LINGUISTIC ENGINEERING: A SURVEY

Anton Nijholt
Computer Science Department, University of Twente,
P.O. Box 217, 7500 AE Enschede, The Netherlands
anijholt@cs.utwente.nl

ABSTRACT
This paper is a short introduction to the field of linguistic engineering (LE), its products, tools and methods. There is no intention to cover present (computational) linguistic theories on syntax, semantics or pragmatics. Neither do we want to consider the associated formalisms in computational linguistics and artificial intelligence. They will be mentioned in passing as far as they play an essential role in the main stream of practical natural language processing systems. Rather we mention software that has been developed to aid authors of documents to produce correct texts, translators to produce correct translations and children, students or non-native speakers to learn about a natural language. Moreover, we want to discuss software that allows users to access information systems using natural language and software for recognition, understanding and production of speech. Whenever possible, examples are taken from projects and research groups in the Netherlands. However, completeness is not pursued. Rather than that, an attempt is made to illustrate the field of linguistic engineering, its methods and some of the goals that can be strived at.

CONTENTS
1. Introduction
2. Theory, Methods and Tools for LE
3. Computer-Aided Writing Programs
4. Machine Translation
5. Linguistic Teachware
6. Document Production Systems
7. Information Retrieval
8. Natural Language Interfaces
9. Speech Processing Systems
10. Knowledge Extraction and Acquisition
11. NL Specification and -Design
12. Multimedia interfaces
13. Video- and Audiotex
14. Computer-Supported Cooperative Work
15. Literature

1. INTRODUCTION

Underlying the natural language processing systems of the nineteen-nineties is research conducted by a wide variety of scientists: (computational) linguists, software engineers, theoretical computer scientists, logicians, cognitive scientists and artificial intelligence researchers. Linguistic software has not been invented in the nineteen-nineties. Especially machine translation and information retrieval have a long history. In the nineteen-fifties and early nineteen-sixties the emphasis was on machine translation. The general idea was that no deep linguistic theory was needed to build machine translation systems. Observations by Bar-Hillel and the publishing of the ALPAC report led to confusion, reduction of the research efforts and a change of research interests to artificial intelligence and the more fundamental area of computational linguistics (see Nijholt[1988] for details). In the late sixties and early seventies many pilot systems were built in which simple language processing tasks were demonstrated. In many cases the linguistic knowledge that was incorporated was shallow. Work on some large machine translation projects continued and research in speech processing was stimulated by several government financed projects in different countries. From the advent of the relational databases in the nineteen-seventies came an impetus to develop natural language interfaces to query a database system. Another impetus came from the text-processing facilities on the personal computers that became available in the nineteen-eighties. Hyphenation and spelling correction turned out to be less simple than at first sight would be the case. Moreover, more than ever texts were stored on hard disks, floppy disks and CD-ROMs and although the information explosion
was already predicted in 1945, it took its full flight in the nineteen-eighties. Whatever text could be stored on these storage media was stored. One may expect that the same will happen for pictures, drawings, paintings, sound (speech, music) and any combination of these with text in the nineteen-nineties. Many approaches to natural language processing that could not be followed in the 1960s, the 1970s and the 1980s are now possible because of available space for storage and speed of computing.

More than ever the technology part of natural language processing is becoming important and the engineering aspects obtain emphasis. That the awareness of these aspects arose may partly be due to pressure of funds providing bodies who want to see results after many decades of promises. On the other hand, it has become clear that non-trivial natural language processing systems are complex software systems and the design and implementation of these systems require every technique that has been developed in the areas of project management and software engineering. Apart from the complexity of these systems it is also the lack of an integrated theory of language processing that includes syntactic, semantic, pragmatic and world knowledge that makes it impossible to design natural language processing systems in a straightforward way. Due to these circumstances we see the emerge of new names to identify the field of research and development in natural language processing systems. Hence, we are talking about language technology, language engineering and linguistic engineering. In 1991 the Commission of the European Communities announced the LRE programme, a subprogram Linguistic Research and Engineering of the program Telematic Systems in Areas of General Interest. In the technical background document it is mentioned that "the move towards the future information society involves parallel progress in three domains, namely i) Information Technologies (IT), ii) Telecommunications, and iii) Natural Language Processing (NLP)." It is argued that only now, with the ability to transfer and store enormous amounts of data, it is possible to enter the NLP field with an emphasis on content rather than on form. The approach that is advocated is technology-based and application-driven. This approach is the 'Linguistic Engineering' approach. The frame below borders the definition and the aims of this field. 'Natural language as it is in everyday use in different domains', means that, as Linguistic Engineering (LE) is an engineering endeavour, which is to combine scientific and technological knowledge in a number of relevant domains (descriptive and computational linguistics, lexicology and terminology, formal language, computer science, software engineering techniques, etc.). LE can be seen as a rather pragmatic approach to computerized language processing, given the current limitations of theoretical CL.

LE is not based on abstract selective models, but aims to provide tools suitable for handling natural language as it is in everyday use in different domains. The linguistic engineer needs to develop linguistic descriptions of real phenomena, treatable by today's computing technology, storable in computer memories and transmissible over communications networks.

LE aims to produce useful applications, matching a product to its functional environment. As with any sort of engineering, the ultimate aim is not to explain, but to make the products we need, balancing technical feasibility, availability of linguistic resources and development cost with user expectations and requirements.


has been argued by Tomita[1991], we cannot confine ourselves to "linguistic" sentences, but we have to look at "real" sentences, and instead of solving "interesting" problems we have to solve "uninteresting" problems with no general linguistic principles behind them.

Presently, the language industry is a growing segment of information technology. This can be seen from the introduction of new language technology products on the market, but also from publications as, for example, LI-Monitor, a bimonthly newsletter that aims at an audience that is interested in up-to-date information about companies, products and people working in the field of natural language processing. Another offspring that should be mentioned are the fairs where the industry shows its language products. Recent fairs that have been held are the London Language Show, the Salon International des Industries de la Langue (Paris, 1991) and the Day of the Document in the Netherlands (Ede, 1992). It is also possible to mention government-financed projects, e.g. in the Netherlands, where we have SPIN (Stimuleringsprojectteam Informatica-
onderzoek Nederland), a government foundation for stimulating research in computer science, that has supported a Men-Machine Communication project accomplished by ITK, NICI, OcE and IPO with support of BSO and Digital Equipment. The same foundation stimulated research in speech technology accomplished by the universities of Amsterdam, Leiden and Utrecht and by PTT and IPO (Stichting Spraaktechnologie). BSO, a Dutch software house, received more than seven million gilders for a machine translation project.

A variety of natural language systems and technologies are being offered in the market. In this paper we present a survey of natural language systems as they become available and visible for users of workstations and pc's. The paper touches on the research topics from which improvements of the existing systems can be expected. In this paper examples are often taken from the language products that are offered in the Netherlands and the R&D efforts that take place at Dutch universities and research institutes. With a single exception we will not discuss connectionist approaches to natural language processing tasks. At this moment practical applications of these approaches are not available and cannot be expected in the near future. However, interesting research is done and lots of areas are covered by this research. For example, there are connectionist approaches to hyphenation, phonemisation (text-to-speech), speech recognition, syntactic analysis, case frame analysis, script-based analysis, word category prediction, word sense disambiguation, anaphor resolution, machine translation, learning of grammar and learning of word co-occurrences. A rather comprehensive survey of this research can be found in Drossaers and Nijholt[1992].

2. THEORY, METHODS AND TOOLS FOR LE

2.1. Problems

If a natural language processing system is based on grammar, then, with a few exceptions, the grammar is constraint-based. That is, there is a context-free skeleton with rules made up of complex symbols consisting of the name of a category and feature specifications. These features express constraints that can be enforced with tests associated with the rules. In general the constraints are syntactic (number agreement, subcategorization, count noun), but in principle other constraints are possible. Syntactic analysis is done with respect to such an augmented context-free grammar. This analysis produces the structure of the sentence. What are the constituents and how do they relate? The structure is made visible in a parse tree with the features associated with the nodes. Semantic analysis aims at understanding the sentence: What does it mean? Meaning is produced by mapping sentences into a formal, meaning representation, language. This is done using linguistic (among others the syntactic structures) and logico-linguistic knowledge. In a logical language constraints on predicates can be introduced to prevent incorrect semantic readings of syntactically correct sentences. The main problem in natural language processing is the resolution of ambiguity. Syntactic and semantic constraints can help in anaphor resolution and handling fragmentary input, that is, what is the antecedent of an anaphor and which words have been left out? With the help of pragmatic information, domain world knowledge or general (common sense) world knowledge remaining ambiguities have to be solved in order to obtain a most likely interpretation and meaning representation of the utterance. There is much research on meaning representation languages. Some existing languages have been derived from first-order predicate calculus or intensional logic. It is clear that a rich and powerful language is needed in order to be able to represent meaning from natural language utterances. Well known is the World Model Language (WML) of Philips' PHILQAI question-answering system (Scha[1983]). Obviously, it depends on the application how complete the meaning has to be represented. If the application involves access to a database then database organisation knowledge has to be used to map the logic or meaning representation onto a search representation. This search representation can be formulated in a database query language (e.g., SQL, Standard Query Language) for retrieving information from a (relational) database.

At this moment, due to the vastness, richness and complexity of natural language, we are not able to define rules and constraints and to construct and use knowledge bases that allow to deal with all details of natural language. This is already the case at the syntactical level. Here the problem is expressed by saying that constructed grammars leak and, moreover, don't scale up. This does not mean that incomplete linguistic theory is not useful. Not for all natural language applications it is necessary to derive the full meaning of the
sentences that are processed. Moreover, even in
tasks that seem simple at first hand such as
hyphenation or grammar checking, it is sometimes
necessary to make an appeal to extra-linguistic
knowledge. Consider the following two sentences
(Sag[1991]):

List the only Frenchman among the programmers
who understands English.

List the only Frenchman among the programmers
who understand English.

We cannot expect a grammar checker which 'only'
uses linguistic knowledge to detect a possible
subject-verb agreement error in these sentences. In
fact, there is no such error. Similarly, someone
with basic knowledge about geometry will assume
a spelling error (rather than a conceptual error) in
the next sentence concerning a right triangle
(Véronis[1988]):

Draw the sides opposite the right angle.

One might say, that only in very restricted subject
areas and by allowing very restricted language use
one may expect to obtain a satisfactory model
based on the above mentioned rules, constraints
and logical languages. Better results can be
obtained by adding more knowledge sources, e.g.,
knowledge about sequences of events in the
domain of discourse or about the structure of
objects in this domain. The AI-world has
introduced scripts, frames, maps (memory
organisation packets) and other sophisticated
schemes to represent these types of knowledge.

2.2. Grammars

Computational linguists devise grammar
formalisms that have nice computational
properties or that express a linguistic or psycho-
linguistic theory. In order to support the
development of a grammar for a natural language
according to a particular theory (or a family of
related theories) work has been done on so-called
grammar workbenches or development
environments. They provide a grammar
specification formalism and they can contain
grammar and lexicon editors, consistency
checkers, facilities for tracing and debugging,
morphological analyzers and facilities for
incremental grammar compilation and sentence
generation. It may be advantageous to have a
grammar workbench that provides an explanation
facility, as has become usual in expert systems.

Such a facility makes it easier to debug the
grammar and to introduce changes in the
grammar. An example of a grammar workbench is
presented by M.-J. Nederhof and others in a paper
in these proceedings. Sometimes natural language
processing tools or systems are offered by research
groups for free or for a low price to fellow-
researchers. An example is D-PATR, a
development environment for unification-based
grammars (Karttunen[1986]). It contains a parser,
a grammar editor and debugging tools. ALS
(Amsterdam Linguistic Systems) offers ProfGlot,
a multilingual natural language processor, based
on functional grammar, with separate modules for
different languages and for parsing and
translation. It is not intended for building
industrial applications. Rather it is a software
toolkit for computational linguists and computer
scientists interested in the implementation of a
particular linguistic theory. Such development
environments, are not necessarily adequate for
applied projects that aim at large scale grammar
development or at building a practical natural
language processing system. Nevertheless,
sometimes comprehensive workbenches result
from these research workbenches (see also the
section on natural language interfaces). Well
known examples of wide-coverage realistic
grammars in applied projects are the Linguistic
String Grammar (Sager[1981]), the DIAGRAM
grammar (Robinson[1982]), IBM's PEG
grammar1) (Jensen et al[1986]) and the grammars
developed in the machine translation project
METAL (White[1987]). Augmented context-free
grammars and procedural programming facilities
to deal with the inadequacies of the pure context-
free grammar formalism provide the flexibility
necessary to realize a grammar. Other projects for
grammar development that started later and that
attempt to adopt a middle course in flexibility and
linguistic adequacy are the Hewlett-Packard NLP
project (Pollard and Sag's head-driven phrase
structure grammar) and the Alvey Natural
Language Tools project (close to Generalized
Phrase Structure Grammar, see Boguraev[1988]).
Another system that has to be mentioned is the
Unisys PUNDIT (Prolog UNDERstanding of
Integrated Texts) natural language processing
system (Grishman and Hirschman[1986],
Hirschman et al[1989] and Lang and Hirschman-
[1988]) that has been applied to the analysis of
Navy messages and has been ported to different

1) PEG stands for PLNLP English Grammar. PLNLP stands for Programming Language for Natural Language Processing.
domains. Moreover, it has been integrated with knowledge representation, reasoning and speech recognition systems. PUNDIT is a modular natural language processing system consisting of syntactic, semantic and pragmatic components.

2.3. Lexicons
A natural language processing system requires a computational lexicon. It is common practice that each research group that designs and builds a natural language processing system also constructs its associated lexicon. Such a lexicon is hand-coded and each entry is tailored to the particular application. Dependent on the application there will be semantic structure (e.g., inheritance relations) in the lexicon. Creating a lexicon is an expensive task and a scale-up is always hard to realize. A lexicon for an application domain can be based on a compromise between storing and computing information. For example, information about inflected forms of words, syllables, word stress or phonological counter parts of entries in a lexicon can be stored with each entry or can be computed with rules (which, however, do not always produce the correct result). Tools based on these rules, for example algorithms that construct pronunciation representations of words, are useful for lexicographic firms. Most countries have institutes that maintain text databases for lexicological research. For example, at INL (Institute of Dutch Lexicology) a representative overview of contemporary standard Dutch is available. The Brown Corpus is a database containing more than one million words of English text hand-tagged with part of speech information. Tagging of corpora (even without syntactic analysis) is a labor-intensive task that should be done automatically. From tagged corpora it is possible to gather statistical information that can be used in disambiguation processes such as attachment of relative clauses and prepositional phrases, determining of scope and pronoun reference resolution. These databases can be used to support dictionary projects and they provide material to test automatic morphological and syntactic analysis. Interactive access with a query system allows a lexicographer or a grammarian to examine the samples.

A different approach for creating a computational lexicon is to start with a machine readable dictionary. In the early 1980s machine readable versions of well known dictionaries became available. However, dictionaries are designed for human use. Therefore a machine readable dictionary has to be converted into a format that suits natural language processing tasks. There are some research projects which aim at automatic or semi-automatic extraction of syntactic and semantic information from machine readable dictionaries in order to obtain a computational lexicon. In doing this, there is need for grammatical description and parsing of the entries of the dictionary. It is not unusual to use a general natural language parser such as the PEG parser or Sager's Linguistic String Parser for this task. The information that is extracted can take the form of synonymy, antonymy and taxonomy relations, but also other relations (e.g., 'part-of' or 'has-member') can be extracted in order to build a semantic network on which a lexical database or knowledge base can be based. This database can be queried and it can make inferences to answer questions. An example of a broad-coverage lexicon that is obtained from a machine readable lexicon is the 'meta-lexicon' from IBM Yorktown Heights. The impetus for building it arose from the needs of a text-critiquing system (CRITIQUE). Entries of the lexicon show the words, their features and their feature-values. The features that are distinguished are syntactic, morphological, phonological, semantic, stylistic and graphemic. From the meta-lexicon information can be extracted and enhanced with theory-specific and application-specific information (Klavans[1988]). Another project is reported in Guthrie et al[1990]. Here techniques are designed for automatically constructing a taxonomy of word senses from a machine readable dictionary. Both projects use the Longman Dictionary of Contemporary English. A large-scale project where a machine tractable dictionary is obtained by hand-coding is the CYC Project. Its original aim was to code one million entries of a dictionary and embed the semantic information in an environment which allows the use of heuristics, rules for generalisation and inferences to understand sentences (Lenat and Feigenbaum[1987]).

2.4. Expectation-Driven Analysis
We already mentioned that not for all natural language applications it is necessary to have a full linguistic analysis of the sentences of a natural language text. A research area with many practical applications is that of scanning text while having expectations for particular information. The expectations are possible because they define what is relevant for the intended user or because the domain is such that at least part of the expectations will be satisfied by the contents of
the text that is processed. Examples of applications are systems for the skimming of texts for particular pieces of information while all other information is ignored, such as in categorizing newspaper stories, and systems that read texts in constrained domains and that extract conceptual information from the text in order to make an abstract. While in the former case an extraction method can be top-down or expectation-driven, in the latter case a top-down and bottom-up (or language-driven) strategy have to be integrated. A top-down strategy is tolerant of ill-formedness and unknown words. However, it can easily be fooled when the information is provided in an unusual way. In a bottom-up strategy the parser is the most important knowledge source.

Much work on expectation-driven text understanding followed from early script and frame based systems that used Schank's conceptual dependency representation scheme. These systems lack the construction of explicit syntactic structures. The most famous is SAM (Script Applyer Mechanism), a story comprehension program. In later systems other high-level structures that control understanding were used: plans, goals, 'sketchy' scripts, e.g., in FRUMP (DeJong[1982]), and various types of thematic and memory organization packets, e.g., in BORIS (Dyer[1983]). See also section 10.

3. COMPUTER-AIDED WRITING PROGRAMS

In the nineteen seventies some firms have offered tools for the author or typist of letters, documents or papers. Well known is a text processing machine (Wang 1200), offered by the Wang company in 1971, with which texts could be stored on tape and recalled for text processing. Some years later a more sophisticated system was introduced by Wang, the Word Processing System (WPS), in which the tape was replaced by a disc. These systems became obsolete with the advent of the micro-computers and their software for text editing. Programs for hyphenation and for detection and correction of spelling errors, however, were already available in the early 1960s (see e.g. Brandt Corstius[1965] and Damereau[1964]). Presently, all text-editing systems, but also presentation programs\(^2\) for pc's offer automatic aids for processing correct texts. Among them are spelling correctors, hyphenation programs and thesauri. More advanced pc-based systems include spelling control for different languages and grammar checkers (Word for Windows 2.0). Spelling programs offer the user the possibility to add new words to the dictionary. In this way it is even possible to make a dictionary for a different language (a Frisian spelling corrector, or rather staveringskorreksje program, was made this way!). Correct capitalization, consistent use of spelling, punctuation and style are other aspects of text processing that can be looked at. Modest but useful capabilities may be the flagging of double words or checking the use of English 'a/an'. While grammar checking deals with ill-formed input (for example, wrong agreement) that must be corrected, style checking deals with input that is grammatical well-formed but that for reasons of style should be corrected. Corrections may improve the readability by, among others, reducing the sentence length or the complexity of sentence construction. Style correction may also amount to conforming to certain guide-lines, for example, guide-lines for writing technical documents. Many people know the Unix style checker, available on Unix operating systems. Like many other style checkers, its judgements about style are judgements about readability based on formula which have sentence length, word length and word repetition as main variables. Hence, there are stylistic norm values and deviations of these values are considered to be incorrect style. Several commercial software systems have been influenced by the Unix Writer's Workbench (Cherry et al.[1983]). An early example of a style checker which uses parse trees is embedded in IBM's EPISTLE, (Evaluation, Preparation and Interpretation System for Text and Language Entities), a system for analyzing texts and with the capability to draw conclusions about the author (sex, intelligence, background, etc.). See Heidorn et al.[1982]. Examples of more recent grammar and style checkers are Grammatik, Qritique and RightWriter (for Windows)\(^3\). These checkers may comment on the choice of words, punctuation, wordiness, verbless phrases, changes in tone and style and may draw attention to possible ambiguities. In RightWriter parse trees can be displayed graphically. CRITIQUE is IBM's successor of EPISTLE. It uses the already

\(^2\) E.g., the editing systems MS-Word, Wordstar, WordPerfect, AmiPro and Word for Windows, and the presentation programs Persuasion and PowerPoint.

\(^3\) And many others: CorrectText, CorrectSpell, MacProof, Correct Grammar, Editor, Readability, PowerEdit, ...
mentioned PEG grammar and the parser attempts to produce parses even if syntactic constraints are violated. Due to its broad coverage grammar and lexicon early versions of the system turned out to be slow. In 1988 it was reported (Richardson and Braden-Harder[1988]) that in order to achieve satisfactory performance an implementation was investigated where parts of the system that could run in parallel were distributed over multiple processors. So-called 'parsing server' programs made it possible to distribute sentences over different processors. Although the implementation used a network of mainframes, it was expected that in the future a workstation based network or a parallel processor machine could be used. More recent information about CRITIQUE can be found in the paper by D. van den Akker in these proceedings. Winkelmann[1990] discusses progress in an ESPRIT II project on style checking. Among the participants are Siemens, Germany and Triumph-Adler, Germany. In this project the style checker is part of an integrated multilingual toolkit (a Translators Workbench) and it uses parser and lexicons of this kit. An interesting aim is to translate style from one language to another. In research projects there are attempts to develop stylistic grammars to provide the basis for stylistic parsers. These grammars and parsers have to be used in machine translation systems.

In order to get a better idea of the present possibilities of a grammar and style checker for a personal computer we take a closer look at Grammatik. The program can be used for interactive checking of documents for grammatical and spelling errors, and offers advice on how to correct these. Rules for detecting errors are grouped in more than forty rule classes that can be enabled or disabled by the user. New rules can be created and existing rules can be modified with a Rule Editor. The rules are divided into four groups called Grammatical, Mechanical, Style and User-defined. The group Grammatical contains checks for, among others, number agreement, double negative, incompatible verb tenses, wrong use of comparatives and articles and agreement between verb and subject. The group Mechanical is concerned with punctuation, capitalization and spelling. It is possible to maintain a customized version of the Grammatik dictionary. Words can be added and deleted. Style flags words and phrases which, among others, can be considered archaic, cliche, jargon, colloquial, pretentious, redundant, foreign, or gender specific (e.g., maiden voyage). User-defined contains rule classes that have been created by the Grammatik user. In Grammatik the rule classes have been used to create checks on five standard writing styles. Each conforms to a particular subset of the set of rule classes. The writing styles are General, Business, Technical, Fiction and Informal. Dependent on the chosen writing style checks from the different rule classes are enabled or disabled. Obviously, it is possible for the user to define an own writing style by composing a set of rule classes. In order to perform its task Grammatik identifies the parts of speech in a document (noun, article, adjective, verb, etc.). It can display readability statistics that can be used to guide authors with respect to average word length (measured in syllables), sentence length (in words) and paragraph length (in sentences). It can provide the author of a document with a word usage profile. Grammatik issues warnings, e.g. against excessive use of the passive voice in a document. Whether there is a warning depends on the writing style that has been chosen. Moreover, it is possible to obtain readability benchmarks on documents by comparing them with Lincoln's "Gettysburg Address" (clear), a short story by Hemingway (simple) and a life insurance policy (formal). Grammatik allows benchmarking against documents that are chosen or written by the user of the system.

If we return to errors made at the sentence level, then a distinction that sometimes is made is that between performance and competence errors. Typographical errors are typical performance errors. Incorrect knowledge about spelling or agreement rules leads to competence errors. Morpho-syntactical errors are errors that are often related to the derivation and inflection of words. They can be caused by agreement violations (gender, number, person, tense and mood). There are also errors that are caused by homophony, as in 'there houses' versus 'their houses'. Also in cases that people don't know the exact spelling of a word they may write what they hear. Correction of these errors can sometimes be done if transcriptions of words into a phonetic form are available (van Berkel and De Smedt[1988]).

Writing aids like the ones mentioned earlier in this section make errors. Most existing systems apply checks on the level of dictionaries (dictionary lookup) and morphological rules (with exception lists, e.g., for loan words). Correct hyphenation and spelling correction may,
Errors caused by the user

- Grammatical errors: 36%
- Style errors: 23%
- Typing errors: 5%
- Spelling mistakes: 3%

Fig. 1. A diagram showing error percentages.

However, require syntactic and semantic analysis. Only recently modest attempts to include syntactic aspects in these systems can be seen. Reports about research projects that involve research in this direction have been presented by, among others, Vosse[1991] and Heemels[1991]. Vosse offers a systematic and grammar-independent approach to detect and correct morpho-syntactic errors using a shift-reduce parser for (augmented) context-free grammars. Other errors that are dealt with are errors in idiomatic expressions. The results of this research have been implemented in a Dutch grammar checker. One of the conclusions of this research is that most morpho-syntactic errors can be effectively detected and corrected. More about the work of Vosse can be found in his paper in these proceedings. Similar research is reported by Heemels. Research conducted at his company (Océ-Nederland, Venlo) shows a division of user errors as displayed in Fig. 1. The aim of the research is to produce a robust parser, i.e., a parser that continues parsing despite the detection of errors (misspelled or unknown words, wrong segmentation of the input, feature violations, etc.).

To conclude, despite their limitations the aids that are available or that are being developed from the above mentioned research efforts are useful for authors of documents. For professional publishing houses they may, however, be too limited. In the commercially available text-processing systems linguistic knowledge and algorithms for using it are available in an integrated way. As will be mentioned in the next section, commercial language translation tools integrated in well known text processing systems are becoming available. Electronic dictionaries are sometimes offered by specialized companies. This can be done on diskette, e.g. the Prisma Electrische Woordenboek, or on CD-Rom, e.g. Van Dale Lexitron from Van Dale Lexicologie. Especially the availability of this low-cost, high density storage medium makes it possible to offer more information than available in traditional dictionaries. CD-Roms can be used in a variety of applications and systems. For example, they may offer information about spelling, hyphenation, meaning and word usage, information about synonyms and proverbs, information about conjugations and inflexions of verbs, nouns and adjectives, information about word etymology and encyclopaedic knowledge (geographical names, flora and fauna, etc.), scientific and technological terms, etc. Obviously, they offer search software for finding information in a convenient way. Microsoft's Bookshelf is a multimedia system that, among others, contains a dictionary that can produce the correct pronunciation of its words. Machine-aided editing of dictionaries reduces the time to produce and update dictionaries. Automatic construction of dictionaries from documents is an advanced research topic. Dictionaries and spelling checking programs play a role in optical character recognition (OCR). Reading errors can be corrected by the use of these programs. Advanced OCR systems include context analysis routines which use syntactic knowledge. Texirus is an example of an OCR system that has a multilingual lexicon and a lexicon that can be filled by the user. One way to improve a spelling checker or an OCR system could be the use of frequency information, e.g., word co-occurrence probabilities or methods to predict the next word or word category. As a final example of a writing aid we mention Mindreader, a program that generates a full word after a few letters have been typed. The program adapts itself to the user.

Author environments may include many of the tools that have been mentioned in this section. In addition we should mention tools for machine translation and information retrieval that will be discussed in the next sections. If we confine ourselves to the writing process it should be mentioned that there are attempts to use cognitive models to obtain outline or 'idea' processors. Research in the area of computer-supported cooperative work has also attacked the problem of allowing several people connected to a network to work on the same document.

4) Studies on the frequency of certain types of spelling errors can be found in Damerau[1964], Peterson[1980] and Peterson[1986].
4. MACHINE TRANSLATION

Machine translation has always been the show-piece of applied natural language processing. In the fifties and sixties all the attention of the researchers went to 'full' machine translation systems. After the collapse that followed the ALPAC report for some time attention drifted away to the more fundamental fields of computational linguistics and artificial intelligence. With lower ambitions the field was entered again in the late nineteen seventies. At that time work started on interactive systems for machine translation. Despite the adjective 'full' the large systems required a certain amount of pre- and/or post-editing. Nevertheless, the computer as a producer of a 'quick and dirty' translation turned out to be a useful tool. Well known large systems that were developed are the Georgetown's GAT system, Systran, Taux, SPANAM and Metal. Users included Euratom, NASA, General Motors, the Pan American Health Organization and the Commission of the European Communities (CEC). The need for machine translation was expressed by the CEC by starting EUROTRA, a long-term and large-scale machine translation research project, in the early eighties. It has been estimated that in the EEC alone some 100 million pages of text are translated annually. In the Netherlands many people have been involved in machine translation research, especially through participating in the EUROTRA project, the Distributed Language Translation (DLT) project of software house BSO and the Rosetta project conducted at Philips Research Laboratories. An overview of the Rosetta system can be found in the paper by J. Odijk in these proceedings.

As mentioned, interest turned also to the development of interactive tools for the professional translator. A distinction was made between human-assisted machine translation (HAMT) and machine-assisted human translation (MAHT). Machine translation systems and translation tools were marketed by specialized firms, e.g. by Weidner and ALPS. These developments were intensified with the advent of personal work stations and the personal computers in the nineteen eighties. Computer memory became cheap. Rather than measuring memory size in KiloBytes (KB), it was measured in MegaBytes (MB) and on a pc's hard disc it became possible to store 40 MB, 60 MB, 120 or even 200 MB of information. Presently, without any difficulty it is possible to store the information available in dictionaries in a pc's memory. Examples of electronic translation dictionaries are Wordisk (Wolters-Noordhoff), the Canon Wordtank, Vertaal! (Vertaal! Publishers) and Polyglot (Linguistic Systems). Often these systems are more than just translation dictionaries. For example, the Canon Wordtank contains basic words and their translations, but also synonyms, thesauri, programs for spelling control, verb conjugations and hyphenation and, moreover calendar, calculator, notebook, etc. Vertaal! and Polyglot are associated with text-editing programs for pc's, for example, MS-Word or WORDPERFECT. Polyglot is expected to develop into a sophisticated translators aid. It can be used for different language pairs. Once a word has been marked for translation, then the system shows synonyms, meanings and translations of the word in different windows. The user can choose the translation that best fits the meaning of the word in the current context. The 'heart' of the program is the 'linguistic engine', an intermediary between different languages that for each word knows the synonyms and word sort information. In the future a morphological and a grammar module will be added. The first module offers the possibility to deal with verb conjugations and plural formation. The grammar module will give information about sentence construction, word order and rules of agreement. In this way the system aims at goals that were the objective of the earlier mentioned MAHT or HAMT systems on work stations. Rather than having a writer's workbench we should talk about a translator's workbench, which provides the professional translator with multilingual text processing facilities (as mentioned in the section on writing aids), translation support (interactive on-line translation and remote access to a fully automatic machine translation system), on-line access to an application and environment dependent term bank and remote access to large term banks. The TRANSLATOR'S WORKBENCH is an ESPRIT project that aims at building an environment that includes these facilities. Languages that are covered are English, German and Spanish. The languages French and Italian will be added in the near future. For grammar and style checking the METAL parser is used. Remote access to the METAL translation system is provided. In addition there is a trainable translation memory that enables the user to retrieve previously stored translations of phrases and to compose translations of business letters and technical manuals from these phrases. One aim of
the project is to port part of the workbench to the MS Windows environment (Kugler et al.[1991]).

It is interesting to note that research conducted for machine translation purposes is now being used for more 'down to earth' grammar and style checking purposes (Thurman[1990]) and for developing commercial pc-based multilingual terminology software for translators that works in conjunction with the standard word-processing packages. Computer-aided translation products for professional translators are offered by, among others, INK Intl. (Amsterdam) and ALPS. Much interest goes to the translation of limited text, for example, telexes, scientific and technical abstracts, business correspondence and technical manuals.

5. LINGUISTIC TEACHWARE

Grammar and spelling instruction is important in a small language community. Children have to learn different foreign languages and grammatical knowledge helps in mastering English, German and French. Teaching grammar with tutoring systems requires grammatical knowledge by the system. Otherwise, errors made by students cannot be detected, corrected and explained in an informative way. A system has to know about grammatical categories, word order constraints, agreement and grammatical functions in order to diagnose problems and to produce explanatory feedback. Hence, an intelligent tutoring system for grammar and spelling checking requires a sentence parser, a sentence generator, modules for diagnosis, tutoring and generating exercises and an interface to the student. Advanced students should have the possibility to manipulate tree structures and to define syntactic properties of new words. Although dealing with ill-formed input is important for each natural language processing system, it is especially important for a language tutoring system. It has to be 'understood' why the sentence is ill-formed in order to offer appropriate help.

As may be expected, many prototypes of systems that help to teach languages have been build. If we confine ourselves to the Dutch situation, we should mention the intelligent tutoring system TDIDT (Tidtd Diagnoses Trouble with DT), a program based on Dutch morphology and phonology to teach the correct spelling of Dutch conjugated verb forms (Daelemans[1989]) and the 'Schooltekstverwerker' of Vosse[1989], an educational text editing system that is able to distinguish between grammatical and ungrammatical sentences by detecting (and correcting) spelling and grammatical errors through syntactic analysis (parsing). In both cases the systems are built with linguistic tools and models that can be used in many different applications. These systems are research projects. There is also commercially available educational pc software for tasks like these5. Programs like these are sometimes made for individual use by children, sometimes they provide a framework for teachers for making new sets of exercises for their students. Speech technology can be employed to develop courseware to improve the pronunciation by non-native speakers of a language.

6. DOCUMENT PRODUCTION SYSTEMS

For large international companies the production, maintenance and translation of documents is an expensive matter. For these companies automatic translation of texts is part of the more comprehensive process of automatic document handling. This process includes storage, retrieval and manipulation of parts of texts. Language processing is then integrated into a document creating and maintaining process. Enhanced quality of automatic translation or human assisted translation can be obtained by imposing formal structures on the input of a translation system. International companies can impose simple language use in their product documents in order to facilitate automatic translation. Among the tools for document handling are so-called mark-up languages (e.g., SGML or ODA) and associated text-editors that control language use and mark-up demands. ODA stands for Office Document Architecture. It allows the definition of logical structure, layout structure and formatting styles. SGML is the acronym of Standard Generalized Mark-Up Language. Publishers use SGML to automate the publishing process. It uses Document Type Definitions (DTD) for describing the structure of documents. Updating, or even decentralized updating, is made easier and on-demand publishing can be aimed at. An example of a company that is interested in SGML and

5 In the Dutch situation we should mention the programs Rekenen-Taal & Lexmaker (Muiderberg), Taalomibus (DAlhamaat VZW), Zuurontleding (De Vries Software), SpellRaum (Meulenbergh) and Nedercom Spelling (TWS Automatisering).
associated tools (they have been experimenting with the Amsterdam Automatic Proof-reader, a syntax-checker for natural language) is Elsevier's Science Publishers. There are some linguistic projects in which text corpora are encoded in SGML or a language based on it. In the "Tree Bank of Written and Spoken American English" project SGML mark-up is developed to encode linguistic information. Millions of sentences will be annotated with part-of-speech assignment, skeletal syntactic parsings and intonational boundaries for spoken language.

Integrated systems for the production and maintenance of technical documents are already on the market, for example, KEEPS (Kodak), Documenter (Xerox) and Publisher (Arbor Text). They allow the reuse of available documents, they offer support for the development of documents by teams of authors, they allow access to older versions of documents, editing and browsing of documents, etc. Some industries already restrict language use in their maintenance manuals to Simplified or Basic English. By restricting the freedom of expression of the author automatic correction and translation of texts may become feasible. An example of a research project that by restricting form and meaning of sentences aims at these goals is the Vleermuis Machine Translation Project (Van der Steen and Van der Kuif[1991]). The system under development consists of a Language Editor and a Translator. Both are based on (restricted) lexicons, thesauri, grammars and translation rules. More about this system can be found in the paper by G.J. van der Steen en A.J. Dijenborgh in these proceedings. An ESPRIT project that aims at developing an integrated system for the production and maintenance of technical documentation is SPRITE. In addition to the usual facilities it will allow access to multimedia databases and data from an ORACLE relational database can be transformed to SPRITE documents (Hoppe[1991]).

7. INFORMATION RETRIEVAL

In 1945 Vannevar Bush, science adviser of the U.S. government, warned against, what he called, an information explosion. In addition he predicted the appearance of new forms of encyclopedias, ready made with a mesh of associate trails running through them (Bush[1945]). The warning triggered research in information retrieval technology in the nineteen fifties. Special purpose machines for this task were proposed and built. Presently the technology that allows interactive storage and retrieval of information is available in general purpose computers. Systems can be built that allow users to browse through titles and documents and to obtain answers to specific queries. Intelligent interfaces allow the user of a pc or a workstation access to online databases. The interface allows the switching between different appropriate databases, it can dial up hosts and transmit queries via telecommunications, it performs the necessary 'house keeping' activities and it can interact with the user. Interaction can take place through one or more natural languages, restricted and simplified natural language, through natural language menus or through a formal query language that has to be learned by the user. The interface should be able to handle the user's query, to ask a user to clarify his query and to process any answer of the user to this request. The databases that have to be searched may contain factual data, e.g. about employees or customers, or they consist of texts: documents, papers, newspaper stories, medical histories, messages, patents, abstracts, etc. If we have an intelligent interface to just one particular database, then it will be efficient to have a direct translation from a natural language query into a search expression for that database. If the interface is meant to allow access to different databases and hosts it is useful to have a standardized intermediate query language. The intelligent interface maps the natural language query into an expression in the intermediate query language. The host of the database that has to be searched performs the conversion from the intermediate expression to the search expression.

This section is concerned with information retrieval interfaces that are meant to search text databases. In the following section we consider interfaces to (relational) databases and problem solving systems. There is no sharp-cut fundamental distinction between these interfaces. However, in general, talking about information retrieval means talking about different databases containing textual information on a specific topic. When we talk about a natural language interface, the underlying assumption is that it is an interface to one particular relational database.

Information retrieval can be based on a search algorithm that attempts to match the words in a query with words in keyword records. In that case keywords are explicitly assigned to a
document, either by hand or by (semi-)automatic processes of indexing. It is also possible to search all texts to satisfy a single query (full text retrieval). Apart from single keyword string matching it is possible to use language-independent n-gram methods. It would be interesting to see how information retrieval can benefit from research on self-learning systems, like neural nets. An example of neural information retrieval with tri-grams is presented in Scholtes[1991]. In order to reduce the enormous amount of data that can match a query, some way of filtering or ranking of the information is useful. This can be done by associating weights with the terms in a query. Nevertheless, it can be expected that a search performed by a human being, given sufficient time, will yield better results than can be obtained with an automatic retrieval system. Judgements about what is relevant and what is not are based on knowledge of the domain and not on tri-grams or keywords. In order to enhance the quality of retrieval mechanisms for meaning determination have to be added.

The conventional way of information retrieval is to create an inverted file containing the significant (stems of) words in texts. These texts can be titles, abstracts, patents, messages or full papers. A user's query can be matched against the words in the inverted file. The query can be expanded by including synonyms from a thesaurus for the terms in a query. However, there are more advanced ways to process queries and texts. Texts can be indexed, i.e., certain information can be tagged to the words in a text. The aim is to improve the process of finding significant words and of determining whether in their particular context they are significant. There exist methods of assigning parts of speech information to individual words in a text. In order to increase the correctness of these methods context information and statistical information can be used. Words and phrases can be matched against a semantic dictionary that allows the assignment of words and phrases to semantic categories. Obviously, the semantic categories can be domain-dependent, as might be the case in a domain of chemical abstracts or patents on chip design. Once we have semantic categories assigned to a domain then it is also possible to define properties of the concepts and relationships between them. A frame-based semantic network with various relationship links can underlie a textual database or part of a textual database. Full syntactic and semantic analysis of the sentences of the texts in a database may help in assigning correct semantic categories to words and phrases in a text. For certain restricted domains the use of 'skimming' techniques (see the previous section) that aim at obtaining a frame-based semantic network representation of texts may be more realistic. In this case relevant concepts, properties and relationships are pre-defined. Possible relations are IS-A, CAUSE-OF, and PART-WHOLE relations, but we can also think of including synonyms, spelling variants and co-occurrence relations. This allows the replacement of terms in a query by terms that are semantically related (synonymy, broader and narrower terms and otherwise semantically related terms) or that are related through spelling, homonymy or co-occurrence.

Obviously, if there has not been a detailed analysis of the texts in a database there still is the possibility to perform an analysis of the query that yields a frame-based semantic representation and that can be used to narrow or broaden a search or to obtain otherwise semantically related terms or paragraphs of texts. During analysis the query should be subjected to spelling correction and in interaction with the user a search has to be tuned. The searcher may have the possibility to browse a semantic network or the system can suggest terms that can be used to denote concepts that underlie the domain of queries. The searcher can be asked to disambiguate query terms or phrases by conducting a dialogue by the system with the user.

There exist several commercial front-ends that help a user to perform a search on different databases and hosts (e.g., SCI-Mate and Tome Searcher). A survey of intelligent interfaces for information retrieval is presented in Vickery[1988]. Some prototypes and commercial systems that carry out syntactic analysis of free text have been built. COPSYS (Content Operator SYstem), a system developed at Siemens Corporate Research Laboratories in Munich, performs a syntactic analysis that yields dependency structures of noun phrases (Ruge and Schwartz[1989]). SIMPR (Structured Information Management: Processing and Retrieval) is an ESPRIT-project on advanced techniques for information retrieval and management in text databases. Morphological and syntactic analysis is applied to all sentences in a text. This analysis is meant to find the 'important' words or phrases in the text, from which an index will be compiled. Semantic analysis is beyond the aims of the project. ITI-TNO, Delft and Cap Gemini
Innovation, Rijswijk are among the participants in the project. More about SIMPR can be found in the paper by C. Barkey in these proceedings.

The birth of the new forms of encyclopedias that were predicted by Bush took place in the nineteen-eighties. In this decade we saw the advent of machine readable dictionaries, encyclopedia on CD-ROM and attempts to introduce semantic structure on this stored information. Semantic structure is important for use of the information in natural language processing tasks and for information retrieval purposes. Structure on the level of text chunks is available in hypertext systems. Here we have networks of text fragments and means to traverse the network, make selections and explore cross-references. Hypermedia systems allow the navigation around a network of multimedia nodes, nodes that consist of text fragments, graphics, digitized speech, audio and video recordings, pictures and animation. Creating and traversing trails in a network of multimedia information based on spoken language requests of a user is a research area which until now has not been explored. Rather than having hypertext available on one (personal) computer, we can think of hypertext or hypermedia communication systems where encyclopedias, libraries and wire services are available in a network of computers. From where the information is retrieved should be of no concern of the user.

8. NATURAL LANGUAGE INTERFACES

General natural language interfaces that can be used for any domain of discourse and any type of discourse are not available and will not be available before the year 2001. An interface that can be talked to without any restrictions can be built, but it will not provide human-like reactions. It would need all common sense knowledge that is possessed by a human being and it should not only be able to handle literal meanings of utterances but also tropes like irony, understatement, overstatement, simile, metaphor and metonymy. For these reasons we have to talk about domain-dependent interfaces and domain-dependent language use. A sublanguage is the language used by a community of language users in discussing a restricted domain. Before building a natural language interface it is necessary to agree about the sublanguage that will be covered. A representative sample corpus is needed. The corpus can be analyzed by hand and a grammar and lexicon can be constructed that covers the corpus. In practice this turns out to be an extremely difficult and time-consuming task. Semantic information should be extracted from the corpus in order to reduce the number of semantically anomalous parses. It is also possible to gather semantic information through interaction with the user. Feedback by the user makes it possible that the system, by using frequency statistics, itself rejects certain parses in the future or performs a kind of preference-based parsing. One approach to the problem of constructing a grammar and a parser is to use a broad coverage natural language grammar and its parser. However, it is not only unlikely that the grammar indeed covers the corpus, but is also probably the case that the parser is too inefficient to be used in a natural language interface. It will be too slow and it will produce many parses that are irrelevant for this particular domain. One solution is to trim the broad coverage grammar. Use its parser on the corpus, eliminate the irrelevant parses and then collect the grammar rules that have been used in the remaining parses. They constitute a grammar for the sublanguage. It may be necessary to add some extensions in order to deal with idiosyncratic syntactic, semantic or discourse properties of the domain.

Natural language interfaces should be robust. A spelling and grammar corrector should make it possible to handle ill-formed questions or answers. However, grammars describe sentences, not dialogues. If the natural language interface is meant to support a dialogue between human user and system then it should allow input that can be considered as ungrammatical at the sentence level, but that is allowed in the context of a dialogue. Fragmentary or telegraphic style input, ellipses and anaphora are dialogue phenomena that have to be dealt with. If necessary, the interface has to interact with the user to confirm assumptions and to resolve remaining ambiguities in the input. In that case presenting a natural language paraphrase of the query by the system is useful. Users, on the one hand, sometimes give unsolicited comments or over-answer questions, or on the other hand, sometimes give very elliptical answers. A system is sometimes expected to provide more information than literally requested. A system is expected to be co-operative, to infer intentions from utterances and to detect and rectify misconceptions of the user about the capabilities of the system with which he is communicating.
order to live up to a user's expectations a system can maintain a model of a user or a group of users, that is, a knowledge source of assumptions on the user's behaviour in the dialogue. In addition one may ask that systems tune themselves to novice or expert users.

In many cases a natural language interface takes the form of a language understander that translates queries into database commands for a relational database system such as Oracle and a dialogue facility to interact with the user to resolve ambiguities in the queries. In this case we can make use of the structure of the database to define semantics. The main goal of the system is to supply information. This is different from an interface to an expert system, where the user statements have to be mapped into the facts of an expert system and the system is meant to solve a problem. Whatever the backend of the interface is, for each domain a new interface has to be build. As with other grammar-based natural language processing products development of natural language interfaces can be speeded up by grammar development workbenches (see section 2).

It is possible to distinguish between domain-independent tools for building and composing the natural language interface and domain-dependent tools or parts of the interface. This separation should be explored to its limit in order to reduce the effort to construct natural language interfaces and to achieve portability of the interface from one domain to another. Portability aims at reducing the efforts to create domain-dependent modules.

In natural language processing talking about portability means talking about accomplished systems. Examples are LIFER (Hendrix[1978]), TELI (Ballard[1986]) and TEAM (Grosz[1987]). In LIFER the domain-independent part is a parser and an interactive language specification facility. LIFER's grammar is a semantic grammar. Its rules describe phrases of sentences in terms of semantic categories. The specification facility allows the user to define (semantic) grammar rules, semantic rules associated with the grammar rules and words. The semantic rules associated with the grammar rules convert user's questions into database queries. TELI (Transportable English-Language Interface) maintains case frame information and has modules for the acquisition of syntactic and semantic knowledge. Customization by end users can be done at any time. TEAM (Transportable English Access Data Manager) interacts with a database expert in an acquisition dialogue to obtain the information about the fields of the database. It can interact with end-users to provide answers to queries.

Acquisition means learning what words will be used to describe the contents of the database, which (and how) questions can be asked, what adjectives are allowed, etc. It is not necessarily confined to the contents of a database. For example, syntactic properties of a new (phrasal) verb can be found out by a system by presenting the user a list of sentences in which the verb is used. The user is invited to point out the ones that are grammatical. In general one might say that in order to configure a portable natural language interface to a new domain (or database) it has to be taught about this domain. That is, it should learn about the concepts and their relationships, the vocabulary that will be employed in the queries and the mapping from the domain model to the database. At BBN several knowledge acquisition tools for this task have been developed. Some of these tools focus on (relational) databases, others focus on expert systems or knowledge bases (e.g., KNACQ: KNowledge ACQuision). The Learner tool has been developed to configure a portable natural language interface to a particular database. It is a software tool for creating the knowledge bases, vocabulary and mappings to the particular database.

The systems that were mentioned above were flexible in their lexicon. One might think of systems that are based on domain-dependent grammars. For example, in a dialogue system questions and imperatives will occur more frequently than assertions. Moreover, syntactic constructs can differ for different application domains. Hence, writing a new grammar for each new application will probably lead to a more efficient system than using a general lexicon and grammar. In the context of the PUNDIT system (see section 2) techniques for automatically pruning a general lexicon and grammar have been investigated. From a large set of sentences from a particular domain (a direction assistance dialogue system) the syntactic constructs, lexical items and associated feature information were extracted in order to obtain a minimal grammar and lexicon from a more general grammar and lexicon. One step further means that in addition of attempting
to adapt a system to a particular domain, the
system itself may adapt itself to a particular user
by maintaining a user model. In these proceedings
natural language interfaces are discussed in the
papers by J. Honig and by A. van Rijn. Honig
designed a natural language translation system
with components that can be used to form a
natural language front-end to a database or an
expert system. Van Rijn reports about her design
of a user interface to a system that automatically
derives a program for the construction of a
product by a flexible assembly cell from a CAD
drawing.

Early examples of commercially marketed
natural language interfaces to databases are
INTELLECT (1981, Artificial Intelligence Corp.)
had to be used on IBM mainframes
(an IBM pc had only 64 Kbytes internal memory)
and cost £70,000. THEMIS was available for
£24,000. Presently, there exist commercial
building tools for developing natural language
interfaces, there exist commercial integrated
natural language interfaces (in a DataBase
Management System) and there exist commercial
optional integrated natural language interfaces.
Among the building tools are systems with prices
ranging from £900 (Language Workbench, Brodie
Assoc.) to £80,000 (Easytalk, Intelligent Business
Systems Inc.). An early example of a commercial
development environment is LanguageCraft
designed by Jaime Carbonell. Apart from a
grammar writers workbench it has a caseframe
parser and support to connect it to different
applications. A well known example of a
commercial integrated natural language interface
is Symantec’s Q&A (Question and Answer,
distributed in the Netherlands by Kernsoftware,
Leiden). Q&A offers an integrated environment
which contains a text-editor, a database
management system and a natural language
interface, the Intelligent Assistant (IA). IA makes
it possible to access a user-constructed database in
natural language, rather than in a formal query
language. It is not interested in the grammatical
correctness of questions that are asked. Its aim is
to convert questions into intended queries. The
interpretation that is determined by the Intelligent
Assistant is shown on the pc’s screen for
confirmation. New words can be introduced by the
Q&A answer. Users sometimes express doubts
whether the efforts needed to learn to manage a
not always intelligent IA outweigh the efforts to
learn a formal query language like SQL. A similar
example is Hal developed by Lotus Development
for its spreadsheet program Lotus 1-2-3. Natural
Language, a language product developed by
Natural Language Inc. (Berkeley, Ca.), allows
users to access relational databases (Oracle,
Sybase Ingress, etc.) in a natural language.
Questions are parsed and translated into SQL
statements. A front-end with similar properties
will be offered by IBM. It is called
LanguageAccess and it parses questions and
commands into SQL statements. The system is
based on syntactic and semantic modules for the
analysis and the generation of natural language in-
and output. It can handle referents in the same or
previous questions and in case of ambiguity it
prompts users to indicate correct interpretations.

Any buyer of a natural language interface
should consider the syntactic and semantic (or
conceptual) coverage of the system, its ability to
deal with elliptical utterances, anaphora and ill-
formed input. Moreover, for database query it is
also possible to learn a formal query language or
to use a menu-based system. There exist systems
that constrain the user in selecting phrases from
menus to compose sentences. An example of a
recent system that allows users to construct SQL-
queries without having to learn SQL is Quest from
Gupta Technologies. SESAME is Bull’s system for
natural language access to relational databases.
Also in this case, rather than being an interface it
is an environment to build a natural language
menu system. Through menus the words and
phrases that make up a query are proposed. The
query is first translated into a logical form and
then into a SQL query. NaturalLink of Texas
Instruments is a similar interface.

9. SPEECH PROCESSING SYSTEMS

Just as in the case of machine translation the
history of speech recognition and synthesis can be
marked with large projects, often supported by
DARPA. Among them are the Hearsay, Harpy
and Dragon systems (all from CMU). Presently, in
a well known large CMU project the Sphinx
speech recognition system is developed and
improved. Current systems employ techniques to
predict the next word or next word category and
syntactic and semantic knowledge is used in word
recognition. Present goals in speech processing
research are systems that have large vocabularies,
are speaker-independent and can handle
continuous speech rather than isolated words. In
real applications systems can encounter speakers that mispronounce words, use interjections, restart phrases and make grammatical errors. Worse, especially for defense applications, for example cockpit systems for pilots, systems should be robust, i.e., function in the presence of noise and be speaker stress-resistant. In speech synthesis prosody is an interesting topic of research. Synthetic speech is monotonous. In order to have 'natural' synthetic speech, prosodic factors (stress, timing and intonation) have to be taken into account. In speech generation syntactic, semantic and pragmatic factors should influence the 'melody' of the output. In speech recognition the question is how the available information about the tune of a spoken sentence can be exploited during parsing for the resolution of ambiguities. The relationship between syntactic surface structure and intonational structure is an important research theme. In the ESPRIT/POLYGLOT project it is studied in the context of a multilingual text-to-speech system.

Speech processing can be integrated with language understanding and speech dialogue facilities. An attempt is SPICOS, a research project of IPO, Eindhoven and Siemens, München. Other projects aim at text-to-speech conversion (systems that scan text and produce speech or systems where the input is entered with a keyboard) or real-time translation of telephone talks, i.e., from speech to speech or from speech to text displayed on a screen.

In the past Philips and the Dr. Neher Research Laboratory in the Netherlands have developed Audiotel, a system for giving information by telephone. However, it has never been put on the market. Apart from research at universities speech processing R&D in the Netherlands takes place at IPO (Eindhoven) and PTT-Research (Leidschendam). One company that is heavily involved with speech and natural language is Bolt, Beranek and Newman (BBN). In the 1970s it produced HWIM (Hear What I Mean), a system that integrated speech recognition and natural language understanding. A more recent project on continuous speech recognition is BYBLOS. Like so many other systems it is based on hidden Markov models of phonemes. In BBN's HARC system the emphasis is again on integration of speech and language understanding. The acoustic part produces a lattice of possible words that is passed to an Earley-like parser which produces all possible parses. The parses are translated by a multi-level semantics component (derived from the PHLIQUA system, see Bronnenberg et al[1980]) to identify the meaning of the sentence. The grammar that is used for the parser is unification-based, i.e., context-free grammar rules that are annotated and during parsing feature sets are unified. Companies have been founded that sell speech processing products and whose R&D departments aim at defense and industrial applications. Systems are offered as information aids to the blind, for voicecommand or for on-demand speech information. Examples are systems for travel and traffic guidance, for monitoring and for assessment. Well known products are CSSL's Monologue (a proofreading assistant) and the Kurzweil Reading Machine. The latter is based on OCR techniques. The user can select text to be read or re-read; it is possible to repeat sentences and to spell out words. Phonetic Engine is a speech recognition system developed by the Californian company Speech Systems. There are systems that deal with financial transactions by bank customers, systems that allow users natural language access to databases by telephone (e.g., the Dectalk Voice Response system) or systems that give information by telephone (e.g., Periphonics). Some years ago IBM was experimenting with a system called TANGORA, implemented on a pc, which transcribes speech input to text in real-time. The system requires that users pause between words and it must be trained to the user's voice. Presently IBM's VoiceType, originally designed for disabled people, gives voice control over DOS computers and their word-processing systems. A similar example is a voice-based spread-sheet system. Dragon Systems markets different speech processing products, among them DragonWriter and DragonDictate. The former system is a speech recognition system that runs in a 386-based MSDOS pc. DragonDictate is a voice-controlled dictate system for IBM-compatible 386 computers. Sound systems for IBM compatibles that include text-to-speech software can be obtained for a few hundred dollars.

10. KNOWLEDGE EXTRACTION AND ACQUISITION

In order to extract knowledge from a natural language text it is useful to know about the domain of discourse. If it is possible to define the kind of knowledge we are interested in, texts can be scanned using expectations for particular
information. The data structures that were mentioned in section 2.4 can allow an incomplete but nevertheless useful representation of the information available in a natural language sentence or a (short) natural language text.

Among the many applications that can be thought of are those in the area of law and legislation, patents, press reports, message handling, information extraction for information retrieval use, sorting electronic mail, etc. Moreover, when we talk about constrained domains we can also mention the domain of dictionary entries (section 2.3), texts in medical records, plant or mushroom descriptions, equipment status messages, etc. In the paper by B. van Baâkel in these proceedings the linguistic module of an information extraction system for chemical texts is discussed. There are systems that process banking payment telexes (Lytinen and Gershman[1986]). Obviously, extraction systems can be connected to continuous sources of on-line information. Generale Bank, the largest banking-institution in Belgium, has been investigating the use of natural language processing techniques for this purpose since 1984. In 1985 it founded together with Roger Schank's Cognitive Systems a joint-venture (GECOSYS) to develop, among others, the Telex Reader. It extracts the relevant facts from a payment telex, represents them in a conceptual dependency notation and translates the message in a standard format (the SWIFT format). All translations are checked by a human operator. The system has been sold to many banking-institutions all over the world. Another well known system is NOMAD. It edits cryptic, errorful naval ship-to-shore messages. The automatic creation of a database is one of the goals that can be pursued in this type of information extraction.

In SCISOR (System for Conceptual Information Summarization, Organization and Retrieval) top-down and bottom-up strategies are merged (Jacobs and Rau[1988]). Their system extracts information from newspaper stories in the domain of corporate mergers and acquisitions, stores it in a conceptual knowledge base and is able to answer questions about that information. The sources of top-down information are role-filler, event and world knowledge expectations. The fourth source of information is a full bottom-up parser that identifies linguistic relationships and passes them to the expectation-driven components. Other projects with the aim to build or update knowledge bases by computer analysis of natural language texts have been started by several IBM Research Centers (Heidelberg, Stuttgart, Rome). In these projects we see the use of attribute and unification grammars for analysis and Sowa's conceptual graphs and Kamp's Discourse Representation Theory for semantic representation. These research projects take as examples sections of tourist guides, law-criminal codes on traffic violations and press agency releases on finance and economics (Bollinger et al[1990], Antonacci et al[1989]).

11. NL Specification and Design

There have been attempts to construct systems that allow a programmer to compose a program from natural language explanations about the task that has to be performed (natural language programming). Natural language programming, suggested by Jean E. Sammet in 1966, received some attention in the nineteen-seventies (see e.g. Heidorn[1976]). From an English dialogue the system acquires a description of the problem. The dialogue can be structured by using a semantic model of the domain. The specification will be converted into a program in a programming language. Rather than continuing this line of research interest turned to natural language specification of systems rather than of programs and knowledge acquisition for configuring systems through natural language dialogue. For example, in addition to the already existing tools for creating a conceptual model for an information system, a natural language interface, possibly integrated with a graphic interface, may help in increasing the productivity of an analyst. Such systems may turn into tools for end users to develop their own applications. As an example we mention an intelligent interview system for the conceptual design of a database as discussed in Kawaguchi et al[1986]. The underlying idea is that many researchers in various field have a wish to construct databases for their studies. The system they have constructed takes the role of a human database expert who interviews the potential user. The system's parser converts the English sentences of the user into a plan structure, a model of the domain. By showing sentences from the plan structure the system asks the user to confirm its interpretation of the user's wishes. Another example of a design of a system that performs requirements analysis, design and implementation with the help of a natural
language interface is described in Black[1987]. Here we find an integrated natural language and graphics environment to develop information systems, among others according to the NIAM approach with its Information Structure Diagrams. The interface employs two windows, one for text and one for graphics. If the input takes place in the graphics window then a corresponding text appears in the text window. And visa versa. A final example is GRAMMARS, developed by PANDATA and Fokker (Dijk et al.[1989]). It translates descriptions in a restricted natural language into tables in a relational database.

More fundamental research that aims at obtaining linguistic tools to support the design of conceptual models has been reported by Dignum et al.[1987]. They describe a prototype system that takes a natural language description of a certain Universe of Discourse and extracts structural information and integrity constraints. Weigand[1989] attempts to unify the 'deep' structure of natural language and the conceptual model of a knowledge base by using the 'deep' structure of Dik's Functional Grammar as a knowledge representation language.

12. MULTIMEDIA INTERFACES

In order to improve human interaction with machines interfaces should be natural and effective. Intelligent multimedia interfaces use graphical displays that are integrated with natural language dialogue. Dialogues can be conducted by speech and by typed natural language input. It is possible to talk about objects in a window on the screen and with a touch-sensitive screen or a mouse or light pen verbal descriptions and pointing gestures are integrated. A pointing device can be used to point at objects or to encircle regions.

Intermixed voice input and pointing can be an efficient way to manipulate objects on the screen or to ask questions about them. Applications that have been dealt with in research projects include the selection of regions of geographic maps, the creation and manipulation of objects (e.g., parts of parse trees or charts) and filling out tax and registration forms. Other applications one can think of deal with technical drawings, e.g. as they appear in maintenance manuals, user scheduling, organization layouts and educational software.

13. VIDEO- AND AUDIOTEX

The combination of data processing and telecommunications has led to the videotex and audiotex services. They represent a potential market for speech and natural language processing technology. The first professional users were the airlines, the travel industry and the providers of financial information services. Systems for videotex and audiotex facilitate booking and confirmation services, they allow home banking and tele-ordering (electronic shopping). Mass-market usage of videotex started with the Minitel and the later Kiosque concept in France. Videotex uses a keyboard and a menu-driven display of services. Audiotex uses a (touch-tone or multi-frequency) telephone and the user is presented an audio menu from which he can choose by entering data via the telephone. The speech component of the system is driven by stored segments of digitised speech. Especially when the video and audio interfaces become more user-friendly one may assume that usage will increase. Speech recognition and language understanding technology can produce the necessary tools for improving the user-friendliness. Apart from the customers of the services that are provided, there are the service providers who have to design, compose and update databases. Changes will be frequent and have to be done by people who are not database experts. One may expect that the same people will have the opportunity to create new data bases, preferably from a natural language specification. The process of creating and modifying spoken text (e.g. for advertisers of the 'Talking Yellow Pages') may amplify the research efforts on text-to-speech systems.

The mass production of text and speech will increase the need of software that weeds out undesirable messages that are received by a text message or voice message mailbox. Software that automatically makes abstracts of certain newspaper stories (of an electronic paper) or clusters stories with similar contents from different sources may help in mastering the growing mass of on-line textual information that will become available.

14. COMPUTER-SUPPORTED CO-OPERATIVE WORK

Natural language plays a role in computer-supported cooperative work. We confine ourselves
to a few examples that appear in the literature. Krzyszkowski[1988] describes a system that provides communication and management support for large software engineering projects. The system is based on a taxonomy of communication acts (cf. Searle's speech acts), such as questioning, complaining, informing and planning, while interacting with a system. The general idea is that problems in software development are, among others, caused by communication breakdowns. An environment that enhances communication among designers is therefore desirable. Different communication acts have to be distinguished and processed. Processing can amount to recording or adapting information, asking for more relevant facts, desimation of information to users or other designers, etc. Winograd[1988] advocates a language/action perspective on human cooperative activity and on systems that support this activity. The perspective emphasizes pragmatics, the existence of structure in conversations and speech acts that participate in conversations. This perspective has been the basis for a conversational system for communications in sales, finance and management, built by Action Technologies, Inc., and part of a workgroup productivity system. The system does not deal with natural language understanding. It has converse menus that provide options in a conversation and it maintains and allows the user to retrieve the status of a conversation. Just as there are different speech acts, different conversations are identified.

Acknowledgements: I am grateful to Wilco ter Stal and Theo Vosse for their detailed comments on an earlier version of this paper.

15. LITERATURE


Wahlster, W. [1989]. User and discourse models for multimodal communication. Conferentie-