features, but conceptually, we distinguish the classes mentioned above. Examples of syntactic features are cat (category), vcat (verbal category), place, v-pos. cat plays a special part, and its values are the categories usually needed in a natural language grammar: sg (sentence), np (noun phrase), vp (verb phrase), etc. vcat is a feature the values of which are intrans (intransitive) and trans (transitive) and prep (prepositional complement); the feature place also subclassifies verbs and its possible values are the numbers 1, 2, 3, standing for the number of complements of a verb. The position of the verb within a sentence is indicated by the feature verb-pos (in German it can be at the top, in the second position or at the end of a sentence). Tense is a feature with the properties pluperfect, imperfect, perf, pres, fut specifying the tense of a verb. Other more frequently cited features are Case, Number, Gender which characterize articles, nouns, adjectives, but also noun phrases and noun groups. Examples of semantic features are sem-char, which belongs to nouns and has values which are animate, concrete etc. subj-sem, obj-sem, are features which classify verbs and have the same value set as sem-char and can be used to describe semantic constraints on verb complements. The value sets of semantic features can be ordered by a
hierarchical relation sup which is also used in the definition of the unification predicate, and which will be introduced in the next subchapter.

2.3. Unification

Our definition of unification is slightly different from the usual definition (see Karttunen 1984) as well as from the definition given in (Schwind 1990) in two respects.

- In our application we need not only to find whether two symbols can be unified but also for what reasons they might possibly not be unified. Hence we need to know all the pairs of elements which cannot be unified. The result of the unification is then the unified elements and the set of the pairs of elements for which the unification did not work. This set is necessary for interpreting and explaining errors.

- There exist relations defined on value sets (e.g. Vsem-char is ordered by sup) which are used in the unification definition and change it.

Unification is defined as a 4-place predicate on sets of complex symbols. unify(a, b, r, e) means that the result of unifying a and b is r and e is the list of features in which a and b disagree together with the pairs of disagreeing values. Given a = \{a_1, a_2, ..., a_n\} and b = \{b_1, b_2, ..., b_m\}, where all \(a_i\) and \(b_j\) are of the form \(\{f_1(v_1), ..., f_n(v_n)\}\) the unification of a and b is defined in the following way:

\[\text{unification}(a, b, r, e) \text{ holds if } r = \tau(r', \exists x \in a, \exists y \in b, \text{ such that } \text{unify}(x, y) = (r', e')); e = \tau(e'; \exists x \in a, \exists y \in b, \text{ such that } \text{unify}(x, y) = (r', e'))\]

The unification is obviously successful when \(r = \emptyset\). unify is defined in the following way:

\[\text{unify}(a, b, r, e) \text{ holds whenever } r = \tau(r'; \forall (v \in a \cap b) \text{ and } v = a(f) \cap b(f) \text{ whenever } a(f) \cap b(f) = \emptyset \text{ or } \exists x \in a(f) \text{ and } \exists y \in b(f) \text{ and } x \supset y \text{ and } v = y); e = \tau(e'; \forall (v \in a \cap b) \text{ and } a(f) \cap b(f) = \emptyset \text{ and } v = \leq(a(f), b(f))}\]

\((-\leq\) denotes the symmetric difference between sets)

r is the result of the unification and e is the set consisting of all the pairs of values for which a and b could not be unified, together with all the symbols contained in the symmetrical difference between a and b.

Example 2. Let c, n1, n2 be the cfs from example 1, unification(c, n1, r, e) evaluates to:

\[
\begin{align*}
r &= \text{Art-cat(def)} \\
&\quad \text{Gender(masc)} \\
&\quad \text{Case( genitive)} \\
&\quad \text{Number(plural)} \\
&\text{Art-cat(def)} \\
&\quad \text{Gender(masc)} \\
&\quad \text{Case(nominaive)} \\
&\quad \text{Number(singular)} \\
e &= \text{Art-cat(def)} \\
&\quad \text{Gender(masc)} \\
&\quad \text{Case(< genitive, neg( genitive))} \\
&\quad \text{Number(<plural, singular>)} \\
&\text{Art-cat(def)} \\
&\quad \text{Gender(<fem,masc>)} \\
&\quad \text{Case(genitive)} \\
&\quad \text{Number(<singular, plural>)}
\end{align*}
\]

\[{\text{Art-cat(def), Gender(masc), Case(< genitive, neg( genitive)), Number(<plural, singular>) }}, \{\text{Art-cat(def), Gender(<fem, masc>), Case(genitive)}, \\
\text{Number(<singular, plural>) }\}, \{\text{Art-cat(def), Gender(<fem, masc>), Case(dative)}, \text{Number(singular) }\}, \{\text{Art-cat(def), Gender(masc), Case(nominaive), Number(<singular, plural>) }\}]}.

unification(c, n2, r, e) gives

\[
r = \emptyset \\
e = \{{\text{Art-cat(def), Gender(neut), Case(< genitive, neg( genitive) )}, \text{Number(<plural, singular>) }}, \{\text{Art-cat(def), Gender(<fem, neut>), Case(dative), \text{Number(singular) }}, \{\text{Art-cat(def), Gender(<masc, neut>), Case(nominaive), \text{Number(singular) }}\}].
\]
The value set intersections are formed in the following way (the list is not exhaustive):

\[or(v, w) \land neg(v) = w \]
\[neg(v) \land v = \emptyset \]
\[neg(v) \land w = w \text{ if } v \neq w \]
\[or(v, w) \land w = w \]
\[or(v, w) \land u = \emptyset \text{ if } u \neq v \text{ and } u \neq w \]

These intersections are calculated by the predicate unify-value:

unify-value(v1, v2, r, e)
where \( r = v1 \land v2 \) and \( e = <v1, v2> \) if \( r = \emptyset \) and \( e = \emptyset \) if not.

Since unification is applied essentially to verify syntactic or semantic agreement of common features of different categories, this predicate will be applied systematically to cfs after suppressing the feature cat, i.e. to some \( a \setminus \{\text{cat}(x)\} \) for some cfs a. Otherwise, two different categories (e.g. article and noun) would never unify. Therefore we define unify-c as unify(a \( \setminus \{\text{cat}(x)\} \), b \( \setminus \{\text{cat}(y)\} \)) and unify-value as unification with unify replaced by unify-c.

The predicates unification, unify, unify-c and unify-value can all occur within feature grammar rules. unify-value is useful when it has just to be verified that two value sets have common elements, without producing the result of the unification (for example agreement of noun phrase cases with the cases taken by a verb or agreement of the number and the person of a verb and of the subject noun phrase).

2.4. Feature Grammars

Feature grammars are defined as formal or transformational grammars manipulating strings of complex symbols and the derivability concept is modified depending on the structures of the cfs. To each production and transformational rule there belong operations on the cfs involved. Often, production rules are independent from the subclassification of the alphabetic elements. Let us consider, for example, the rule \([\text{cat}(np)] \rightarrow [\text{cat}(det) \text{, det-art}(\text{indef})] [\text{cat}(ng)] \) replacing "noun phrase" by "indefinite article" "noun group". The rule which replaces "noun phrase" by "definite article" "noun group" has the same structure. We therefore wish to write a rule \([\text{cat}(np)] \rightarrow [\text{cat}(det) \text{, det-art}(\text{def})] [\text{cat}(ng)] \) representing all rules of the type \([\text{cat}(np), \ldots] \rightarrow [\text{cat}(det), \ldots] [\text{cat}(ng), \ldots] \). The feature grammars provide this possibility as well as the possibility of adding logical predicates to the rules. A rule can then be applied only when these predicates are satisfied.

Definition 1. A feature grammar over \((F, V)\), \(V = \{V_1\}_{F \in F}\) is a tuple:

\((F, \{V_F\}_{F \in F}, \text{cat}, s, \Sigma, \Pi)\), where

\(F \neq \emptyset\) finite set of features
\((V_F)_{F \in F}\) family of finite sets, value sets for features
\(cat \in F\) starting feature
\(s \in V_{cat}\) starting value of cat
\(\Sigma \subseteq V_{cat}\) set of terminal values of cat
\(\Pi \subseteq C(F, V)^* \times C(F, V)^* \times EQ\) set of production rules, where for every \(a \in C(F, V)\) occurring within a production

\(p \in \Pi: \text{cat} \in d(a)\)

\(EQ\) is a conjunction of literals formed with the predicates and operators, \(\neq, \subset, \text{unification, unification-c, unify-value, t, p, and} \\setminus\).

(For an alphabet \(A\), \(A^+\) the set of nonempty strings over \(A\).) The derivability concept for feature grammars has to be extended in such a way that, given a production rule \((p, q)\), we can apply all production rules \((p', q')\) provided that \(p \subseteq p'\) and \(q \subseteq q'\).

The alphabet of a feature grammar consists of all cfs containing the feature cat, the terminal alphabet of all those where cat has a terminal value, i.e.

\(C_{\Sigma}(F, V) = \{a \in C(F, V) \text{ such that cat} \in d(a) \text{ and } a(\text{cat}) \in \Sigma\}\)

Definition 2. Given a feature grammar, let \(x, y \in C(F, V)^*\). Then \(y\) is derivable from \(x\) in \(\Pi\), \(x \Rightarrow_\Pi y\)
if \(x\) and \(y\) have the same structure.

The set of sentences derivable by a feature grammar \(G\) is

\(L(G) = \{x: x \in C(F, V)^* \text{ and } [\text{cat}(sg)] \rightarrow x\}\).

\(L_\Pi(G) = L(G) \cap (C_{\Sigma}(F, V))^*\) is the language generated by \(G\).

The type of a feature grammar is defined in exactly the same way as the type of a formal grammar (type 0, 1, 2, 3).
In (Schwind 90b) we have shown that for every feature grammar \( G \) of type 1 there is an equivalent formal grammar of the same type.

Syntax trees for type 2 grammars are defined analogously to the usual way. Clearly, the nodes of feature grammar trees are marked by cfs. Any other type of grammar (transformational, gapping, etc.) can be straightforwardly defined according to an alphabet which manipulates features. For the purpose of this paper, we will use transformational feature grammars below.

**Example 3.** The following rule of German defines noun phrases of the form

"article" "noun group".

\[
\text{[cat(np)]} \cup r \rightarrow \text{[cat(det)]}_1 \text{[cat(nf)]}_2 \\
\] unification-cf \( v_1, v_2, r ), e \neq \emptyset
\]

In order to write grammar rules economically, we can index the cfs occurring within a grammar rule and use these indexes within the predicates just like variables \((v_1, v_2)\). Note that \( \cup \) is defined only for compatible cfs. By theorem 1, \( v_1 \) and \( v_2 \) are compatible iff \( r \neq \emptyset \). This rule analyses

\[
\{(\text{cat(det)}, \text{Art-cat(def)}, \text{Gender, Case(genitive)}, \\
\text{Number(plural)}), \text{[cat(det), Art-cat(def), Gender[gen]}, \\
\text{Case(genitive, dative)]}, \text{Number(singular)}, \text{[cat(det),} \\
\text{Art-cat(def), Gender(masc), Case(nomnativaive)}, \\
\text{Number(singular)])} \\
\]

\[
\{(\text{cat(nf)}, \text{Gender(masc), Case[genitive]}, \\
\text{Number(singular)}, \text{sem-char[human]}), \text{[cat(nf),} \\
\text{Gender(masc), Case[genitive]}, \text{Number(plural), sem-char[human]]} \\
\]

which represents "der Lehrer", to

\[
\{(\text{cat(np), Art-cat(def), Gender(masc), Case[genitive]}, \\
\text{Number(plural), sem-char[human]}), \text{[cat(np), Art-} \\
\text{cat(def), Gender(masc), Case(nomnativaive),} \\
\text{Number(singular), sem-char[human])} \\
\]

2.5. The logic programming treatment of feature grammars and unification

Feature grammars and unification can be straightforwardly implemented in PROLOG using definite clause grammars enriched by calls of cfs manipulating predicates. Every rule \((p,q,c)\) can be translated directly into a PROLOG clause where \( c \) is a sequence of predicates written in PROLOG. A cfs is represented as a tree. We have not opted for a representation by acyclic graph or oriented tree representation as described in PATRII (Shieber et al. 1983) or in (Kaplan and Bresnan 1982; Karttunen 1984; Pereira and Shieber 1984) because we wanted features not to be ordered within a symbol. We accept any permutation because we view a cfs as a set, which is not ordered. Hence in our system, there are no predefined places for features. We will not explain here how the definite clause grammar formalism works and how transformational grammars can be represented in this framework and implemented in PROLOG (see Abramson and Dahl 1989). We used a formalism which resembles these formalisms but which is adapted to our application, where transformations are basically needed in order to move the verb or parts of the verb phrase. It can be seen quite easily from the examples which we give in the next chapter how we treated transformational rules by equations on trees.

3. ERROR DEFINITION

The formalism introduced in the preceding paragraph is used for a formal definition of errors and the occurrence of errors within a sentence according to a syntax tree. In what follows, we will first introduce the error definitions and then discuss the treatment of syntactic and semantic errors within the language teaching system. For the purpose of natural language analysis application, the set of features \( F \) of a grammar \( G \) is divided into three disjoint subsets, \( F_{AG}, F_{SYN}, \) and \( F_{SEM} \). Now what is an ill-formed sentence? An ill-formed sentence with respect to a grammar is a string of words which cannot be derived by means of the production rules, i.e. it is an \( x \) such that not \( \text{[cat}(sg) \Rightarrow x \). This is surely not sufficient as a definition. We rather want to know by what manipulations within \( x \) we can obtain an \( x' \) such that \( \text{[cat}(sg) \Rightarrow x' \). The nature of these manipulations depends on the type of errors involved. For agreement errors, it is sufficient to change morphosyntactic properties of the words in the ill-formed sentence. For syntactic errors we have to add or to suppress single function words (prepositions or articles) or to permute words or word groups.

The definitions of 2.1. are extended to strings. Let \( a, b \in C(F, V)^{+} \), \( a = a_{1}...a_{k} \) and \( b = b_{1}...b_{k} \). Then

\[
a \sqsubseteq b \text{ iff } a_{i} \sqsubseteq b_{i} \\
a \sqsubset b \text{ iff } a_{i} \sqsubset b_{i}
\]
\[ a \rho b \iff a_1 \rho b_1 \]
\[ a \rho_G b \iff a_1 \rho_G b_1 \]

3.1. Agreement errors

Definition 3. Let G be a feature grammar over C(F, V \{ f | f \in F \cdot s \in C_G(F, V) \}) contains an agreement error situated in a and b according to G, iff not [cat(sg)] \Rightarrow s and there are x, y, z \in C(F, V), a', b' \in C(F, V), G \subseteq F_{Ag} s' such that xaybz \Rightarrow s and [cat(sg)] \Rightarrow xa'ybz and a'_G a and b'_G b.

Example 4. Consider the sentence

(5) * Der Kind spielt.

(5) cannot be analysed, because the syntactic rule for analysing the noun phrase (example 3) cannot be applied. The condition \( r = \emptyset \) is not satisfied. The efs representing "der" is c, the one for "Kind" is n2 (see example 1). unify(c, n2, r, e) gives \( r = \emptyset \) (see example 1) and

\[ e = \{ [\text{Art-cat}(def), \text{Gender}(neur), \text{Case}(\text{genitive}, \text{neg}(\text{genitive}))], \text{Number}(\text{plural}, \text{singular}) \}, [\text{Art-cat}(def), \text{Gender}(\text{fem}, \text{neur}), \text{Case}(\text{dative}), \text{Number}(\text{singular})], [\text{Art-cat}(def), \text{Gender}(\text{masc}, \text{neur}), \text{Case}(\text{nominative}), \text{Number}(\text{singular})]\} \]

\[ \text{Der} \quad \text{Kind} \quad \text{spielt} \]

By changing "der" to "das", we get \( c' = \{ [\text{Art-cat}(def), \text{Gender}(neur), \text{Case}(\text{or}(\text{nominative, accusative})), \text{Number}(\text{singular})]\} \) and unification (c', n2, r, e) with

\[ r = \{ [\text{Art-cat}(def), \text{Gender}(neur), \text{Case}(\text{or}(\text{nominative, accusative})), \text{Number}(\text{singular})]\} \]

Clearly, this definition does not yet indicate a strategy for analysing and correcting agreement errors. We need such a strategy because in German this analysis is very complex: the words are morphologically highly ambiguous. There are 24 different definite articles (4 cases, 3 genders and 2 numbers) but there are only 6 different forms for them all, each of which can have between 2 and 8 interpretations (or meanings). In the same way, every noun has up to four different forms which can have 8 different morphosyntactic meanings (four cases and two numbers). Adjectives are even more ambiguous, because there are (at least) 4 different declensions depending on their context within a sentence, i.e., on whether they are preceded by a definite or an indefinite article, by no article or by a negation. Our grammar contains these four declensions, i.e., \( 4^3 \times 2^4 \) ( \(- 96\) ) adjective meanings and only 5 forms for them (ending by "en", "en", "em", "er", "es"). But the case and number of a noun phrase in a sentence depend on the verb, since a verb takes a certain case and determines the number. Hence an error in the number of a noun phrase could also be an error in the number of the verb. As we have already seen, when two elements of a phrase do not agree, there are frequently several possible ways of analysing and explaining the disagreement (the set \( e \) of unification(a,b,r,e) is always large). In order to fully explain errors to a student, we will frequently use all these disagreement results. Consider again example 2. The noun phrase *der Kind* cannot be unified and we want to explain to a student why. In the above example, three different analyses have been found. It depends on the context within a sentence which explanation is the right one. We have found that case filtering gives plausible explanations. In German, verb complements have cases. Hence, for any noun phrase in a sentence, there is an expectation as to the case. Consider the following sentences:

(5) *Der Kind spielt (The child plays).

(6) *Er gibt der Kind Milch (He gives milk to the child).

(7) *Sie kennt der Kind (She knows the child).

In (5), *Der Kind* is the subject of the sentence and the expected case is the nominative. Case filtering gives the third error analysis: disagreement in the gender, since *der* is masculine whereas *Kind* is neuter. In (6), *der Kind* is the indirect object of the sentence and the expected case is the dative. Case filtering gives us the second error analysis:
disagreement in the gender, since *der* is feminine whereas *Kind* is neuter. In (7), *der Kind* is the direct object of the sentence and the expected case is the accusative. By case filtering, we find that *der* cannot be accusative. Let us point out that this strategy does not presuppose that the student always knows the case. First it is only applied in order to choose an interpretation of what case or number or gender the student had applied when two elements inside of a noun phrase (i. e. determiner and noun group or adjective group and noun) could not unify. This is a situation where anyway the student had not really known how to form the correct case of one of the components and moreover since no clear analysis can be performed the erroneous noun phrase cannot have any case. On the other hand, when the student does not know the case of a verb (or preposition) but is able to form a correct noun phrase given a case, the error analysis is straightforward because it finds a disagreement between the case of the noun phrase and the case required by the verb (see example 5).

Example 5.

```
  [cat(np)  [cat(np)]
   ---------  ---------
  [cat(np)]  [cat(np)]
    ---------  ---------
  [cat(nom)] [cat(nom)]
    ---------  ---------
  der          Schüler
    .........  .........
  Lettner      enwiser
    .........  .........
  den          Schüler
```

The problem inside of noun phrases comes from the fact that there are three (or four) features which can have wrong values (gender, number, case or context for adjectives) whereas between a verb and its complement only one feature can be wrong (the case).

The most obvious strategy for analysing agreement errors consists of placing an error as high as possible in a syntax tree. Hence "den Schüler" is first analysed as a correct noun phrase in accusative case. The sentence structure rule

\[ vp \rightarrow [verb, case(c)] np[\ldots, case(c')] \]

\[ \text{unification (c, c', r, e)} \]

tries to unify the expected case, required by the verb, c (which is accusative) with the current case c' of the noun phrase (which is dative) and an appropriate error message is generated since they do not agree. But this procedure can be eliminated in the following situation. Take the sentence:

(8) * Der Götter zürnen (The gods are angry).

unification(a, n, r, e) gives

\[ r = \{ [Art\text{-}cat(def), Gender(masc), Case(genitive), Number(plural)] \} \]

\[ e = \{ [Art\text{-}cat(def), Gender(fem, masc), Case(genitive), Number(singular, plural)], [Art\text{-}cat(def), Gender(masc), Case(nomulative), Number(singular, plural)] \} \]

*Der Götter* is the subject of the sentence and the expected case is the nominative, *der Götter* is genitive plural and this would be the error signalled (disagreement on the case). But this analysis is not at all plausible. It is very unlikely that a student should try to construct a genitive plural, which is a "difficult" case, when the nominative is required., which is the "easiest" case. People make errors in order to make their lives easier! Hence, the strategy of analysing errors as high as possible is not applicable when a subject noun phrase, which should be in the nominative case, could be analysed as having another case whereas parts of it are in the nominative. Now; we have seen, that in the definition of unification even when the unification is successful, the set of nonunifiable elements is produced. Besides computational issues, because the algorithm runs only once through the lists, this set is very useful when a noun phrase already analysed, such as the one in our example, has to be reviewed in order to find a possible disagreement between its parts. Case filtering of the disagreeing interpretations gives us the correct error analysis: disagreement in the number, since *der* is singular and *Götter* is plural.

During a number of test sessions essays of the system, its explanations of agreement errors have always turned out to be plausible.

3.2. Syntactic Errors.

We presuppose that all the words of the sentence to be analysed belong to the dictionary of the language or are formed by concatenation of a verb prefix and a verb. We distinguish between low level and high level syntactic errors. Low level syntactic errors involve the omission or addition of functional words such as articles or prepositions. High level syntactic errors involve the permutation
of groups of words. High level errors are mostly due to non-application of obligatory transformational rules or to application of the wrong rules, usually derived from the native language of the student. Hence high level syntactic errors are transfer or overgeneralization errors. In (Schuster 1986) this relationship between errors made by second language students and the grammar of their first language is systematically used for handling errors. As pointed out in the introduction, our approach is rather to use transfer rules as supplementary rules. And it seems to us that Schuster's idea is only applicable when one can have at hand analysers of both of the languages, which are based on the same grammar model and apply it in the very same way.

Definition 4. Let \( G \) be a feature grammar over \((F, V), s \in C(F, V)^*\). Then \( s \) contains a syntactic error iff \( \text{cat}(sg) \Rightarrow s, s = a_1...a_n \) and there is \( t = b_1...b_m \in C(F, V)^* \) such that \( \text{cat}(sg) \Rightarrow t \) and \( \{a_1, ..., a_n\} \cap \{b_1, ..., b_m\} \neq \emptyset \) and \( \{a_1, ..., a_n\} \neq \{b_1, ..., b_m\} \) contains only prepositions or articles; i.e. \( \forall x \in \{a_1, ..., a_n\} \Rightarrow \{b_1, ..., b_m\} \) is \( \text{cat}(\text{prep}) \Rightarrow x \) or \( \text{cat}(\text{art}) \Rightarrow x \).

A syntactic error is low level iff

\[
\begin{align*}
& s = a_1...a_k...a_{k+1}...a_n \text{ and} \\
& t = a_1...a_k...a_{k+1}...a_n \text{ (addition) or} \\
& t = a_1...a_k...a_{k+2}...a_n \text{ (suppression).}
\end{align*}
\]

A syntactic error is high level iff \( s \) trans \( t \) or \( \exists x, y, z \in C(F, V)^*, \exists a, b, c \in C(F, V) \) such that \( \text{cat}(sg) \Rightarrow xyz \Rightarrow t \) and \( xbyz \Rightarrow s \).

Example 6.

(9) *Er gibts an Marie einen Brief (He gives a letter to Marie).

The analyser tries to analyse the sentence with the rule

\[
\begin{align*}
& \text{cat}(\text{vp}) \rightarrow \\
& \text{cat}(\text{verb}), ..., \text{obj-case1}(c1), \text{obj-case2}(c2)) \\
& \text{cat}(\text{np}), ..., \text{Case}(c1'), ... \\
& \text{uni-value}(c1, c1', r1, e1) \\
& r1 \neq \emptyset \\
& \text{cat}(\text{np}), ..., \text{Case}(c2'), ... \\
& \text{uni-value}(c2, c2', r2, e2) \\
& r2 \neq \emptyset
\end{align*}
\]

because the verb "geben" requires two objects, the indirect one in the dative case, the direct one in the accusative case. The rule cannot be applied because "gibt" is not followed by a noun phrase (np) but by a prepositional phrase (pp). Syntactically (9) could be analysed by the rules

\[
\begin{align*}
& \text{cat}(\text{vp}) \rightarrow \\
& \text{cat}(\text{verb}), ..., \text{pobj}(p), \text{obj-case2}(c).) \\
& \text{cat}(\text{pp}), ..., \text{prep}(p), ... \\
& \text{cat}(\text{np}), ..., \text{Case}(c'), ... \\
& \text{uni-value}(c, c', r, e) \\
& r \neq \emptyset
\end{align*}
\]

and

\[
\begin{align*}
& \text{cat}(\text{pp}), ..., \text{prep}(p), ... \rightarrow \\
& \text{cat}(\text{prep}), ..., \text{prep}(p), \text{Case}(c), ... \\
& \text{cat}(\text{np}), ..., \text{Case}(c'), ... \\
& \text{uni-value}(c, c', r, e) \\
& r \neq \emptyset
\end{align*}
\]

if there existed another verb taking the preposition "an".

Low level syntactic errors are systematically treated by introducing error rules of the following form. For suppression, there is the rule

\[
\begin{align*}
& \text{cat}(\text{prep}), ..., \text{error}(\text{missing-preposition}), ... \rightarrow e \\
& (e \text{ is the empty word})
\end{align*}
\]

and the analogue for determiners. When this rule is applied, the second syntactic rule can be used.

High level syntactic errors are treated by special error rules. We will show by giving two examples, one for the permutation of a group, and the other for a nonapplied transformational rule, how such types of errors are treated.

1. In German, adjectives precede the noun group, whereas in French they frequently follow it. The rule

\[
\text{cat}(\text{prep}), ..., \text{error}(\text{missing-preposition}), ... \rightarrow e
\]

\[ \text{[cat}(ng), ...] \rightarrow \text{[cat}(ag), ...] \text{[cat}(noun), ...] \]

which analyses a noun group ng like "blaue schone neue Auto", is accompanied by the error rule

\[ \text{[cat}(ng), ...] \text{error(syn)} ..., ... \rightarrow \text{[cat}(noun), ...] \text{[cat}(ag), ...]. \]

For the sake of clarity, we have simplified these rules by suppressing all terms relating to the morphological and semantic analysis and the properties of categories. The noun phrase \textit{das blaue Auto} would be analysed correctly as \text{[cat}(np), ...], whereas the incorrect noun phrase \textit{das Auto blaue} is analysed as \text{[cat}(np), ... error(syn)), ...]. In PROLOG, an error predicate \text{F} is produced which treats the error, i.e. produces an explanation.

2. In German, verb groups in the perfect tense are frequently split up. The auxiliary takes the place of the verb, and the participle goes to the end of the sentence, as, for example in:

\begin{align*}
(11) & \text{Ich habe dem Baby Milch gegeben} \\
& (I have to the baby milk given).
\end{align*}

French (and equally English) students of German might say

\begin{align*}
(12) & \text{*Ich habe gegeben dem Baby Milch.}
\end{align*}

This transformation rule, as well as its erroneous omission, is represented in PROLOG as follows:

\begin{align*}
\text{vp}(X, XE, \text{correct}) \rightarrow \\
& \text{verb}(X, X1, t, XH), \text{compls}(X1, X0), \\
& \text{eq}(X0, XH, XE). \\
\text{vp}(X, X0, \text{error(verb, part-perf)}) \rightarrow \\
& \text{verb}(X, X1, \text{perf}, XH), \\
& \text{freeze}(X2, \text{compls}(X2, X0)), \text{eq}(X1, XH, X2). \\
& \text{verb(ist}, \text{Y, pres}, 0), \\
& \text{verb(ist}, \text{Y, perf}, "gefahren").}
\end{align*}

Again, this description has been simplified in order to make clear how these transformation rules function in PROLOG. freeze is a predefined predicate of PROLOG II (PrologIA 1985). freeze(X, P) delays the evaluation of P until X takes a value. compls analyses the verb complements of the sentence. The order of the sentence parts is produced by the equations between them (predicate \text{eq}). The verb phrase is analysed as

\[ \text{vp("habe","dem","Baby","Milch","gegeben".nil, nil, correct)} \]

where the composed verb "habe gegeben" is described by

\[ \text{verb("habe",X1, X1, Tense(perfect), "gegeben").} \]

The complements are analysed by

\[ \text{compls("dem","Baby","Milch".X0, X0).} \]

The underlying transformational rule which moves the perfective participle "gegeben" to the end of the verb phrase (and of the sentence) is performed by the PROLOG unification expressed by eq(XO, XH.XE), where XH = "gegeben".

3.3 Semantic errors.

The only type of semantic errors on which we have been working so far concerns the violation of semantic restrictions on verbs and their complements.

\textbf{Definition 8.} Let G be a feature grammar over C(F, V), \( \rho_{F} \in F \cdot s \in C_{\Sigma}(F, V)^{s} \) contains a semantic error between a and b according to G, iff not \([\text{cat}(sg)] \Rightarrow s \) and there are x, y, z \in C(F, V)^{s}, a', b' \in C(F, V), G \subseteq F_{\text{Sem}} such that \( xaybz \Rightarrow s \) and \([\text{cat}(sg)] \Rightarrow x'ay'b'z \) and \( a' \rho_{G} a \) and \( b' \rho_{G} b \).

Note that this definition is identical to the definition of agreement errors with \( F_{\text{Agr}} \) replaced by \( F_{\text{Sem}} \).

The semantic relationships between verbs and their complements are all described by semantic features. Semantic features have the property that there is a hierarchical relation (called sup) on their value sets. Figure 4 shows a part of the semantic network which can represent semantic features and their relations.
Nouns, noun groups and noun phrases are all characterized by the semantic feature sem-char, verbs are characterized by the features subj-sem, obj1-sem, obj2-sem and pobj-sem (at least). There are other semantic features for verbs concerning the character of the event the verb describes and its meaning. The first one is described by the feature event-char with the values durative, perfective, resultative, punctual, etc., the second one by signification which has the values static, dynamic or action. But within this paper, we are not concerned with errors on tenses and temporal or conditional conjunctions, for which event-char and signification are important. This aspect of verb semantics will hence not be pursued any further here.

Consider the following sentence (13) and its analysis in figure 5.

(13) The book works.

The appropriate grammar rule has the form

\[
\text{cat}(sg), \ldots \rightarrow \text{cat}(np), \ldots, \text{sem-char}(s1), \ldots \\
\text{cat}(vp), \ldots, \text{subj-sem}(s2), \ldots \\
\text{unify-value}(s1, s2, r, e) \\
r \neq \emptyset
\]

The feature value network is also used to explain the error. We feel it is not sufficient to tell the student something like "A book is not human". Instead the system tells him "A book is a written object and objects are usually not human, but the verb "to work" requires a human subject. In (Weischedel and Sondheimer 1987), relaxation of classes (e.g. by personification or by metonymy) is treated by meta rules for semantic errors. Hence in their system, a sentence like

(14) My doll is weeping.

would be considered to be ill-formed. We have incorporated the relaxation of semantic classes into the network (see figure 4). One of the subclasses for human is personified and another one is place standing for humanized places, so that sentence (15) can be analysed as well-formed.

(15) The third floor has protested.

It is clear that for our application, it is important to analyse sentences (14) and (15) as correct sentences because the relaxation of classes is not based on a failure of semantic rules. It is even important for language learning to study how facts are expressed in the foreign language, especially when they are expressed differently in the mother tongue of the student. "My car drinks gasoline" is typical of English and is not expressed by the same verb in German ("Mein Auto fagt mit Benzin") or in French ("Ma voiture roule l'essence").

4. PERFORMANCE OF THE SYSTEM

The system described so far has been realized in PROLOG II. It is integrated into a larger language teaching system for German (Schwind 1990). We
have tested it with French students, adults and children but not in a sufficiently broad framework for having statistical data. Our system is based on a very complete grammar of German used simultaneously as an analyser and as a knowledge base consultable by the students. The syntax analyser is somewhat restricted. The reason is that we want always treat only with certain, well-defined grammatical phenomena. So we have one exercise where the student can form "free" sentences from verbs, nouns and adjectives. The idea is to study always only some very precise grammar problem and not all at the same time. The reason is that on the one hand the error explanations can become very long and complicated in the case of several errors in a sentence; on the other hand the real language learning situation where restricted and precise phenomena are studied and not all together at the same time is better reflected. Nevertheless some kinds of errors are always checked and corrected, namely, agreement of noun phrases, verbs, good preposition and word order (see examples 4 and 5). The vocabulary of the underlying grammar of our system comprises about 600 verbs, 500 nouns and 60 adjectives. The grammar base comprises all determiners (der, ein, manche, solche, alle einige, etc.), all prepositions and all sentence conjunct words like weil, und, wenn, sobald wie, ... (because, and, if, as soon as, ...). We have also some temporal and local adverbs (heute, morgen, hier, etc.) but this list is not complete. The description of the verbs is very complex, (more complex than is actually used): for every verb, not only all morphological information is given (comprising all tenses and the conjunctives) but all grammatical properties like case and prepositional complements together with semantic information about the nouns (or adjectives or sentences) allowed as complements. The grammar for the first type of exercise (formation of sentences out of a list of verbs, nouns, and adjectives) can analyse simple sentences in the form subject+verb+one or more complements included prepositional complements and sentences beginning with da$ (ich glaube da$ Karl kommt)". Moreover German specialties like detachment of the prefix can be treated (it is analysed and erroneous detachment or nondetachment is treated).

Parsing of a sentence (included the unification) is about 300 milliseconds (on MacintoshPB 180). The lexical analysis depends on whether we use direct dictionaries or radical dictionaries. In the second case it can be very long because for a given word (even when the category is well chosen by the parser which evidently is not always possible) all words of this category will be tried. In this case the cost of the lexical analysis depends on the size of the dictionaries. Since we are working now with rather important dictionaries, we have (for the syntactic analysis) a direct access dictionary, i.e. every word is a PROLOG identifier and the access is direct and does not more depend on the size of the dictionary. But this gives about 4 to 10 times the indirect dictionary since we create all forms of verbs, nouns and adjectives

5. CONCLUSION AND FURTHER RESEARCH

The results of our research and experiments can be summarized as follows:

1. Agreement errors can perfectly well be handled in a very general way, i.e. they do not have to be anticipated.

2. High and low level syntactic errors as well as lexical (semantic) errors can be satisfactorily dealt with but high level syntactic errors have to be anticipated, so that their treatment is not very general. Consequently, totally scrambled sentences cannot be analysed (but should they be?).

3. Ambiguously interacting errors present a serious problem. Consider the following example:

(16) "Er schreibt dem Hefl
He writes to the notebook.

The error could be analysed as a semantic error (schreiben requires a human dative object) or as a low level syntactic error (schreiben requires the preposition in). Obviously, there is no means of deciding which error the student has committed if there is no contextual information.

A very interesting problem not addressed in this paper is the treatment of the tenses of verbs in composite sentences containing temporal conjunctions. The choice of the tenses depends on

We used the vocabulary and the grammar of "Deutsch als Fremdsprache" elaborated by the cultural institutes teaching German
the conjunction and on the meaning and the aspect of the two verbs involved. Errors of this kind can also be defined within our framework (by means of feature grammars) but they are no longer classifiable as "syntactic" or "semantic" because they involve the tense, which is more a syntactic property, as well as the meaning and the aspect of the verbs, which are semantic properties. We are at present working on these problems.

REFERENCES


TELL INTO THE MAINSTREAM CURRICULUM

June Thompson, CTI Centre for Modern Languages, University of Hull, Hull HU6 7RX, UK
email CTI.Lang@hull.ac.uk

ABSTRACT

This paper will outline how Technology Enhanced Language Learning (TELL) has benefited from
national, government-funded initiatives which aim to introduce and integrate computer-based materials
throughout the UK higher education sector. It will
describe two major national initiatives and their
relevance to language learning; it will demonstrate
how expertise in software development is being
encouraged among language teachers, and will place
the UK initiatives in a wider, international context.

THE COMPUTERS IN TEACHING INITIATIVE (CTI)

As a result of the Nelson Report in 1982, which
recommended the wider use of information technol-
yogy throughout the UK higher education sector, the
Computers in Teaching Initiative (CTI) was estab-
lished by the Universities Funding Council in 1984. A
total of 139 projects were funded, in various disci-
plines, aimed at exploring the potential of computer
assisted learning (CAL) in UK universities. Although
few of the projects resulted in finished software which
could be used by students, the experience of this first
phase of CTI convinced the funding council of the
desirability of setting up national CTI centres, for a
wide range of academic disciplines, to collect and
disseminate information and to build up a core of
expertise with the emphasis on integrating CAL into
the academic curriculum, rather than on the technol-
ogy itself.

Thus, the CTI Centre for Modern Languages
was established at the University of Hull in April
1989, initially a three-year project. The remit of the
Centre was "to promote and encourage the use of
computers in the teaching of Modern Languages in
higher education." All the CTI Centres were required
to produce and disseminate regular newsletters, to set
up a database of available software in their discipline,
and to visit university departments, both to discover
the range and extent of computer-based activities
already taking place, and to introduce non-users to the
possibilities and potential for integrating the use of
computers into their teaching.

PUBLICATIONS

In November 1989, the CTI Centre for Modern
Languages issued the first of its publications under the
title ReCALL. Since that time the ReCALL journal has
been issued twice a year, a Newsletter is issued three
times a year, and the ReCALL Software Guide is
currently in its third issue. These publications have
been issued free of charge, over the last five years, to
all Modern Languages lecturers in the UK higher
education sector.

The Software Guide lists over five hundred
items which may be used in language learning,
focusing mainly on users in the higher education
sector. It gives a brief description of the software,
details of hardware requirements, the approximate
price and various sources of supply. It does not
attempt to give an evaluation of the software as, in our
experience, much depends on the target user and the
way in which the software is integrated into the
curriculum. Evaluation is therefore covered separately,
by means of software reviews published regularly in
ReCALL. Copies of reviews published elsewhere are
also collected at the CTI Centre in Hull and made
available to language lecturers who use our enquiry
service.

ENCOURAGING INTEGRATION
OF LANGUAGE LEARNING SOFTWARE

Over the first three or four years of its existence, the
Centre collected a large archive of relevant informa-
tion, through visits to university departments, receiv-
ing visitors at the Centre, attending national and
international conferences, and through a range of
other contacts. We also built up a comprehensive
library of language learning software for higher
education, mainly through donations by developers
and suppliers, who have become increasingly aware of
the important role played by the Centre in promoting
the use of these materials.

On a typical visit to a university Modern
Languages Department or Language Centre, we will
take with us five workstations and between eighty and
a hundred language learning programs, including
CD-ROM-based materials and interactive video. The
day normally starts with a short presentation outlining
the kinds of materials available and the possibilities
for their use in the context of the particular depart-

ment concerned. We place considerable emphasis on the role of teachers, whether they aim to use the materials in the classroom or in self-access mode. We attempt to allay the fears of some teachers that computers will replace them or diminish their status, and to demonstrate some of the interesting and exciting ways in which computers can genuinely enhance the learning process.

After a general discussion period, the programs are available for teachers to try out for themselves and staff from the Centre are available to discuss questions relating to implementation and integration, or to help lecturers who may be unfamiliar with computers, of whom there are still a sizeable number.

A similar pattern is followed at the CALL Awareness Days which are held regularly at the University of Hull. We are careful to obtain feedback from these events, as part of our self-evaluation process, and this feedback indicates that the formula outlined above is successful and much appreciated by teachers.

As I stated earlier, we do not attempt to allocate grades or ratings to language learning software. However, we have in the course of our work identified a number of programs which are generally found to be useful by a significant number of users; given the limitations of time and space, therefore, we tend to direct new users towards this group of material although, of course, new software is continually being developed and we make sure that our information is as up-to-date as possible.

As well as identifying useful software, we are able to identify examples of good practice in the use of CALL. In particular, we soon became aware of the significant gaps in the provision of suitable software for use in universities. Much of the commercially-produced material is aimed at beginners or non-specialist linguists, and this material is itself of varying quality. Very little software is produced for specialist linguists at the more advanced levels.

HARNESSING EXPERTISE

From the start we were aware, at the CTI Centre for Modern Languages, that there were a number of academics, scattered amongst various universities, who were working in the area of CALL, either as researchers or as developers of computer-based materials which they were using with their own students. However, a survey which we conducted in 1989 showed that these academics represented only a tiny minority of the ten per cent of language lecturers who were using computers in their teaching. A further survey in 1992 showed an increase in users to thirty per cent: it is highly likely that the activities of the CTI Centre for Modern Languages contributed towards this increase, though it is evident that there is still a long way to go before CALL is integrated into the mainstream language teaching curriculum.

This small core of experienced CALL researchers and developers, identified in the early days of CTI, formed the basis for a series of Expert Seminars, organised by CTICML under the general title of Educational Technology in Language Learning. The seminars aimed to promote discussion between software developers and language acquisition theorists, and also to attempt to set an agenda for the future direction of CALL in the UK. The major elements highlighted in a discussion paper produced by a working group set up on behalf of the seminar group were (a) the need for a major programme of software development, and (b) the need for a major programme of research into how languages are learned, and how beneficial (or otherwise) CALL can be. The group also emphasised the need for the continued existence of the national CTI Centre for Modern Languages, to provide the necessary information and co-ordination for these programmes.

When, in March 1992, the Universities Funding Council launched its major Teaching and Learning Technology Programme (TLTP), with funds of approximately £8 million available throughout all disciplines, we felt that both the objectives (a) and (b) above might be achievable, provided that substantial funding could be made available. Proposals were requested both from institutions and from consortia, from all academic disciplines including the major sciences. Emphasis was placed on the production of deliverables and their implementation within institutions. Institutions were expected to make their own contribution, especially through the provision of hardware for student use. The timescale for submitting proposals was extremely short.

Working closely with the core group of academics mentioned above, some of whom were already developing software, the CTI Centre for Modern Languages decided to put together a proposal for a consortium to develop language learning materials, with CTICML as the lead site. In the event, we had rather more volunteers to join the consortium than could reasonably be included in a coherent proposal. The final number consisted of thirty-seven: fifteen developers and twenty-two affiliate sites who would be carrying out formal evaluation.

THE TELL CONSORTIUM

The proposal was successful, carrying an award of approximately £1.3 million over three years. With a deliberate emphasis on the importance of language learning rather than technology (and a logo designed
to reflect this emphasis) the consortium was named TELL (Technology Enhanced Language Learning). The remainder of this paper will describe how the consortium operates, the scope of the materials being produced, and the measures being employed to integrate them into the mainstream language learning curriculum.

THE NATIONAL CONTEXT

The Single European Market and greater European integration generally has sharpened an awareness in Britain of the part language learning has to play in terms of economic success, as well as cultural commerce. In response to this need, most UK higher education institutions have implemented or are implementing a 'languages-for-all' policy, by which is meant the making available of language learning to all those who require it: non-specialist students, learners with special needs, advanced honours students, professional updaters. Within the general explosion of the student population, there is a second explosion in the number of language learners.

The challenge that this poses to higher education language teachers can be addressed through retraining of staff, an increase in teaching provision, an increase in self-access learning (already a part of the language learner's experience), and an increased exploitation of educational technology.

MEMBERS OF THE CONSORTIUM

Developers are working for the TELL consortium at fifteen universities in England, Scotland, Wales and Northern Ireland. These are the Principal (developmental and reception) sites.

A further twenty-two universities act as affiliate or reception sites. Their main function is to carry out the rigorous evaluation programme which is a major element of the consortium's work.

AIMS AND OBJECTIVES

The aims of the TELL Consortium, as defined in the original TLTP proposal are:

- To make the teaching and learning of modern languages within UK higher education more productive and efficient by harnessing modern technology.
- To integrate the use of technology into current teaching and learning of modern languages within a 'languages-for-all' policy.
- To concentrate on the major European languages (French, German, Italian, the Hispanic languages), while recognising that the methodologies will be transportable to other European and non-European languages.
- To address the various levels of learner competence (specialist and non-specialist learners, beginners and advanced students), the various specific learning targets (e.g. German for scientists) and the various constituencies (full-time, part-time students, continuing and professional education).
- To enhance cost-effective, open-learning opportunities by using a range of computer-based technology, ranging from text-centred applications to multimedia (both integrated multimedia on a single workstation and discretely delivered multiple media).

The objectives are:

- To develop new courseware, using existing and widely available authoring tools.
- To exploit, modify and enhance existing courseware.
- To improve the quality of materials produced through developmental testing.
- To document the lessons learned from the implementation of developed courseware at reception sites.
- To disseminate courseware to the system as a whole.
- To increase the confidence of UK higher education language teachers as users and adapters of CALL courseware.

Various measures are taken in order to achieve a manageable, coherent operation:

- The Consortium concentrates on a limited range of major languages.
- It adopts a broadly common approach to language pedagogy (communicative, functional, contextualised).
- A commonly-agreed ‘house-style’ for new materials has been established.
• The Consortium consists of interlocking developmental groups, working in areas of learner competence from beginner to advanced student.

• The whole project is subject to central evaluation by an evaluator external to the developmental groups.

• The central management structure, led by the CITCML office, handles finance, publication, reporting, training and dissemination.

• The Consortium has adopted a common technical approach, involving cross-platform transportability between Macintosh and Windows environment, with the starting point being Apple LCII and 386 machines with Windows and soundboard; a limited range of authoring tools, such as HyperCard, Toolbook and Authorware are being used.

THE THREE-STRANDED APPROACH

The consortium has adopted a three-stranded approach which aims at both comprehensiveness and flexibility, a mixture of ‘ready-made’ core materials at beginner level and more general material allowing the teacher and learner to exploit resources and tools at appropriate levels. The three strands are complementary and are designed to be accessible on the same platforms.

Courses
The primary focus of the Courses Team is the need to cater for the non-specialist language learner in French, German, Italian and the Hispanic languages; however, the materials will benefit a wide variety of learners with different ability levels and varying needs, as well as a range of preferred learning styles. It is therefore, multimedia, non-sequential and modular in structure. Later outputs will include levels up to Intermediate. Courseware is designed to lie alongside existing teaching materials in other media, ensuring an integrated approach. The Courses team is led by Dr Paul Bangs (South Bank University); other members are Professor Graham Davies (Thames Valley University), Dr Gordon Burgess (University of Aberdeen), Dr Michael Harland (Glasgow University), and Dr Miranda Stewart (Strathclyde University).

During the second phase of the project, the Courses team incorporated an additional small group who are producing CALL materials to complement a course that it is already designing for French for Scientists and Engineers at both beginners and post Higher/A-level. It is intended to extend the methodology to other major European languages. This project, funded by the Nuffield Foundation, is led by Dr Robin Adamson (University of Dundee) and Professor Samuel Taylor (University of St Andrews).

Resources
The second strand of the consortium’s programme seeks to provide access to language learning resources rather than specific courses with defined aims and objectives. It is based substantially on the exploitation of authentic foreign language materials (text, audio or visual). The Resources team is both devising routines with which language lecturers can access and exploit such resources and producing a flexible set of lessons and practice exercises in grammar (including remedial grammar where necessary), lexis and pronunciation, for integration with ‘home-based’ material. The emphasis is on intermediate to advanced levels. There will be extension modules for sublanguages (business, science and engineering).

The Resources team is led by Mr Paul Hickman (La Sainte Union College of Higher Education, Southampton). Other team members are Dr Lawrence Wright (University College of Wales, Bangor), Ms Marie-Christine Hayet (UMIST), Dr Andrew Crompton (University of Nottingham), Mr Douglas Jamieson (University of Hull), and Dr Marie-Madeleine Kenning (University of East Anglia).

Language Handling Tools
This strand of the programme concentrates on making language-handling software exploitable by the language teacher and learner, through giving on-line access to translation aids and other tools (bilingual dictionaries, spell-, style- and grammar-checkers, thesauri, writing assistants). The products from this strand include translation courses, a student-generated grammar/vocabulary/idiom notebook, and a customisable on-line dictionary. Many of the materials are designed to integrate with those from the Courses and Resources strands.

The leader of the Language Handling Tools team is Professor Doug Thompson (University of Hull). The other members are the CAT/CALL team at the University of Coventry, Dr John Gillespie (University of Ulster), and Dr Christoph Zähner (UMIST).

IMPLEMENTATION AND MANAGEMENT

Management Structure
The lead site, the CTI Centre for Modern Languages at the University of Hull under the direction of Graham Chesters, is financially responsible for the
project and, with the advice of a Management Committee, contracts other sites to fulfil agreed aspects of the whole programme. June Thompson, Manager of CTICML, is responsible for co-ordinating the Consortium's activities and executing the decisions of the Management Committee.

The Management Committee includes the leader of each of the three teams mentioned above, as well as the external evaluator. In the first year of operation, the Management Committee was assisted by two reporting groups, the first with responsibility for pedagogic matters, the second with responsibility for technical matters. From Year Two, these groups were replaced by a smaller Standards Group, who work closely with the teams to ensure that agreed standards in all areas are adhered to. The Management Committee makes decisions on the distribution of funds and oversees matters of general importance such as negotiation with funding bodies, copyright and intellectual property rights, publishing, publicity, dissemination and co-ordination of activities across the three strands.

Evaluation
The management of evaluation is the responsibility of Dr Diana Laurillard, an experienced educational evaluator and head of a CALL developmental and research team at the Open University. At each production site, material is being developmentally tested with target students, using observation, computer monitoring, interviews, and, where appropriate, pre- and post-tests, to establish the qualitative improvements students achieve through their use of the courseware. The evaluation uses standard methodology and evaluation tools across all sites, using joint workshops to unify approach and integrate findings. This series of formative evaluation studies will be written up in the form of design principles for CALL for future work to build on.

At each reception site (including the production sites receiving others' materials), the integration of developed materials with existing courses will be evaluated to determine the quality of learning outcomes achieved by students, the optimal conditions for successful implementation, and the productivity gains brought about by the use of the new materials. Each reception site will receive all materials free of charge, and will receive limited funding for the staffing required to administer and carry out the data collection and analysis for that site. Site evaluation studies are coordinated by the evaluator, who is external to all developmental sites.

Dissemination
The project has immediate access to the constituency of higher education language teachers through the services of the CTI Centre for Modern Languages. Regular information about the progress of TELL materials is included in the ReCALL Newsletter which is received in all higher education Modern Languages departments.

Training is seen as an important aspect of dissemination and implementation, and the first of a series of workshops was held at the University of Hull on 14 May 1994. The workshop was entitled 'Using Computers for Teaching Translation' and introduced participants to the preliminary versions of a selection of courseware produced by the Language Handling Tools team.

Under the terms of the Teaching & Learning Technology Programme, one copy of all deliverables will be made available to UK Higher Education institutions, free of developmental costs. In order to make some of the materials available as quickly as possible, and to elicit as much feedback as possible from users, twelve programs were distributed to 130 higher education institutions in April 1994. Comments received as a result of this distribution will be carefully analysed and all possible improvements incorporated into the final versions, and into the various foreign language versions of each program. We expect to deliver final versions of the complete range of programs towards the end of 1995.

THE INTERNATIONAL CONTEXT

Both the major projects described above are, by definition, part of a national initiative, designed to improve the quality and efficiency of learning in the UK higher education sector. However, strong connections between the CTI Centre for Modern Languages and the wider CALL community have been in place throughout the existence of CTI. We make every effort to ensure that we are aware of relevant developments in Europe and further afield and, within the limits of our funding, to disseminate information and expertise as widely as possible.

A major step forward was the formal establishment of the European Association for Computer Assisted Language Learning (EUROCALL), on the occasion of the EUROCALL 93 conference, organised by CTICML at the University of Hull in September 1993. The conference itself attracted 260 participants from 30 different countries. The President of the new organisation, Professor Graham Davies, has long been recognised as a leading figure in the discipline, and, with the help of LINGUA funding from Europe, we were able to attract onto the Executive Committee leading academics from eight countries. Membership of EUROCALL offers language teachers access to the publications of CTICML and to its enquiry desk, as well as other benefits.
The ReCALL Newsletter has recently been mounted on the Gopher system, and it is hoped that electronic communication and perhaps even electronic delivery of materials will figure more prominently in our activities over the coming years.

In turn, we hope to benefit from the experience of developers and practitioners in other countries, and to participate in the major but exciting task of pushing Technology Enhanced Language Learning into the mainstream of the curriculum.

Enquiries about CTICML, TELL or EUROCALL should be addressed to Mrs June Thompson, CTI Centre for Modern Languages, School of European Languages and Cultures, University of Hull, Hull HU6 7RX, tel: 0482 465373, fax: 0482 473816, email: CTILang@uk.ac.hull.

NOTES AND REFERENCES:

1. Powell B, ‘Call in UK Higher Education: a Preliminary Survey by the CTI Centre for Modern Languages’ in ReCALL No. 2, May 1990, CTICML, University of Hull.


3. At this point, all the CTI Centres had come to the end of the three-year funding period envisaged at the start, and were maintained under a series of short-term funding arrangements, pending the change in the UK higher education system which merged the universities and the polytechnics under new Higher Education Funding Councils. In March 1994 the HEFCs accepted a recommendation that the CTI Centres should continue for a further five years.

4. TELL publication P003 Style Guidelines for Developers.

5. Relevant TELL publications to date include P001 Program Design Principles, P002 Evaluation Planning.

6. See Appendix for details of these beta-test programs.

APPENDIX

Preliminary TELL Products

Beta-test versions distributed to UK HE institutions for formative evaluation, April 1994

C001: Spanish Dialogues

Hardware: IBM with soundboard and fair-size RAM memory

Target language: Spanish

Level of language ability: Zero to faux débutants

Comments: To accompany existing course materials and provide contextualised practice of language used in ten basic situations common to most teaching contexts, and to enhance this practice with contextualised help and support.

R001: GramEx French

Hardware: IBM; Apple Macintosh

Target language: French

Level of language ability: A-level plus

Comments: GramEx intends to provide tuition and reinforcement in essential areas of grammar. The format of the exercises is essentially multi-choice gap-fill, but the application offers context-sensitive explanations to users, and will ultimately be accompanied by an on-line summary grammar. The items are presented in random order each time the application is run, thus enhancing the re-usability of the material. Errors are recorded and then represented in a ‘remedial run’ when the exercise has been completed.

R002: French Periodicals Database

Hardware: IBM

Target language: French

Level of language ability: A-level plus

Comments: This database of over 5,500 references to articles in the French weekly press, going back to 1984, aims to facilitate students’ ability to use the press as an information source in the study of contemporary French society. Once copyright clearance has been obtained, the system will be accompanied by text files of the articles themselves.

R003: Méli-Mélo

Hardware: IBM (DOS and Windows versions)

Target language: French (multilingual interface)

Level of language ability: Advanced

Comments: Méli-Mélo will support the study of French language structures, from morpho-syntax to discourse. It can be used to focus on questions relating to ordering of elements, and the functional significance of the ordering of elements when units are fitted into a wider linguistic context. Méli-Mélo comes with
a database of ready-to-use exercises, but teachers can also construct their own exercises, based on authentic documents or written from scratch. The program may be used as part of a teacher’s workbench of CALL applications and integrated into the teacher’s own approach.

R005: Oyex! and Fermin
Hardware: IBM with Teleste 1000 voice card or ASC D15 voice card
Target language: French*
Level of language ability: All levels
Comments: Authoring package for the use of voice cards, enabling voice cards to use conventional tape-based language laboratory materials and combine audio work with text-based CALL exercises. Oyex! is the student program, Fermin is the teacher program.

R008: GramDef French
Hardware: IBM
Target language: French*
Level of language ability: A-level plus
Comments: This application will enable students whose awareness of formal syntactic relationships is weak or non-existent to explore, expand and test their knowledge of such relationships within the framework of a series of relatively simple texts. It also aims to teach basic grammatical terminology in the target language, so that classroom and personal study of the foreign language can be pursued in the language, with little or no recourse to English.

T005: TransIT (German)
Hardware: IBM
Target language: German*
Level of language ability: 4th year university
Comments: A technical translation course using word processing, designed as a taught course or partially self-access. It aims to introduce advanced learners of German to the translation of a range of ‘technical’ languages and their equivalents in English.

T006: MetaText
Hardware: Apple Macintosh
Target language: French, German, Spanish
Level of language ability: Language specialists in second and final year
Comments: A translation aid which allows the student to translate a text in one window (textcard) into an empty window below it and to access and store vocabulary and grammatical data on datacards, thus using the program as an interactive notebook. It aims to develop translation skills and to improve the student’s retention and re-use of insights gained from the corrections of previous work which are incorporated into the datacards.

T007: MCQ
Hardware: Apple Macintosh
Target language: French*
Level of language ability: All levels
Comments: A multiple choice program which enables the tutor to set up a suite of questions to test grammatical knowledge of an appropriate level and complexity using a simple HyperCard-based Lesson Builder. The order of answers is randomised, and the sequencing of questions may be as well. The program scores each user and allows a printout to be produced. User and grammatical help facilities are provided online.

* Other language versions will be available in due course.
LANGUAGE IN ACTION
or learning a language by watching how it works

Michael Zock
LIMSI, Langage & Cognition
B.P. 133 - 91403 Orsay / France
zock@limsi.fr
FAX: 69 85 80 88

ABSTRACT

We discuss in this paper two versions of a system that has been designed to help students to learn the rules governing linguistic form (sentence generation). The shortcomings of the first version were due to the fact that learning, i.e. rule inferences was possible only on the basis of correlations between input and output: different meanings (conceptual input) yielding different forms (linguistic outputs). The problem with this kind of approach is that the final form depends not only on conceptual choices made at the input, but also on a number of linguistic and pragmatic choices made later on. In the newer version of the system we have tried to remedy these shortcomings: whenever the user provides some information, i.e. every time he makes a choice, he can see its consequences (syntax, morphology, or both).

All information is stated in a unique formalism: conceptual graphs. This being so, we have a simple, yet powerful means for showing directly on the screen the consequences of a choice. By visualising at each step the outcome of a choice we show how the different components (conceptual, pragmatic or syntactic) interact. Showing the dependencies we explain not only the functioning of the system (language), but we also simulate certain aspects of the process (generation in real time). In doing so we have transformed the initial black box into a glass box.

Key words: Computer assisted language learning, simulation based-learning, natural language generation, intermediate structure, visualization of dependency relations, black box vs. glass box, process vs. product

1 Problem & motivation of this work

Discourse generation is decision making under specific space and time constraints. While human performance is quite remarkable, given the complexity of these constraints, our understanding of the underlying process is quite deceptive. Despite our enormous amount of practice, we are hardly ever aware of the nature, the order, or the consequences of these choices.

That our understanding of the process remains so poor, is due to the number, the diversity and the interdependency of the choices. While it is easy to understand the relationships between a glass falling and its breaking (causal relationship), it is not easy at all to understand the dependency relationships holding in hierarchically organized systems like biology, society, or natural language.¹ In other words, viewing the functioning of the mind, hence, the functioning of natural language in similar ways as the functioning of a complex society (oligarchy).

The two systems are organized in a similar way: (a) problem solving is decomposed; the result is produced not by a superexpert, but by a team of specialists; (b) the different agents (components) contributing to the solution have a certain amount of autonomy; (c) the agents negotiate, that is, they do not only communicate their results and draw on them, but can also adapt their behavior to allow for accommodation of the results produced by the other components.

The advantage of such a hierarchical kind of organization are multiple: (a) freedom of processing: various orders are possible to reach the solution; (b) time-sharing: each agent can work on its own without having to wait for an order coming from a supervisor; (c) flexibility: information flow is bidirectional; (d) opportunistic planning: as information becomes available at different moments and in unpredictable ways, and since the different components can accommodate it, it is possible to have the different agents compete and to use the first result produced by any of them.

The major drawback with this kind of system, where everything is more or less interdependent, is that it becomes extremely difficult to see the dependency relationships, that is, it is hard to see what causes what, or what action has what outcome. This is quite

¹ One could view the functioning of the mind, hence, the functioning of natural language in similar ways as the functioning of a complex society (oligarchy).
Type of change | linguistic consequence | level
--- | --- | ---
Pragmatic: status | Paul aide Marie (Paul helps Marie) | Syntax: position of object
Marie -> new

Conceptual: tense | Paul l'a aidé (Paul has helped her) | Morphology: agreement with the object
present -> present perfect

Linguistic: verb | Paul lui est venu en aide (Paul has helped her) | Morphology: pronoun: lui, auxiliary: être
aider -> venir en aide

---

figure 1

---

figure 2
words, what causes what, or what action produces what effect can not be inferred readily in such complex systems. There are various reasons why natural languages are so hard to learn, to use and to explain:

1.1 A great number of interdependent choices or constraints has to be satisfied.

The figure 1 here below illustrates some of the dependencies holding between the different components (pragmatic, conceptual, linguistic). Suppose that the meaning to be conveyed were the following message: [HELP (AGENT: Paul, OBJECT: Marie)]. Suppose further that one performed progressively certain local changes. All these changes, be they pragmatic, conceptual or linguistic, may reflect not only in the final form, —the output (see figure 2: *Paul aide Marie vs. Paul l'a aidée vs. Paul lui est venu en aide*)— but also in the intermediate structure. For example, a concept may surface as a complex noun-phrase, as a noun or as a pronoun.

1.2 The result of a choice may be indirect.

The choice at one level may affect the range of choices or their outcome at an entirely different level (hence remote in time). For example, a pragmatically motivated choice like focus may constrain, i.e. reflect in morphology, in the lexical item, in syntactic structure, or concurrently in each one of them. If you look at the figures 1 and 2 you will notice that the difference of the personal pronouns (*la vs. lui*) is an indirect consequence of the verb choice (*aider qn vs. venir en aide à qn*). The underlying conceptual structure being the same in both cases: Paul helps Marie.

(a) Paul l'aide. pronoun = la
(b) Paul lui vient en aide. pronoun = lui

1.3 A choice may have several consequences.

For example, in French, a conceptual choice like *negation* may affect morphology (form) as well as syntactic structure (word order).

<table>
<thead>
<tr>
<th>Interaction morphology- syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A1) Parle-MOI d'elle ! V-OI-pp-OI</td>
</tr>
<tr>
<td>Tell ME about her !</td>
</tr>
<tr>
<td>(A2) Ne ME parle pas d'elle! neg-OI-V-pas-pp-OI</td>
</tr>
<tr>
<td>Don't tell ME about her !</td>
</tr>
<tr>
<td>(B1) Donne-le MOI V-DO-IO</td>
</tr>
<tr>
<td>Give it to ME !</td>
</tr>
<tr>
<td>(B2) Ne ME le donnez pas neg-IO-DO-V-pas</td>
</tr>
<tr>
<td>Don't give it to ME !</td>
</tr>
</tbody>
</table>

1.4 Synthesis requires information belonging to different levels or components.

This has at least two consequences: (a) words can not be synthesized in one pass; (b) words cannot be synthesized in a linear order: information becomes available at different moments, i.e. in a discontinuous way. This discontinuity is reflected in the fact that the relevant information is scattered all over. Hence, there can’t be a word-to-word, or, concept-to-concept processing. The synthesis of personal pronouns in French illustrates quite nicely this fact. The indirect object pronoun requires information outside of the referent (SPEAKER). See figure 3.

Last but not least,

1.5 The result of a choice may not be accessible to our senses.

It may appear at an intermediate level (for example, part of speech). In general, only the input (visual scene) and the output (text) are visible. What happens in between —the progressive synthesis of the final product (process)— has to be inferred. In short, which choice determines which outcome, or what determines what, is far from self-evident. (Figure 4)
Synthesis of the indirect object

**Figure 3**

**INPUT:**
visual scene

**OUTPUT:**
text/sounds

**Processes**

- Conceptual
- Linguistic
- Motor

**Processes**

- Determination of concept
- Formulation of discourse
- Syntax
- Morphological operations
- Phonological operations

**Processes**

- Visual, auditory, motor
- Part of speech
- Morphological operations
- Phonetic operations

**Processes**

- Lexical/functional (message -> word)
- Synthesis functions: object, type of complement

**Processes**

- Speaker
- Listener
- Listener

**Processes**

- Subject
- Direct Object
- Indirect Object

**Processes**

- Speaker
- Listener

**Figure 4**
2 Our goal: visualize the process and provide a causal analysis of linguistic performance.

Given the difficulties mentioned here above, it would be useful to have a system which could visualize the nature, the implications and the adequacy of the various choices that one has to make when producing natural language. Put differently, it would be nice to have a system that could model the kind of functional decomposition given in figure 4. The idea is to allow the user to inspect where and how a choice reflects. The procedure would be the following. Every time the system receives some information, that is, every time the user makes a choice, he can ask the system to show him where this information reflects. This is precisely what we've tried when we started to build SWIM an interactive sentence generator for French.²

2.1 Assumptions

The major goal of the system is to provide a "natural" environment for learning in an intelligent and meaningful way to produce sentences in French. Let me briefly explain what I mean by using the attributes natural, intelligent and meaningful.

The major difference between a natural and an artificial environment (school) lies in the learning objectives and in control. The primary goal in a natural setting is communication. Language being an indirect means of achieving goals, people are interested in finding out what linguistic means are instrumental with regard to their discourse goal. In other words, people learning a language are interested in getting a message across, not in learning the grammar rules. Secondly, most institutional environments allow only re-actions, that is, students are not meant to act (to talk) unless asked to (extrinsic motivation), while people living in a natural environment act whenever they feel that there is a need for it. Obviously, the moment of action and the nature of the goals in whose realisation people are most interested, are the ones they have defined themselves (intrinsic motivation).

The label “intelligent learning” is used to draw a distinction between insightful learning and rote learning. The former (learning by observation) requires the learner to be able to describe and to compare a set of data, and draw the necessary conclusions.

Meaningful learning refers (a) to the integration of new knowledge to existing knowledge and (b) to relevancy: students quite often know what they do not know. For example, they know what they want to say, but they lack the means of saying it (words and sentence structures). As people’s questions are generally related to their goals, their messages are necessarily meaningful. It’s probably only in school, or in very mundane settings that people ask questions whose answer they already know, or about which they do not care.

Language generation is the process of making decisions at various levels (pragmatic, conceptual and linguistic) in order to reach a specific goal. In other words, to be meaningful means to make a choice and to communicate it. The newcomer to a society has thus to learn what the choices are, and what effects they produce. As the final product varies as a function of different choices, it is important to learn how each one of them affects this form:

Conceputal choices: different meanings generally yield different forms (He sings vs. they sing)
Linguistic choices: a given meaning can be expressed by different words or syntactic forms (paraphrase, synonymes: help vs. assist);
Pragmatic choices: different linguistic means serve different discourse purposes, i.e. different forms are used to achieve different goals (active vs passive voice; simple vs. complex clause).

The goal of our system is not only to show how linguistic output (form) varies as a function of the conceptual input, but also to show more generally what the consequences of a given choice are, regardless of whether this action reflects in the output or only in the intermediate structure (lexical, syntactic, morphological).

---

² The acronym SWIM stands for "See What I Mean"? It has been implemented by A. Larouci in LeLisp on a Macintosh.

³ Please note, that the system is concurrently built for French, Arabic and Russian.
To sum up, in order to reach the goals defined above, we’ve tried to reason from the learner’s point of view (learner-centered approach). First and above all, the student is fully integrated into the process. He is given most of the control: he asks the questions and the system answers them. In doing so the user actively explores an environment by asking meaningful questions. If learning is searching, than the questions of what to search for, and how to search (strategies) arise. SWIM is an environment that allows a search space to be incrementally built and to be systematically explored (hypothesis testing). The user thus acts like a child, or an adult learning a new language in a natural environment, while the system acts as an expert (native speaker, parent, teacher) trying to help him.

By asking any of the following questions and by reasoning, the user is expected to gain enough insights about the language in order to produce well formed sentences in French.

1. How does one say < idea >?
2. Can one say < linguistic form >?
3. How should one say < idea >?
4. Why does one say < form-1 > and not < form-2 >?
5. What would happen if < conceptual modification >?
6. What would happen if < syntactic modification >?
7. What would happen if < word y > instead of < word x >?
8. What is the difference between <form x> and <form y>?

3 Description of the system

The dialogue is initiated by specifying the communication mode. Let us suppose that the user had started with the first question: How does one say < idea >? In that case the system presents a menu (type hierarchy) from which the user has to choose:

Speech act: statement, question, command
Actions: taxonomy of verbs
 tense: present, past, future, etc.
Discourse objects: taxonomy of nouns
 number: singular, plural
 communicative status: definite, indefinite, etc.
 attributes: taxonomy of attributes, etc.

By choosing (clicking) specific values from a set of alternatives, the user tells the system what he wants to say. As the dialogue develops, the system builds the underlying meaning in the form of a semantic network. It then invites the user to try to express this meaning, after which it will output its own form. (figure 5)

As one can see from Figure 5, the screen is divided into five parts. The large window at the top represents the underlying meaning of the message a student is trying to convey. The next two windows contain respectively the user’s attempt to express this meaning and the system’s version. Possible mismatches between these two versions are highlighted on the screen. This allows the system to draw the student’s attention to errors. Actually, by looking at Figure 5 you can see that the student made a mistake in the verb agreement “regardé”.

Finally, the two windows at the bottom represent, from left to right, a snapshot of the system’s memory of meaning and the user’s memory of form. The former is a device to display, hence to recall the underlying conceptual representation of a specific sentence chosen from the memory of form window, whereas the latter is an incrementally built database (trace) of all the sentences encountered so far.

The idea behind this separation is to allow the user to make a contrastive analysis of meaning and form between two sentences. Choosing a sentence in the memory of form window gives a representation of the sentence’s meaning in the memory of meaning window. By comparing the surface form and the underlying meaning of two sentences, the user can appreciate the relationship between meaning and form. The critical feature, the one that is responsible for the difference of form, is highlighted by the system. In our example it is the value “singular.”

---

4 For more details concerning the interface controlling the conceptual input (What to say component), see Zock, 1991.
5 See communication mode 8: What is the difference between <form x> and <form y>?
Past this point, the user has various options: either he continues in the basic communication mode (*How does one say *<i>idea</i>*?), or he changes the kind of question he wants to ask. Let us suppose that he wants to build a completely different sentence. In this case, he could either go through the whole routine, which is quite cumbersome, or he could perform the conceptual changes directly on the graph. Obviously, this latter method is much faster.  

---

Actually, everything you see on the graph, except deep-case relations, is considered by the system as a variable whose value may be changed directly by the user. In order to do so, he clicks on any of the shown attributes (verb, tense, mode, etc.) and chooses a new value. For example, if he clicks on the verb "regarder" (to watch) the system answers by displaying a list of candidates from which we must choose. By clicking on the feature "present", the system shows different values (present, past, future, etc.) of the variable "tense", etc. Everytime the system is given some piece of information it will change the meaning representation accordingly and, if asked, output the corresponding form.

For example, if one started from sentence (a) asking the system to change the number of the direct object from "singular" to "plural" it would
Figure 6: Meaning changes by direct manipulation of the conceptual structure

Figure 7: Changes of the syntactic structure
produce (b). If one asked to change the tense from "past" to "present perfect", it would produce (c), etc. (see figure 6)

(a) Le garçon regardait la fille
    (the boy watched the girl)
(b) Le garçon regardait les filles
    (the boy watched the girls)
(c) Le garçon a regardé les filles
    (the boy has watched the girls)

As one can see, this method allows for local as well as for global changes. Actually, the dialogue described corresponds to the communication mode 5. If one changes only one value each time, asking the system to tell right away how this meaning change is reflected in form, one can very quickly build and explore a large search space. Actually, this kind of dialogue between man and machine is much faster and less tedious than it would be between a student and a teacher. The change of meaning and the system's generation of the corresponding form take about a second, even for an untrained user.

Another communication mode that system allows for is called: What would happen if <syntactic modification>. This mode allows the user to perform certain transformations such as passive or pronominalization. In the latter case, the system will ask the user to specify which element he wants pronominalized (the boy, the girl, or both). Assume that, starting from the conceptual structure underlying sentence (a), the user wants to pronominalize respectively the agent, the object, and finally both arguments. In those cases he would get the following output (see figure 7):

Il regarde la fille.  (he watches the girl)
Le garçon la regarde.  (the boy watches her)
Il la regarde.  (he watches her)

By exploring and contrasting the different possibilities, the user is meant to learn how variations of meaning are reflected in form.

4 How to transform the black box into glass box?

In its first version SWIM was only a black box in that the user could learn the rules only on the basis of covariations between an input (meaning) and output (linguistic forms: sentences). He had no way to see what happens between these two extremes.

We shall now turn to the problem of what changes were necessary to convert the system into a glass box. Actually, the solution turned out to be quite simple: using a hybrid form of knowledge representation (in our case graphs à la Sowa, 1984). Encoding all knowledge in a single format allowed us to visualize immediately on the representation the consequences of a choice, irrespective of its nature (pragmatic, conceptual, linguistic). Put differently, every time the system is given some kind of information, it will show on the graph how it is reflected.

Let us take an example to illustrate what has been said here above. The conceptual input being the following:

Please note, that we will vary pragmatic (topicalisation, focus) or linguistic variables (words; choice of syntactic category: noun vs pronoun) but not the conceptual input. The core meaning shall remain constant.

Let us suppose that we decided to start our sentence with the concept PERSON-1. In this case the system would produce the following changes: what used to be called Agent would now be called Subject, the feature voice would be set to active. Had we chosen to topicalize PERSON-2, we would get a passive voice, the AGENT link would change into a prepositional object and the DEEP OBJECT would become the grammatical subject. (figure 9)
Next, suppose we were now to lexicalize the concept HELP. Let's have a look at the consequences. According to the verb chosen (aider, venir en aide, rendre service) we would get critical information concerning the grammatical object. In the case of aider the deep object would map onto a "direct object", whereas in the other two cases the deep object would surface as an "indirect object" (figure 10).

Again these choices do have consequences, in particular, if we decided subsequently to express the concept PERSON-2 as a pronoun rather than as a noun. This case nicely shows that both the form of the pronoun (la vs lui), as well as the nature of the auxiliary may depend on the specific verb chosen (avoir vs être [to be vs. to have]). Both results, as well as those of the syntactic function (direct vs indirect object) are important in French as they codetermine the form of the pronoun as well as the need of agreement between the object and the verb (aidé vs. aidé).
Paul l’a aidée (verb agrees)
Paul lui est venu en aide (no agreement)
Paul lui a rendu service (no agreement)

As we can see, the form of the grammatical object (la vs. lui) depends not only on the coreferential’s intrinsic features (gender & number) but also on the syntactic requirements of the verb (aider requires a direct object, whereas venir en aide requires an indirect object). Put differently, since pronoun synthesis requires not only information concerning person, number and gender, but also information concerning (surface) case (dative vs. accusative, etc.), and since this information is verb dependent, verb choice plays an important role in synthesizing pronouns. It is a relevant parameter, as it codetermines (indirectly) the pronoun’s form.

When we make choices we also make commitments, and it is important to become aware of both of them.

5 Conclusion:

We have presented a system under development whose architecture is cognitively motivated. Its goal is twofold: (a) assist language learning, (b) shed light on the process of sentence generation: what is processed in what order, what are the choices, what are the consequences, etc.?

The qualities of this approach lie in the fact

(a) that the system decomposes the process and visualizes the nature, the implications (outcomes) and the adequacy of each choice. It shows language in action (process vs. product, externalization of mental operations), that is, it helps the user to discover how changes in meaning or discourse purpose are reflected both at the extremes (form side) and at the intermediate levels. In doing so, the system reduces not only the user’s mental load, but it also increases his understanding, hence his control of the whole process.

(b) that the system lends itself to self-instruction (autonomous learning)

(c) that the user is maximally integrated into the process (learner centered approach). In doing this, we hope that the user will learn not only the product (language), but also a method of processing and learning a language.

Finally, the system could serve as a psycholinguistic testbed. As a matter of fact, it allows not only the testing of a specific linguistic theory, but also the strategies people put to use in order to learn a language. In consequence, watching people using this tool, we may gain important insights about natural learning, thus moving eventually from artificial to natural intelligence.

6 References

Sowa J.

Zock, M.
1991 SWIM or SINK : the Problem of Communicating thought, in : Swartz,M. & M. Yazdani (Eds.), The Bridge to International Communication: Intelligent Tutoring Systems for Foreign Language Learning, Springer Verlag, Berlin
The Rosetta Stone

Eurolinguist Language House
Tel.: +31 80-600700

A multimedia CD-ROM program designed by teachers and students for beginning language learning.

The Rosetta Stone has been designed to make language learning enjoyable and natural for beginners. It replicates the give and take of learning that takes place when children acquire their natural tongue.

We all began learning our own language by associating sounds with objects as they are introduced to us in a context where the meaning is clear and reinforcement is immediate. This is the principal learning method behind The Rosetta Stone. By using thousands of carefully selected pictures, it presents a context where the meaning is clear along with the choice of words that confirm the meaning. This follows with immediate feedback to the response, further increasing the students understanding.

This self study program has been developed as a sequenced, step-by-step process. Words build to phrases, phrases to sentences and then dialogues.

Aimed at beginners, the program can be used by all ages. Students can make their way through the exercises at their own pace, concentrating on listening and reading comprehension. The student can also compare their own voice with that of a native speaker, using the listen-record-playback facility (a similarity with the traditional language lab techniques).

The Rosetta Stone is available in English, French, German and Spanish. There is a 'demo disc' available which also offers Russian (an excellent language for first time users to experience the effect of the program).

"Who needs it?
Everybody needs it!"

Thomas Beyer, Ph.D.
chairman, Russian Language Dept.
Middlebury College, Vermont

"Our students begged to use The Rosetta Stone.
They fought to use it."

Kandice Nagy, Teacher
Plains Elementary School
Timberville, Virginia

"The students love it.
I think it's phenomenal."

Amy Frye
Forest Hills Public Schools
Grand Rapids, Michigan

For more information call: Eurolinguist Language House tel.: +31 80 - 600700

A demo is available with all the languages for Fl. 100. Compatible with both Mac. & MS-DOS machines.
A SPEECH-MELODY TRAINER FOR FOREIGN-LANGUAGE INTONATION

Dik J. Hermes and Gerard W.G. Spaai

Institute for Perception Research / IPO
P.O. Box 513, NL 5600 MB Eindhoven, The Netherlands

ABSTRACT

Until now, educational software for training foreign-language intonation has suffered from various technical problems. These problems will be discussed and an outline of a system will be presented in which these problems are overcome.

SPEECH-MELODY TRAINING

In foreign-language teaching, intonation has only rarely received the attention it deserves. Moreover, especially when the teacher is not a native speaker, it is almost impossible to get a correct feeling for the intonation of a foreign language and to acquire the skills necessary for producing correct intonation. In the absence of correct examples and a measure of the correctness of the intonation of utterances, the pupil will not become aware of the errors made. Consequently, the speech of foreign-language speakers is often characterized by improper intonational features derived from the speaker’s native language (Tahta et al., 1981).

The intonation of different languages can differ considerably. For example, in some languages, such as Dutch, many questions are characterized by a rising pitch movement late in the last accented syllable, in many cases not the last syllable of the utterance. This pitch movement is not a rise, but a rise immediately followed by a fall. Another example relates to the British English tags like “Isn’t it?”, “Do you?” and “Was it?”. Whether a question or statement is meant depends on the pitch movements realized on the tags, but this is in many cases misunderstood by the foreigner. Because of these differences, sentences meant as questions by the foreign-language speaker are frequently understood as statements by the native speaker, and the other way round. Also, in many other aspects the intonation of different languages can differ considerably.

In the past, a few attempts have been made to use computers in foreign-languageintonation training (e.g., Aberton et al., 1976; De Bot, 1983; Hiller et al., 1993). In most cases, the unprocessed series of pitch measurements were displayed to the speaker on the screen of a computer. The interpretation of unprocessed pitch contours is often difficult, however, even for people trained in the study of intonation. Unprocessed pitch contours show gaps where the speech sounds are unvoiced and contain microintonation, i.e. short fluctuations in the pitch contour which are not intended by the speaker. Because of their position in the syllable these fluctuations are very difficult to hear, and almost impossible to
imitate even by normal-hearing persons. Therefore, pitch contours should be displayed in a stylized way.

This is related to the study of intonation by means of close-copy stylization (De Pijper, 1983), in which pitch contours are reduced to as small a number of continuous straight lines as possible, without affecting the perceptually relevant properties of the intonation of the utterance. Supplying feedback in the form of close-copy stylizations has proven fruitful in previous research, in which deaf children were taught to produce appropriate intonation by presenting them with stylized copies of the pitch contours of their own utterances (Hermes and 't Hart, 1987; Spaai et al., 1993). Not applying automatic stylization has probably contributed to the lack of widespread use of visual intonation display systems in schools. Other factors are the deficiency of reliable speech-processing algorithms such as pitch-determination algorithms, the lack of a well-designed curriculum for teaching intonation; and the lack of information on the effectiveness of visual displays in teaching intonation.

We now propose to design a software package, meant for teaching the intonation of a foreign language (Spaai and Hermes, 1993). In this system, students will be asked to imitate example contours presented on the upper part of a computer screen. The imitation will then be presented on the lower part of the screen. Both example contour and imitation will be presented as stylized contours. This software package will be called Speech-Melody Trainer, which we plan to develop as a user-friendly, interactive, educational software package, which can be used in teaching the intonation of a foreign language.

REFERENCES


PROMISE : STEPS TOWARDS COMMUNICATIVE ENGLISH TEACHING
IN AN INTERACTIVE CALL SYSTEM

Roul Sebastian John
Universität Osnabrück, Institut für Semantische Informationsverarbeitung, D-49069 Osnabrück, Germany

ABSTRACT
The PROMISE system shows a way how an interactive CALL system for communicative English teaching might be constructed. The system is situation-based and the exercises are created on the basis of concrete communicative circumstances which show up in these situations. As an example, the situation of a car accident has been implemented. The pupil will have to report the accident in a dialogue with a simulated policeman. Later on, he/she has to use spatial relations to describe how the accident happened and to report what he/she has seen. This is again done in an interactive, communicative way. Different problems show up when trying to implement a CALL system for communicative language teaching. These are briefly discussed, as well as some issues concerning the integration of other, "classic" forms of exercises in such a system.

WHAT IS "PROMISE"?
The PROMISE system (PROjekt Mediengestütztes Interaktives Sprachenlernen für das Englische, i.e. project media-supported interactive language learning of the English language) is currently designed and implemented in a one-year student project at the Institute for Semantic Information Processing (University of Osnabrück). With the help and advise of university teachers of CL/Al and language teachers engaged in media-supported language teaching at local schools, the students try to develop an integrated knowledge based CALL system which takes into account not only lexical and grammatical but also communicative aspects of language.

COMMUNICATIVE LANGUAGE TEACHING AND CALL
In discussions with teachers it became apparent that the CALL systems currently available do not meet the demands of modern language teaching (for further information cf. (1)). Therefore such systems are rarely used in practice. Ideally, the PROMISE system should provide for the requirements of communicative language teaching. In this approach, it is emphasized that all elements of the foreign language ought to be introduced in the context of (almost) realistic situations. The pupils should be aware that language is a means of communication and they should use language to express their intentions, not just to complete an exercise with no communicative background.

In PROMISE, this idea is realized by putting the pupil in an adventure-like setting. Concrete situations are shown on the screen and all the pupil's actions and the exercises initiated by him/her or by the system will be connected with this situation.

There will possibly be only one exception to this rule: if the system finds out that the pupil constantly uses a wrong syntactic construction (e.g. a wrong AcP-construction), it may start a "classic" exercise like a gapped text where he or she has to put in the correct form. After this exercise, the system returns to the situation, where the pupil now has to use the newly acquired syntactic construction in a concrete communication.

The situation will change with the pupil's actions. The system will also initiate successive changes to the situation to compel the pupil to communicate in unforeseen circumstances.

EXAMPLE 1:
REPORING A CAR ACCIDENT
Until now only one situation has been integrated into the system: the pupil will see a sequence of pictures (in the future, the pictures should be animated but this has not been worked out yet) which show a traffic accident. The dog of an old lady standing at one side of the street tears at his leash and it snaps. The dog runs onto the street and causes a car to crash into the traffic-light.

The system makes the pupil go into a telephone booth to call the police. Thus a dialogue with a policeman, simulated by the system, is started. There is a special module in the system to simulate such dialogues with the aid of scripts. Here a "report of accident"-script is used. It specifies the questions the system asks and the information the pupil has to convey in his answers. If a particular piece of information can not be found in the pupil's answer, the system requests by asking a specific question which is also specified in the script. All the pupil's inputs are parsed to check for syntactic errors. If the system can not make any sense of the pupil's answers, the "policeman" requests the pupil to reformulate his or her sentence. To make the policeman seem more realistic, there is a wide field
of expressions the system can choose from in order
to put into words that "the policeman did not
understand and wants you to paraphrase what you
said". One of the main aims of the near future will
be to enable the system to identify the same
information given in different paraphrases.

MOTIVATING THE PUPIL TO AVOID
ERRORS

In communicative language teaching (see
above), it is recommended not to interrupt the
pupils to correct their errors while they are talking
because the main aim of this approach is to enable
the pupil to make clear what he wants to
communicate in the foreign language. So there is
no hunt for the perfection of the pupils' expressions.

Nevertheless, in the PROMISE system we want
the pupils to use correct and not just intelligible
English sentences. Chiefly with a view to
discussions it is highly feasible not to correct any
mistake as it crops up since this would interrupt the
pupils' flow of speech and demotivates their will to
contribute (the errors may be collected and
corrected later on). But simulated discussions seem
to be outside the scope of CL (and it could also be
questioned whether they would be of much use). So
it seems not to be too much of a restriction
regarding the concept of communicative language
teaching to force the pupils to use correct English
sentences as far as they know them.

To effect this, the situations should be put in a
wider setting: the pupil will take the role of an
alien sent to Earth on a special mission. He can not
be detected by his physical appearance but only by
the errors he commits in his everyday
communication. He is obliged to find a fugitive
from his planet. To execute his order, he has to
move through the different situations in the system
and he has to take care not to make too many
mistakes.

This part of the system opens a realm of
interesting questions for future research (e.g.
pragmatic issues and the question of how flexibly
such a CALL system could react if certain
conventions are broken and how this can be
detected). But we have not put much effort into this
issue until now.

EXAMPLE 2: COMPLETING A SKETCH

To get back to the traffic accident: after having
reported the accident, the pupil will have to help
another simulated "policeman" to draw a sketch of
how the accident happened. For details cf. (2).

EXAMPLE 3: INTERFERENCE ERRORS

To demonstrate how interference errors might be
handled, another exercise was integrated into the
system. "At the court", the pupil has to describe
what he/she saw when the accident happened.
German pupils will typically use constructions like
"I saw how ..." resembling the German construction
"Ich sah wie ...". This is a typical example of an
interference error. If the system detects an
interference error, it prints out a message stating
that a German construction was used. It will also
display the correct English grammar rule and an
example sentence. The pupil is then asked to type
the correct sentence which is again checked for
further (interference) errors. This exercise is
embedded in a simulated dialogue, too.

INTEGRATING "CLASSIC" EXERCISES

After such a communicative exercise there
should always be the possibility of starting a
"classic" exercise if this is the best way to train
grammatical forms the pupil did not use correctly
(see above). In the context of the "alien-setting"
described above, this may be interpreted as an
exercise established by a computer which is orbiting
round the Earth and takes care that the "alien" does
not make too many mistakes. This "computer" may
report errors that would normally not be noted in
communications with (simulated) human beings.

The pupil should also be given the opportunity
to look up unknown words (although according to
the ideals of communicative English teaching, too
extensive use of a dictionary ought to be prevented
during communication). A situation-oriented
vocabulary learning system already exists: CAVOL
(see (2) in this reader). It was implemented at the
Institute for Semantic Information Processing, so
we decided to integrate it into the PROMISE
system. The system allows the pupil to look up
words by clicking on objects on the screen and he or
she may start the exercises of CAVOL to train
his/her vocabulary. The PROMISE system thus
gives an idea of what an integrated communicative
CALL-system of the future might look like.

MEMBERS OF THE PROMISE PROJECT

advisors: Dr. habil. Helmar Gust, Burkhard
Imeyer, Dr. Petra Ludewig (head of the project),
Mehthild Ricke, Ulrich Sauer
students: Pascal Bauer, Roul Sebastian John,
Friedrich Kronenberg, Anja Krüger, Ulf Krumnack,
André Menzel, Veit Reuer, Robert Unsöld

REFERENCES

(1) Gust, H. / Ludewig, P. / Reuer, V. / Unsöld, R.F.
(1994) : Evaluation auf dem Markt verfügbarer
Software zum computergestützten Spracherwerb; on
behalf of the IBM Germany Information Systems
Gmbh
(2) Kronenberg, F. / Krüger, A. / Ludewig, P.
(1994) : Contextual Vocabulary Learning with
CAVOL (in this reader)
A VOCABULARY WORKBENCH FOR LANGUAGE LEARNERS AND TEACHERS

Maria Klijn-Wuisman and Margriet Jagtman
Delft University of Technology
Applied Linguistics Unit
P.O. Box 5050, 2600 GB Delft
The Netherlands
e-mail: jagtman@wtm.tudelft.nl

ABSTRACT

WOORD is a vocabulary workbench for teachers and learners of Dutch as a foreign language. Teachers can read texts into WOOD and feed new words in these texts into WOOD's computational dictionary. They can also use WOOD to construct computer-aided vocabulary exercises. Students can read the texts contained in WOOD and they can immediately obtain the meanings of "unknown" words. In addition, they can do the vocabulary exercises constructed by the teacher.

1. WOOD

WOORD is a vocabulary workbench for teachers and learners of Dutch as a foreign language. It consists of a lexical database, a text database, and an exercise component. To each headword in the lexical database, the following information can be added:

- a translation into English, German and French;
- a Dutch paraphrase and/or synonym;
- one or more contexts for the word in question;
- a grammatical category;
- a meaning category (based on the classification of McArthur 1981);
- a level marker (beginners, intermediate or advanced).

At present our lexical database contains about 7000 words: (a) all words occurring in the beginners' and the intermediate courses of the Delftse Methode (Montens and Sciarone 1991, Blom et. al. 1993), and in the texts used in our advanced courses; (b) complemented with words from the 4000 most frequent Dutch words not included in (a). However, it should be noted that our workbench is method-independent, so that it can be filled with words from any other method.

WOORD has two groups of users: teachers and students. For each group a separate application has been developed. In the next two sections we describe both applications.

2. THE TEACHER PROGRAM

The teacher program can be used to read ASCII texts into WOOD's text database. First, the teacher has to mark idiomatic expressions in the text. Next, the text can be analysed. "Analysed" in this context means that all words in the text are reduced to headwords in the lexical database through the automatic application of a set of morphological rules. The words which have not yet been incorporated into the dictionary are put in a list and will have to be added to the database. In addition, if a given word in the text can be related to headwords with different grammatical categories, the teacher has to select the category which is appropriate in the context in question. The process of analysis outlined above yields an analysed text in which all words are invisibly linked to their headwords.

The teacher application allows its user to change the contents of the lexical database by adding, changing or deleting words. In addition, the database management system enables teachers to construct queries in order to select specific sets of items from the lexical database. For example, they may select words relating to a specific semantic field, such as politics or minorities, by creating queries with the aid of the meaning categories.

Finally, WOOD's teacher program provides facilities for constructing computer-aided vocabulary exercises with feedback. The program...
automatically computes students’ scores on the exercises and keeps a record of all the answers so that they can be used for further research.

3. THE STUDENT PROGRAM

The student program offers the possibility to read the analysed texts contained in WOORD and quickly search for all kinds of “unknown” words and expressions. The students can use all the information provided by the teacher, thus, they may ask for an English, German or French translation, a context or a synonym. Since students work with texts in which all words are invisibly linked to their headwords, the search process is quick, even when a group of students is using WOORD simultaneously on a network.

After reading a text, the students have access to the vocabulary exercises constructed by their teachers.

Students also have access to the lexical database, but they are not allowed to change its contents. However, they are allowed to use the search routines, so that, for example, they may select words on a specific subject in order to write their own texts.

4. FUTURE WORK

Compared to using a normal dictionary, the use of WOORD has several advantages: it enables teachers to easily provide their students with new texts; it enables them to give specific information on words to be looked up; and most importantly, it enables them to select specific sets of words as a basis for vocabulary exercises. In addition, WOORD enables students to save lots of time through its efficient look-up facility for words, and it provides them with a large variety of exercises allowing them to acquire a particular jargon with ease.

Our plans to enhance the advantages of the workbench include the following:

(1) to create help facilities for both teachers and students;
(2) to extend the number of words in the database;
(3) to develop a routine which uses the level markers of words to enable teachers to determine whether a new text is suitable for their students;
(4) to explore the possibility of adding a parsing facility of texts so that teachers need no longer manually select appropriate grammatical categories of words;
(5) to develop a program for registration of the search behaviour of students when they use the computer dictionary;
(6) to construct a demo version of the program;
(7) to make WOORD suitable for use with other languages than Dutch.

ACKNOWLEDGEMENTS

Special thanks go to Reza Ahmadi, who has done all the programming work on WOORD.

REFERENCES


VERBARIUM AND SUBSTANTARIUM: TWO CALL PROGRAMS FOR LATIN

Joke van de Plassche*, Edwin Bos*, Elena Jongen-Janner
*SOS Nijmegen
Nijmeegsebaan 156
6564 CN Heiliglandstichting
The Netherlands

In this article we describe two Computer Assisted Language Learning (CALL) programs for the rather unusual domain of Latin. Also rather unusual is the idea for making these two programs did not come from software-developers themselves but instead from teachers, members of the CALL-commission of the ‘Vereniging Classici Nederland’ (Classical Association of The Netherlands). On the basis of this commission’s report ‘Lingua Latina ex Machina’ on desired CALL-programs for Latin in high schools, we developed the two computer programs Verbarium and Substantarium.

Verbarium is a drill-and-practice program that aims at automating feature analysis of Latin verb forms. The analysis of verb forms is not trivial since the Latin verb has over 130 forms, and, in conjunction to this, 4 different types of verbs (‘conjugations’), so that the total of different verb forms is over 500. One particular form type can have different affixes for different conjugations. Moreover, there is a lot of ambiguity: one particular affix can signal different verb forms within a verb, and some words are different verb forms of different verbs (e.g., pareris is within the paradigms of both parere and parare, indicative and subjunctive respectively). The proper determination of what verb form a word is is an essential skill in Latin where translating texts is almost the single activity. Substantarium is a similar program for mastering the analysis of noun forms. Though the Latin noun has only 10 different forms, it too has several types (5 ‘declensions’ to be precise) with different word form endings. Here too there is plenty of ambiguity. For example, the suffix ‘-a’ can be found in the paradigm of nouns of the first declension (nominative or ablative singular), and of the second, third and fourth declension (nominative and accusative plural for neuter nouns). In Substantarium the learner must, apart from providing an analysis of the form, also select a correct translation from a predefined list of approximately 600 Dutch words.

Below you see a screen dump from Verbarium. The student should analyze a given verb form by selecting the proper cells in the matrix, one cell in

<table>
<thead>
<tr>
<th>Opgave</th>
<th>VERBARIUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benoem de volgende vorm:</td>
<td>reget</td>
</tr>
<tr>
<td>Opgaven:</td>
<td>5</td>
</tr>
<tr>
<td>1e Keer Goed:</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MODUS</th>
<th>TEMPUS</th>
<th>GENUS</th>
<th>PERSOON</th>
<th>GETAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>indicatius</td>
<td>praesens</td>
<td>activum</td>
<td>eerste</td>
<td>singularis</td>
</tr>
<tr>
<td>coniunctivus</td>
<td>imperfectum</td>
<td>passivum</td>
<td>tweede</td>
<td>pluralis</td>
</tr>
<tr>
<td>imperativus</td>
<td>futurum</td>
<td>dente</td>
<td></td>
<td></td>
</tr>
<tr>
<td>participium</td>
<td>perfectum</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>infinitivus</td>
<td>plq. perf.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Lopen door rijtjes: → ↑ ← ↓ Kiezen: spatie Klaar: enter Help: F1
each column. The noun program Substantarium has a similar interface, yet has just three columns (case, number, and translation). For now we focus on Verbarium.

Verbarium contains all rules for the conjugation of the Latin verb. This has two advantages. First, we did not have to enter 130 words for each of the 247 verbs (32110 words!), but instead just the infinitive and the irregular forms. Second, knowing the conjugation rules itself, Verbarium can provide specific feedback in case the student makes an error. Depending on the situation, Verbarium will provide one of the following types of corrective feedback which should help the student in analyzing the form properly in a second attempt:

- Conjugation feedback: “regit comes from regere, third conjugation”
- Morphological feedback: “regit is structured as reg-e-t”
- Contrastive feedback: “in this conjugation (conj: 3), the subjunctive presens has the suffix -a-”
- Response feedback: “your choice would be regit”
- Other, more general feedback: “look again at the person ending”; “a word with an auxiliary form of esse is always passive”; etc.

On request Verbarium can provide several kinds of information the student may find useful in analyzing the verb form:
1. Conjugation: the type of verb this verb form belongs to.
2. Structure: the structure of the current verb form (e.g. lauda-v-isse-nt).
3. Translation: the main Dutch term for the verb to which the verb form belongs.
4. Paradigm: the list of verb forms of the current verb.
5. Morphology: extensive information about roots and affixes per tense.
6. Tense and mood suffixes: a quick overview of the suffixes (e.g. "e" or "a" in Subjunctive Presens).

The programs have been tailored to the four most commonly used text books in the Netherlands, but they can also be used in conjunction to other text books, opting for a generally accepted terminology. The first thing a student has to do is select his or her text book title and enter the number of the book’s lecture he or she is currently studying. The computer program takes this number into account when selecting a verb or noun and subsequently when selecting a form of its paradigm. This prevents the presentation of words or forms the student has not yet studied in the text book. The most recently studied words and forms are selected a bit more often than the ones studied before. The entire session (all student answers, correct answers, requests for hints, etc.) can be saved on disk in a log file. At the end of a session a detailed overview of the number of incorrect answers per form type is provided, as well as a recommendation as to study which forms a little more. This overview is also saved in the log file for quick inspection of the student’s performance by the teacher.

Verbarium and Substantarium are simple CALL programs that offer one kind of exercise. They are tailored to the teachers’ needs and are easy to use. Generally, teachers of Latin consider it necessary for their students to have paradigms readily available. Our programs are suitable tools to support this goal. Teachers seem to agree, since over 100 schools in both the Netherlands and Belgium currently use our programs. At the time of writing we are finalizing a next program for Latin: Congruarium. This program deals with the agreement between noun and adjective, a notorious difficulty for students of Latin.
Interactive Video in language and culture

Three interactive video programs will be presented at the demonstration: ItalCultura, RumboHispano and IVANA.

ItalCultura

The program is based on the Italian videodisc Deltaria from the Agnelli Foundation. Still images and texts enable the student to study various aspects of Italian culture and civilization.

Once a topic has been selected, a screen with an image and an accompanying Italian text will be displayed. Two main options are available to help the understanding of the texts: translation into Norwegian, a list of words taken from each text.

Furthermore an option to enlarge the image to full screen is available. By using this option a teacher may use the program in an auditorium when giving a lecture. Thereafter the students may study both images and texts in a pc lab.

In addition to exploration of the images and vocabulary the program provides several multiple choice tests pertaining to the factual information.

RumboHispano

This self-study program in Spanish is based on the American videodisc Rumbo al Mundo Hispano from Maryland University. The disc contains 60 minutes video from TV broadcasting, constituting 13 different topics. The program allows for stopping the video at any moment, replaying the current sentence and then having access to a transcription of the sentence and a translation into Norwegian.

In addition to the main video section there is a section based on the transcription of each topic and a section containing a dictionary. These two last sections are fully interactive in the sense that the student may click on any of its sentences and have access to the video, or only to the sound and the translation.

The last part of the program contains two exercises, one which requires the student to reconstruct the original dialogues based on the translation, and the second one consisting of a speaking exercise.

IVANA

This program is the result of a 2-year university project with departmental funding. The program aims to teach both language and culture to students of Norwegian as a second language. Approx. 60 minutes of video have been shot. The video can be explored in the same way as for RumboHispano and in addition there is a database of small video sequences representative of a range of various speech acts, which can be displayed and studied in the same way.

About 300 still images show various aspects of Norwegian culture and civilization, and there are also sound clips giving further backing to the understanding of the video.

The video has been divided into 4 main topics, each of which containing 2-5 video sequences. For each of the video sequences there are 3 different exercises focusing on vocabulary and textual understanding. The program is meant to be used as a point of departure for further writing and following up in class.
In all the programs a notepad is available so that the student may take down vocabulary or start writing essays connected to the programs.

**Technical specifications:**

Hardware: Macintosh II or higher.

14' colour monitor.

Video board: RasterOps 24STV or DVA-4000.

Videodisc player: Pioneer or Sony.

Programming Language: SuperCard.

Developer: Senior Lecturer Signe Marie Sanne
A COMPUTER BBS IN LANGUAGE INSTRUCTION

Timothy C. Stewart
English Programs for Internationals
University of South Carolina
Byrnes 311
Columbia, SC 29205 USA
Timothy_Stewart@epi.sc.edu

ABSTRACT

Multiple options exist for communication using computers. E-mail packages, the internet, file transfer protocols, and gophers have become an increasingly important part of instruction. The world of computer communication is a confusing array of programs and platforms, yet the value of using this type of technology for language learning cannot be overestimated. Our program has chosen a Macintosh-based telecommunication and information services system, FirstClass, which also supports Windows and DOS. The presenter will show how the use of a computer bulletin board facilitates communication among students, teachers, staff, and administration.

DEMONSTRATION

The presenter will demonstrate how a consistent, user-friendly interface promotes interaction and encourages even basic learners to explore the system over a computer network or by modem. It allows students to send messages, transfer files, 'chat' with other students or teachers in real time, and access the internet. In addition, the internet gateway enables the program to receive newsgroups, providing a wide range of content-based reading material.

We will show samples of the various ways our program utilizes the program to integrate the BBS into existing instructional and program goals. As time and equipment allow, the participants in the workshop will have an opportunity to work with the program as students, teachers, and administrators.

COMPUTER-BULLETIN BOARDS

Computer bulletin boards allow for an optimal mixture of messaging, providing access to both individuals and groups. Unlike basic e-mail, where messages have to be addressed specifically, a bulletin board allows the option of posting information which can then be retrieved by anyone who is interested in it. Of course, e-mail can be sent as well. In addition, files containing text, graphics, and sound can be included with messages or postings.
PARLEVINK Research Topics

The Parlevink Project started in January 1992. It did not start from scratch. In previous years research took place in the area of theory of formal and programming languages (theoretical computer science, compiler construction) and more and more this research became influenced by potential applications in the area of natural language processing. Currently the following three research directions are distinguished:

- research which concentrates on syntactic formalisms and where syntax is the starting point for studying the description and processing of semantic and pragmatic aspects of language;
- research which concentrates on the representation of meaning in dialogue modelling and where syntax is of secondary importance;
- research which concentrates on modelling language behaviour with the help of neural networks and where language learning and integrated use of syntactic, semantic and pragmatic knowledge are the main characteristics.

In 1993 a start has been made with the integration of the different research tracks in the design of a natural language interface that allows a user to ask information about theatre performances in a city. This has taken the form of joint research with PTT-Research.

PARLEVINK Researchers

More than ten researchers, including Ph.D. students, are involved in the project. In 1994 four Ph.D. students are involved in the research. Programming support is provided by the Computing Laboratory of the Department of Computer Science. A large number of computer science students are performing their M. Sci. work in the project. It is not unusual that they spend part of their education in companies in the Netherlands or with research groups in the USA.

PARLEVINK Activities

PARLEVINK participates in the Centre of Telematics and Information Technology (CTIT) of the University of Twente. Research is published in books, journals and proceedings of (international) workshops and conferences (COLING, ICANN, KONVENS, IWPT, etc.). A complete list of publications is available on request. Twice a year a workshop (TWLT: Twente Workshop on Language Technology) is organised. Proceedings of these workshops are available. In 1991 there were workshops on Generalised LR Parsing and on Linguistic Engineering. In 1992: Connectionist Natural Language Processing and Pragmatics in Language Technology. In 1993: Natural Language Interfaces and Parsing Natural Language. Students and project members are informed about research, lectures and other activities during weekly meetings and in the PARLEBODE, a monthly newsletter.
Twente Workshops on Language Technology

The TWLT workshops are organised by the PARLEVINK project of the University of Twente. The first workshop was held in Enschede, the Netherlands on March 22, 1991. The workshop was attended by about 40 participants. The contents of the proceedings are given below.

Proceedings Twente Workshop on Language Technology 1 (TWLT 1)
Tomita's Algorithm: Extensions and Applications
Eds. R. Heemels, A. Nijholt & K. Sikkel, 103 pages.

Preface and Contents
A. Nijholt (University of Twente, Enschede). (Generalised) LR Parsing: From Knuth to Tomita.
G.J. van der Steen (Vleermuis Software Research, Utrecht). Unrestricted On-Line Parsing and Transduction with Graph Structured Stacks.
R. Heemels (Océ Nederland, Venlo). Tomita's Algorithm in Practical Applications.
M. Lankhorst (University of Twente, Enschede). An Empirical Comparison of Generalised LR Tables.
K. Sikkel (University of Twente, Enschede). Bottom-Up Parallelization of Tomita's Algorithm.

The second workshop in the series (TWLT 2) has been held on November 20, 1991. The workshop was attended by more than 70 researchers from industry and university. The contents of the proceedings are given below.

Proceedings Twente Workshop on Language Technology 2 (TWLT 2)
Linguistic Engineering: Tools and Products.

Preface and Contents
A. Nijholt (University of Twente, Enschede). Linguistic Engineering: A Survey.
B. van Bakel (University of Nijmegen, Nijmegen). Semantic Analysis of Chemical Texts.
T. Vosse (NICTI, Nijmegen). Detecting and Correcting Morpho-syntactic Errors in Real Texts.
A. van Rijn (CIAD/Delft University of Technology, Delft). A Natural Language Interface for a Flexible Assembly Cell.
J. Honig (Delft University of Technology, Delft). Using Delta in Natural Language Front-ends.
D. van den Akker (IBM Research, Amsterdam). Language Technology at IBM Nederland.

The third workshop in the series (TWLT 3) was held on May 12 and 13, 1992. Contrary to the previous workshops it had an international character with eighty participants from the U.S.A., India, Great Britain, Ireland, Italy, Germany, France, Belgium and the Netherlands. The proceedings were available at the workshop. The contents of the proceedings are given below.

Proceedings Twente Workshop on Language Technology 3 (TWLT 3)
Connectionism and Natural Language Processing
Eds. M.F.J. Drossaers & A. Nijholt, 142 pages.

Preface and Contents
L.P.J. Veenenturf (University of Twente, Enschede). Representation of Spoken Words in a Self-Organising Neural Net.
P. Wittenburg & U. H. Frauenfelder (Max-Planck Institute, Nijmegen). Modelling the Human Mental Lexicon with Self-Organising Feature Maps.
W. Daelemans & A. van den Bosch (Tilburg University, Tilburg). Generalisation Performance of Back Propagation Learning on a Syllabification Task.
E.-J. van der Linden & W. Kraaij (Tilburg University, Tilburg). Representation of Idioms in Connectionist Models.
J.C. Scholtes (University of Amsterdam, Amsterdam). Neural Data Oriented Parsing.
M.F.J. Drossaers (University of Twente, Enschede). Hopfield Models as Neural-Network Acceptors.
R. Reilly (University College, Dublin). An Exploration of Clause Boundary Effects in SRN Representations.
S.M. Lucas (University of Essex, Colchester). Syntactic Neural Networks for Natural Language Processing.
R. Miikkulainen (University of Texas, Austin). DISCERN: A Distributed Neural Network Model of Script Processing and Memory.

The fourth workshop in the series has been held on September 23, 1992. The theme of this workshop was "Pragmatics in Language Technology". Its aim was to bring together the several approaches to this subject: philosophical, linguistic and logic. The workshop was visited by more than 50 researchers in these fields, together with several computer scientists. The contents of the proceedings are given below.

Proceedings Twente Workshop on Language Technology 4 (TWLT 4)
Pragmatics in Language Technology

Preface and Contents
Preface and Contents
A. Nijholt (University of Twente). Natural Language Parsing: An Introduction.
V. Manca (University of Pisa). Typology and Logical Structure of Natural Languages.
R. Bod (University of Amsterdam). Data Oriented Parsing as a General Framework for Stochastic Language Processing.
M. Stefanova & W. ter Stal (University of Sofia / University of Twente). A Comparison of ALE and PATR: Practical Experiences.
J.P.M. de Vreught (University of Delft). A Practical Comparison between Parallel Tabular Recognizers.
M. Verlinden (University of Twente). Head-Corner Parsing of Unification Grammars: A Case Study.
Th. Stürmer (University of Saarbrücken). Semantic-Oriented Chart Parsing with Defaults.
G. Satta (University of Venice). The Parsing Problem for Tree-Adjoining Grammars.
F. Barthélémy (University of Lisbon). A Single Formalism for a Wide Range of Parses for DCGs.
C. Cremers (University of Leiden). Coordination as a Parsing Problem.
M. Wirén (University of Saarbrücken). Bounded Incremental Parsing.
V. Kubon and M. Platek (Charles University, Prague). Robust Parsing and Grammar Checking of Free Word Order Languages.
V. Srinivasan (University of Mainz). Punctuation and Parsing of Real-World Texts.
T.G. Vosse (University of Leiden). Robust GLR Parsing for Grammar-Based Spelling Correction.

The seventh workshop in the series took place on 15 and 16 June 1994. It was devoted to the topic "Computer-Assisted Language Learning" (CALL). The aim was to present both the state of the art in CALL and the new perspectives in the research and development of software that is meant to be used in a language curriculum. By the mix of themes addressed in the papers and demonstrations, we hoped to bring about the exchange of ideas between people of various backgrounds.

Preface and Contents
L. Appelo, F.M.G. de Jong (IPO / University of Twente). Computer-Assisted Language Learning: Prolegomena
M. van Bodegom (Eurolinguist Language House, Nijmegen, The Netherlands). Eurolinguist test: An adaptive testing system.
B. Cartigny (Escape, Tilburg, The Netherlands), Discusen CD-ROM XA.
H. Alsay Guvenir, K. Oflazer (Bilkent University, Ankara). Using a Corpus for Teaching Turkish Morphology.
G. Kempen, A. Dijkstra (University of Leiden, The Netherlands). Towards an integrated system for spelling, grammar and writing instruction.
F. Kronenberg, A. Krueger, P. Ludewig (University of Osnabrueck, Germany). Contextual vocabulary learning with CAVAL.
S. Lobke (Rotterdam Polytechnic Informatica Centrum, The Netherlands). Teachers, Students and IT: how to get teachers to integrate IT into the language curriculum.
The fifth workshop in the series took place on 3 and 4 June 1993. It was devoted to the topic "Natural Language Interfaces". The aim was to provide an international platform for commerce, technology and science to present the advances and current state of the art in this area of research.

Proceedings Twente Workshop on Language Technology 5 (TWLT 5)
Natural Language Interfaces
Eds. F.M.G. de Jong & A. Nijholt, 124 pages.

Preface and Contents
F.M.G. de Jong & A. Nijholt (University of Twente). Natural Language Interfaces: Introduction.
L. Boves (University of Nijmegen). Spoken Language Interfaces.
J. Nerboune (University of Groningen). NL Interfaces and the Turing Test.
D. Speelman (University of Leuven). A Natural Language Interface that Uses Generalised Quantifiers.
W. Menzel (University of Hamburg). Title.
G. Neumann (University of Saarbrücken). Design Principles of the DISCO system.

The sixth workshop in the series took place on 16 and 17 December 1993. It was devoted to the topic "Natural Language Parsing". The aim was to provide an international platform for technology and science to present the advances and current state of the art in this area of research, in particular research that aims at analysing real-world text and real-world speech and keyboard input.


C. Schwind (Universite de Marseille, France). Error analysis and explanation in knowledge based language tutoring.

J. Thompson (CTI, Hull, United Kingdom/EUROCALL). TELL into the mainstream curriculum.

M. Zock (Limsi, Paris, France). Language in action, or learning a language by watching how it works.

Description of systems demonstrated:

APPEAL (Institute for Perception Research, Eindhoven)

Bonacord, Meli-Melo, etc. (School of European Languages & Cultures, University of Hull)

Computer BBS in language instruction (English Programs for Internationals, University of South Carolina)

Discatext (Escape, Tilburg)

Error analysis and explanation (CNRS, Laboratoire d'Informatique de Marseille)

ItalCultura, RumboHispano and IVANA (Norwegian Computing Centre for the Humanities, Harald)

It's English (Department of Educational Sciences, Utrecht University)

Multimedia course for learning Dutch (SLO, Enschede)

Part of CATT (Department of Computer Engineering and Information Science, Bilkent University, Ankara)

PROMISE (Institut für Semantische Informationsverarbeitung, Universität Osnabrück)

Speech-Melody trainer (Institute of Perception Research, Eindhoven)

The Rosetta Stone (Eurolinguist Language House Nijmegen)

Verbarium and Substantarium (SOS Nijmegen)

WOORD (Applied Linguistics Unit, Delft University of Technology)

FLUENT-II (George Mason University, Washington)