An Intelligent Computer-Assistant for Stylistic Instruction

Julie Payette*

Department of Electrical Engineering, University of Toronto, Canada, M5S 1A4
e-mail: julie@cs.toronto.edu

and

Graeme Hirst

Department of Computer Science, University of Toronto, Canada, M5S 1A4
e-mail: gh@cs.toronto.edu

Abstract: This article describes an intelligent computer-assisted language instruction system that is designed to teach principles of syntactic style to students of English. Unlike conventional style checkers, the system performs a complete syntactic analysis of its input, and takes the student's stylistic intent into account when providing a diagnosis. Named STASEL for Stylistic Treatment At the Sentence Level, the system is specifically developed for the teaching of style, and makes use of artificial intelligence techniques in natural language processing to analyze free-form input sentences interactively.

An important contribution of STASEL is its ability to provide stylistic guidance according to the specific writing goals of clarity and conciseness. In an attempt to remedy some of the deficiencies of existing instructional software, STASEL's design demonstrates how stylistic instruction can be effectively computerized, while laying the groundwork for the creation of intelligent tutoring systems for teaching writing.

Julie Payette received her B.Eng. in electrical engineering from McGill University (1986) and her M.A.Sc. from the University of Toronto (1990). She has recently been selected as a Canadian astronaut, and is currently training at the Canadian Space Agency in Ottawa.

Graeme Hirst is an associate professor of computer science at the University of Toronto. He is the author of Semantic Interpretation and the Resolution of Ambiguity *(Cambridge, 1987), and many papers on natural language understanding.*

Key Words: intelligent computer-assisted instruction, intelligent tutoring, instructional systems, ITS, computer-assisted language instruction, natural language understanding, stylistics

1. Introduction

Learning to write a language well can be a long and arduous process. Correspondingly, teaching writing techniques requires patience, insight, and a solid background in the structures and rules of the language being taught. Much emphasis is usually given in language classrooms to the teaching of grammar, but we sometimes overlook the importance of style[1] and its effect on the resulting composition.

Style remains a fundamental characteristic of good writing, but the task of mastering it does not come easily. Indeed, writing with style is a skill that must be perfected through practice and experience. Yet before students can develop a style of their own, they must be taught the rudiments of stylistic effectiveness. In other words, they must learn the principles and conventions of the English language, beyond those of grammar, that contribute to achieving good style in formal writing. Stylistic rudiments, in themselves, do not ensure that the resulting prose will be coherent and effective, but they represent an essential step in the acquisition of basic composition skills.

The system implemented in this research is a prototype sentence-level tutor program that aims at teaching stylistic rudiments and at providing guidance as to how the syntax of English influences the stylistic shape of a sentence. Named STASEL for Stylistic Treatment At the SEntence Level, this prototype demonstrates how computerized parsing and sentence analysis can be tailored to stylistic instruction purposes.

STASEL is an interactive system that accepts a student's composition one sentence at a time and responds with a set of stylistic messages, a diagnosis of clarity, and a description of the sentence structure. Example 1 shows a typical STASEL response for a clear and stylistically adequate sentence, while Example 2 demonstrates how the system responds to a structurally awkward and stylistically poor input. (A more thorough description of the output format of the examples is given in later sections).

STASEL was developed to provide tutorial instruction for students of English who are beginning to acquire stylistic principles, but are also able to construct complex syntactic patterns. Although the tutorial is intended for any English composition class, its emphasis on structural elements makes it particularly suitable for second-language teaching classes in which instructors wish to emphasize the subtleties in the arrangement of sentence components.

```
-----------------------------------------------------------------
ANALYSIS OF:  John, my brother who is tall, is an engineer.
-----------------------------------------------------------------
 <=> GOOD: stylistically correct
-----------------------------------------------------------------
 <=> SENTENCE IS CLEAR: positive interruption
-----------------------------------------------------------------
 <=> Structure: simple clause with interruption

    interruption -->
        apposition: np with relative clause

    clause -->
        subj: simple noun phrase (np)
        comp: simple noun phrase (np)
-----------------------------------------------------------------
```

Example 1

```
-----------------------------------------------------------------
ANALYSIS OF:  Surely, I will definitely never personally show you
              the solution of the problem that was given to the
              class.
-----------------------------------------------------------------
  *  1 passive detected

  *  excessive number of adverbs (4) detected in the sentence

  USAGE--> "definitely" in the sense of -certainly- or -clearly-
           has been devalued by overuse

  WARNING: use negative ("never") only if necessary for emphasis
         or contrast. Otherwise, use a positive form

 [W] restrict the use of vacuous words such as "surely"

 [W] restrict the use of vacuous words such as "personally"

 [W] the "solution of" is a weak construct which says
     nothing. It is better avoided

 [W] wordy passive construct in relative clause
     Adjectival wordiness occurs when the writer uses a relative
     clause to introduce a participle that could be attached to
     the noun directly. Occasionally, clarity and emphasis justify
     writing out the entire clause, but in most cases, it is
     better to simply drop the relative pronoun and be auxiliary.
-----------------------------------------------------------------
 *** sentence is unclear ***

  >> failed resolution
  >> excessive number of adverbs
  >> complex structure in the predicate
-----------------------------------------------------------------
 <=> STRUCTURE: initial modifying component + main clause

    initial modifying element -->
                              adverb

       clause -->
           subject: simple noun phrase (np)
        complement: simple noun phrase (np)
        complement: complex >> embedded structure in
                              prepositional phrase
-----------------------------------------------------------------
```

Example 2

Though it can also perform some of the functions of a conventional "style checker" (that is, flagging diction and usage errors in texts), STASEL goes well beyond such systems. It departs significantly from other experimental or commercial stylistic processing programs, such as the UNIX *Writer's Workbench* (Macdonald *et al.*, 1982) or IBM's CRITIQUE (Richardson and Braden-Harder, 1988), in that it displays all of the following features:[2]

- it bases its analysis upon the writer's stylistic intent; that is, the system performs a goal-directed stylistic analysis;
- it is based upon a comprehensive parser that can recognize stylistic features and analyze an

important set of English structures (see section 3.2 for a subset of the possible structures recognized by the system);
- it generates explicit representations of a sentence's syntactic structure in the form of parse trees, one of which, the stylistic parse tree, is specifically constructed for stylistic analysis and for teaching the mechanics of English sentence construction;
- it is designed for educational purposes and provides a communicative environment where the student can "learn by doing."

The prototype's design is highly modular, thus facilitating the extension and refinement of implemented features, the addition of new functions, and the integration of student models and interfaces. For the purpose of demonstration, STASEL's current implementation provides guidance regarding two categories of syntactic stylistic variants: diction (word choice) and sentence construction, and analyzes sentences according to two writing goals: clarity and conciseness. Because of its modularity, the system could be extended to include other stylistic goals such as emphasis, formality, or even deliberate obscurity.

STASEL's most important contribution to the field of computer-assisted language instruction lies in its ability to provide stylistic guidance according to specific writing goals. The system not only detects conventional points of style such as usage, wordiness, and positional problems, but also explicitly identifies the building elements of the input sentence. A formal definition of goal-directed syntactic style (based on DiMarco [1990]) a precise analysis of the inner structure of a sentence allows the system to comment on the stylistic shape of the input according to precise stylistic goals.

Moreover, STASEL has been designed to be part of the architecture of an intelligent tutoring system (ITS) for teaching writing, which would emulate the role of an experienced tutor. Such an architecture would include integrated user interfaces, thorough instructional content, and sophisticated student models (see Neuwirth [1989] and Payette [1990] for a description of an ITS architecture for teaching writing). The development of ITS expertise is an attempt to remedy some of the deficiencies and limitations of existing computer-assisted language instruction (CALI) technology. To date, CALI programs have made only limited use of artificial intelligence (AI) techniques, and for relatively simple language tasks. This research is premised on the notion that AI techniques will enable programs to tackle language instruction more "intelligently," so that in the future, programs will be able to communicate meaningfully with the students with appropriate, individualized instruction based on their past performance. The resulting intelligent language courseware will provide effective interactive guidance to students, thereby relieving the instructor of repetitive remedial tasks.[3] The intent of ITS research is not to replace the language instructor altogether. A program like STASEL is an instructional tool that can interact successfully with a student, but that still necessitates the direction and supervision of an instructor.

The remainder of this paper presents a brief overview of the field of ICALI and describes the design and implementation of STASEL. Selected examples of the system's output have been included. In particular, section 4 concentrates on the stylistic processing capability of the system. Concluding remarks discuss the strength and limitations of STASEL while laying the foundation for future research.

2. Research in ICALI

Intelligent Computer-Assisted Language Instruction (ICALI) is a sub-field of artificial intelligence that draws on expertise from computer science, linguistics, language pedagogy and cognitive psychology. Like other research endeavours in computational linguistics, ICALI is not only concerned with the fundamental properties of language, but also with developing practical applications and computerized emulations of language-related behaviours. In contrast with earlier CALI efforts, which mostly produced automated and rigid drill-and-practice exercises, "intelligent" CALI aims at providing the student with flexible and communicative instructional software through the use of AI techniques in knowledge representation and natural language processing.

Our review of present CALI software (Payette, 1990) found that the vast majority of available teaching programs present a number of important deficiencies. In general, only a limited approximation of the knowledge a teacher possesses about teaching a subject is incorporated into these programs. The questions and answers are determined in advance ("canned"), and the programs leave little control and flexibility to the learner. Moreover, it was found that conventional CALI programs rely on strict accuracy of response, and thus, are commonly unable to accept partial answers or accommodate the different levels of ability and creativity found in students. In all, the degree of real interaction with the learner is highly restricted.

The work under way in the field of ICALI, while trying to provide an answer to present inadequacies, aspires beyond the conventional approach of drill-and-practice programming. New interactive platforms on which CALI design can be based, such as microworlds, intelligent text editors, and dialogue systems, have received much attention in computer science. Most of the recent ICALI designs make use of AI techniques in natural language processing and knowledge representation, focusing on areas of language processing where such techniques can be applied and tackling important aspects of language teaching with greater success than is possible with simple, unintelligent programs, or even through classroom instruction. In fact, many ICALI researchers believe that the incorporation of AI and NLP strategies has the potential to radically improve the efficiency and communicative ability of current and future instructional programs (Last, 1989; Underwood, 1989; Bailin and Levin, 1989; Chapelle, 1989; Neuwirth, 1989; Farghaly, 1989; Mulford, 1989; Hamburger, 1990; Ferney, 1989; etc.).

The prototype described in this article follows the philosophy of intelligent tutoring systems in that it addresses a subset of the knowledge and procedures that constitute intelligent language tutoring. More precisely, STASEL provides a subset of the expected components of the ITS models,[4] notably,

- the linguistic component through a natural language processing (NLP) interface that is composed of a parser, a lexicon and a response generator;
- the syntactic and stylistic analysis portions of the ITS's expert module.

Yet also, STASEL has been designed to facilitate the insertion of additional ITS components such as an intelligent grammar checker, a tutorial strategy, a student model, and a natural language response generator.

Though still in its infancy, the potential of ICALI research is widely recognized in the literature and is attributable, not only to the increased degree of flexibility and control it can offer the student, but also, to the potential benefits ICALI can bring to language instruction as a whole (Bailin, 1988; Bailin and Levin, 1989; Burston, 1988; Duchastel, 1988; Underwood, 1989).

3. System Description

STASEL is an interactive system that analyzes the syntactic and stylistic features of sentences for the purpose of language instruction. As input, the system accepts complete and grammatically correct sentences of which it analyzes the lexical, structural, and positional characteristics. (In its present state, STASEL does not include grammatical and spelling corrector modules, but its modular design ensures that the incorporation of such modules is readily feasible.)

STASEL concentrates its instruction at the sentence structure level, and is thus not equipped to process connected text or the semantic content of the input. As it makes no correlation between successive entries, STASEL does not have the capability of analyzing the coherency and organization of paragraphs. Though some would argue that proficiency in the techniques of inner sentence manipulation is not sufficient to ensure that a student writes well, sentence-level treatment follows from a recognized "divide and conquer" teaching approach: a student should understand and master the basic elements that constitute a written text, notably the sentence, before tackling the more challenging task of putting these elements together.[5] With its explicit and well-defined structure, widely documented in writing textbooks and readily decomposable for computer applications, the sentence offers an ideal medium for computer-assisted language instruction. However, STASEL is intended to allow text and semantic processing capabilities to be incorporated as additional functions, as suitable techniques are developed.

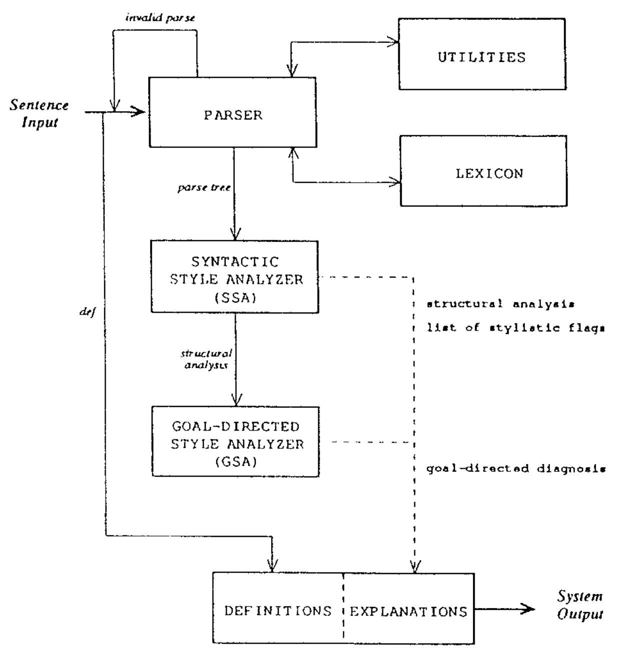

Figure 1. STASEL's system architecture

3.1. System Design and Operation

STASEL consists of 4,400 lines of C-PROLOG code[6] running under UNIX, more than 500 grammatical and stylistic rules, 100 definitions and descriptions, and an 800-word lexicon used for demonstration purposes (see Payette [1990] for more details). The higher-level syntax used to encode the grammar rules is known as definite clause grammar notation (Pereira and Shieber, 1987). The design structure of the system consists of six functional modules, as shown in Figure 1:

1. A syntactic parser that recognizes stylistic as well as syntactic features in the input sentence;
2. A prototype lexicon;
3. Prolog utilities for string manipulation and I/O functions;
4. The syntactic style analyzer (SSA), which detects stylistic problems in diction, usage, and sentence structure (discussed in detail in section 4);
5. The goal-directed style analyzer (GSA), which determines the structural clarity of the input (discussed in detail in section 5);
6. An information module that contains the definitions and explanation of strings that are used to generate the system's final output message.

Once typed, each input sentence is read and parsed, but only those free of grammatical and spelling errors are allowed to proceed to the stylistic analyzer modules. If a grammatical error is found, the system suspends all processing and informs the user of the presence of a grammatical mistake. As no grammatical corrector module exists in the present implementation, no attempt is made at defining the source of the error, nor at correcting the problem.[7] However, STASEL is quite rigorous in its grammatical error detection, disallowing sentence fragments, subject/verb disagreement, prohibited verb transitivity, and incorrect case choice.

The parser, lexicon, and utility modules serve as a front end for the subsequent stylistic analysis. With the I/O routines of the utility module, the parser reads in the input sentence and stores the words in a list structure that is examined from left to right, with backtracking as necessary[8] and lexicon lookups, until a parse is produced, or until all the structural combinations known to the parser have been attempted. If the parser fails to find a valid analysis of the sentence, STASEL stops processing and prompts the student to enter a new input. Otherwise the parser produces a structured representation of the sentence, the parse tree, that serves as a basis for the stylistic analyses that follow.

The parser gives control to the stylistic processing part of the system only if a valid parse tree has been produced. Obviously, stylistic errors and problems are allowed. After the analyses of the SSA and GSA modules are completed, STASEL provides instructional feedback by means of a structured message that states the result of its findings for each sentence interaction. As shown in the examples, this feedback consists of a four-segment message that summarizes the analysis of each sentence. After the first segment, which simply restates the object of the analysis, the next three segments provide stylistic feedback to the results obtained in the SSA and GSA modules:

- a list of stylistic messages (segment 2);
- a diagnosis of clarity (segment 3);
- a description of the sentence structure (segment 4).

The system also performs spelling checks and offers a built-in definition help facility that provides linguistic and stylistic information upon request.

All of the pedagogical feedback available in STASEL is contained in the information module in the form of explanations and definitions that are displayed to the student in the output message. The module effectively "translates" the output of the two stylistic modules into an adequate instructional response. This allows for easy modification of the message's content and provides a convenient means by which to tailor the instruction to a particular teaching method.

3.2. The Parser

A central component of the design of STASEL is the parser module, which breaks the sentence into its lexical and syntactic constituents according to the rules of the English language. Parsers are of particular interest in many CALI applications because they allow students to interact with the system using free-form input. In the case of STASEL, the parsing process not only provides a flexible interaction, but is also tailored for stylistic processing. STASEL's parser has several computational characteristics that enhance its operation and provide a comprehensive basis on which to build the stylistic components:

- it analyzes an important range of English sentence constructions, an essential feature if any form of stylistic analysis is to be worthwhile (see examples below);
- it decomposes the sentence into its grammatical and lexical components using grammar rules and integrated disambiguation heuristics to determine in the case of ambiguous constructions which reading is most plausible;
- its internal representation of the sentence, the stylistic parse tree, contains information on the stylistic impact of certain words (drawn from a lexicon that associates each word with its syntactic and stylistic attributes) and on the stylistic relevance of certain constructions (recognized during the parsing process). This information is made available as input to both stylistic modules so that detailed feedback on stylistic errors can be produced.

Parsing, in STASEL, involves three concurrent processes:[9]

- A lexical process that associates each lexical item with its function within the sentence, deciding which word is a noun, an adjective, a verb, etc.
- A syntactic process that groups a set of words according to its grammatical contribution to the sentence. This involves identifying each word group as a syntactic structure (noun phrase, relative clause, etc.) and each syntactic structure as a functional entity (subject, complement, modifier, etc.).
- A stylistic process that associates stylistic features to certain elements found during the lexical and syntactic processes. These features are used to identify the special characteristics of certain words (usage, vagueness) and certain phrases (inverted structure, passive construction, split infinitives) that are relevant to the stylistic analysis to follow.

The main function of the stylistic process of the parser is to construct a stylistic representation of the input sentence for use in a subsequent stage by the stylistic modules. This representation, called a stylistic parse tree, is built concurrently with the syntactic parse tree and thus follows the same parsing sequence. The structure of the stylistic tree, however, is very different from the corresponding syntactic structure. It is based on a frame statement notation derived from a semantic structure representation developed by Hirst for his Absity semantic interpreter (Hirst, 1987). Each grammatical entity defined during the syntactic process is associated with a frame statement headed by a frame determiner and contains two elements: a frame name and a list of attributes consisting of slot pairs or other frames. A frame statement is written in the following form:

Fdet(Fname, AttList)

where Fdet is the frame determiner, Fname is the frame name and AttList is the list of attributes. For example, the frame statement for the noun phrase *the cop* is

style(np, [slang = cop])

There are two possible types of frame determiners:

struct if the grammatical entity is a sentence;
style for all other entities that are not sentences.

Frame names identify a grammatical entity and cover the entire range of structures recognized by STASEL's parser. Examples of possible frame names are:

- "struct" frames:
  - simple for simple sentences;
  - compound for compound sentences;
  - inter for interrupted sentences;
- "style" frames:
  - np for noun phrases;
  - active for active verb phrases;
  - desc for adjectival phrases;
  - complex for relative and embedded clauses;
  - gerund for gerundival clauses.

The list of attributes describes the stylistic features of the grammatical entity to which it is associated. If no relevant attribute is detected, the list remains empty. An attribute can either be

1. a frame, as is the case in the attribute list of a simple sentence frame that holds noun phrase and verb phrase frames;
2. a slot pair that either identifies the stylistic feature of a particular lexical item (as in slang = cop) or describes any other pertinent stylistic characteristics found during the parsing process, such as the presence of complements or modifiers in the predicate, the detection of split infinitives, and the number of adverbs.

The grammatical entities that make up a sentence are thus expressed as "style" frames and are combined into a "struct" frame to represent the complete sentence. A "struct" frame can be embedded in another "struct" frame, as is the case in complex and compound sentences. The resulting frame structure representation becomes the stylistic parse tree of the sentence. The stylistic parse tree of *John bought a book* illustrates the relationship between the various elements of the frame structure and the sentence's components:

```
struct(simple, [
                style(proper, [ ]),
                style(active, [
                               comp=style(np, [ ])
                               adv=0])])
```

John, the subject, is represented by a "style" frame named proper and has no attributes, while *bought the book*, the predicate, is represented by another "style" frame named active and lists two attributes: a complement (noun phrase frame) and no adverbs. The entire sentence is contained in a "struct" frame named simple that has the subject and predicate "style" frames as attributes.

The advantages of using frame statements to represent the stylistic structure of the input sentence are numerous. Frames provide a readily accessible structure that not only preserves the hierarchical organization of the sentence but also presents the information concisely. Thus, only the information that is relevant for stylistic processing is included in the frame representation. In practice, this notation has greatly simplified the development of the stylistic modules, and significantly reduced the number and size of the rules needed to evaluate the stylistic content and structure of a sentence.

Examples of STASEL's parsing ability

One of the strengths of STASEL is its ability to parse an important range of sentence structures. The following represents only a subset of the possible variations allowed by the system:

- sentences with gerundival subject and noun modification;
- sentences with two complements;
- sentences with several levels of nested prepositional phrases;
- passive sentences or passive constructions;
- sentences with parallel three-way noun conjunction in the complement position;
- complex sentences with
  - several levels of embedded clauses;
  - subordinate clauses;
  - infinitive initial modifying elements;
  - adjectival final modifying elements;
- compound sentences;
- sentences with two coordinated verb phrases;
- sentences interrupted by
  - appositions;
  - adverbial clauses.

4. Stylistic Analysis

Clarity in sentence construction, precision in the choice of words, and conciseness in structure are all essential features contributing to the stylistic effectiveness of a sentence. These features also form the basis of the stylistic instruction provided by STASEL. This section shows how the stylistic processing capabilities of STASEL are implemented and describes the operation of the two stylistic analyzer modules: the syntactic style analyzer (SSA) and the goal-directed style analyzer (GSA).

4.1. The Syntactic Style Analyzer (SSA)

STASEL's SSA module provides conventional stylistic instruction at the word and sentence level. This module compares the input sentence against the principles and conventions of English syntactic style that are recognized, by stylists and English teachers, as contributing to stylistic effectiveness in sentence writing (Kane, 1983; Williams, 1981).[10] The module produces a summary of its findings that is later printed as feedback to the student. The instruction provided by the SSA module is geared toward the following objective: writing in formal settings, with precision and conciseness.

As input, the SSA module uses the stylistic parse tree built during the parsing process. From the tree, the module (1) produces a list of stylistic flags for each word choice or sentence construction that violates conciseness requirements and formal writing principles, and (2) generates a high-level representation of the sentence's structure called the structural analysis. The result of the SSA module analysis is summarized in the second and fourth segment of the output message of the system, as shown in the examples.

List of stylistic flags

In the SSA module, conventional "style-checking" is implemented with a flagging technique. Unlike many other style-checking systems, STASEL does not attempt to compute statistical parameters such as average sentence length or frequency of word occurrences, but instead, the system detects and analyzes a set of problem features that are not considered grammatical errors, but that lessen the stylistic impact of the sentence and interfere with the writer's ability to communicate with the reader. Examples of problem features that STASEL flags are:

- inappropriate usage choices (clichés, jargon, slang),
- wordiness contributors,
- misused phrases,
- informal constructs.

When processed by the information module, each flag is associated with a detailed explanation that contains a description of the cause of the problem and a remedial statement to help the student correct it. Each explanation is printed in the output message in conjunction with a heading that identifies the type of problem encountered. STASEL differentiates between five categories of stylistic problems:

Diction: problems that may refer to a lexical choice that is inappropriate in formal writing, to a word that is incorrectly used, or to an expression that has been devalued by overuse;
(flag heading: USAGE →)

Wordiness: lack of conciseness and the presence of wordy elements, redundant forms, unnecessary constructions, and meaningless structures;
(flag heading: [W])

Structure: syntactic constructions that may be potentially damaging to the style of the sentence, including excessive noun modification, split infinitives, faulty parallelism, and the use of the passive voice;
(flag heading: *)

Warning: special cases that do not fall under the above three categories, such as the use of intensifiers, double negations, vagueness, and unusual syntactic constructions;
(flag heading: WARNING:)

Appraisal: messages that appear whenever the SSA fails to detect stylistic problems in the input sentence, or if a particularly noteworthy sentence construction pattern, such as parallelism, has been found.
(flag heading: ⇔ GOOD)

Example 3 (stripped of segments 3 and 4 for readability) shows the output generated by the SSA module for a particularly confused sentence.

With its list of stylistic flags and associated replies, STASEL is thus able to inform the student of usage and wordiness problems, point out potential structural violations, issue warning and appraisal messages, and provide remedial feedback.

```
------------------------------------------------------------------

ANALYSIS OF:  In my personal opinion, it should be known that the
              problem of computers is not well understood.

------------------------------------------------------------------

  *  2 passives detected

 [W] wordy construct around "opinion"
     In good writing, adjectives and adverbs should link as
     directly as possible to what they really modify.
     A better link is to replace the prepositional phrase with
     an adverb derived from the adjective modifying "opinion".

 [W] "personal" redundantly modifies "opinion"
     It is a pointless repetition of the same idea

 [W] anticipatory construct introduced by "it"

 [W] the "problem of" is a weak construct which says
     nothing. It is better avoided

WARNING: use of attitude verbs such as "know" is unnecessary

WARNING: use negative ("not") only if necessary for emphasis or
         contrast. Otherwise, use a positive form

------------------------------------------------------------------
```

Example 3

Structural analysis
It is with the SSA's structural analysis that STASEL starts going beyond conventional style-checking programs. The information contained in the analysis is fundamental to the design of the system as it provides the basis for the implementation of goal-directed stylistic analyses that follow in the GSA module. The analysis serves two purposes: it is used internally as input to the GSA module for the goal-directed analysis, and it is displayed in the final output message, along with the result of the SSA and GSA analyses, to help the student understand the structure of the sentence and better appreciate the content of the stylistic messages.

The structural analysis not only gives a high-level description of the sentence's constituents, but also judges the complexity of the structures encountered. In the analysis, elements of a sentence are described as functional and grammatical entities that are either termed simple or complex according to the structure they display. Each constituent of the sentence is thus:

- described according to its function within the sentence (subject, complement, modifier, interrupting element, modifying element) that, in turn, is expressed as one or more grammatical entities (noun phrase, prepositional phrase, adverb, relative clause, etc.);
- identified as being either structurally simple or complex.

For example, consider one of STASEL's structural complexity principles, which states that prepositional phrases that consist of a conjunction of noun phrases are complex since they may lead to confusion, and therefore, contribute to making the sentence stylistically unclear. To illustrate this process, consider a sentence that displays such structural complexity:

John bought books about computers and music.

This sentence is ambiguous and structurally complex because it is not clear whether John bought books on computers and books on music, or whether he bought books on computers and then bought some items of music (scores, tapes, or records). In other words, without contextual information, it is not possible to determine whether *computers and music* or simply *computers* is the object of the preposition *about.* If the complex element, in this case the prepositional phrase, is placed at the end of the sentence, the ambiguity no longer remains:

John bought music and books about computers.

The stylistic parse tree of the ambiguous sentence is shown in Figure 2. Specifically, it consists of

1. two top-level "style" frames: the subject frame style(proper, [ ]) and the verb frame style(active, AttList), where AttList is the list of attributes of the frame;
2. one second-level style frame that describes the two-way conjunction of noun phrases (conjNP2) that form the complement;
3. two np style frames on the third level that describe the structure of the complement in detail;
4. one pp style frame on the fourth level that represents the prepositional phrase introduced by *about*;
5. a np frame on the fifth level that describes the object of the prepositional phrase.

The SSA module analyzes each style frame by applying to each of the attributes of the frame a series of rules designed to detect stylistic prob-

```
SUBJECT:

   style(proper,[])

VERB PHRASE:

   style(active,[
            comp=style(conjNP2,[
                          style(np,[
                                  style(pp,[
                                           pp=prep(about)
                                           style(np,[])])])
                          style(np,[study=music])])])
```

Figure 2. Stylistic parse tree of Example 4

lems. Whenever a rule succeeds in finding a problem, a flag is inserted into the output list of stylistic flags. The SSA module first examines the top-level frames and then proceeds in a recursive fashion until all levels have been examined. Thus, the first style frame that produces an output, consisting of an analysis and a list of flags, is the fifth-level np frame. This output is then passed back to the fourth-level pp frame, the fifth-level list of flags is appended to that of the fourth-level, and a new analysis is generated. The recursive process continues until each frame has been processed. Conversion heuristics are then used to translate the content of a sub-level analysis into a higher-level analysis. The complete recursion necessary to process the sentence *John bought books about computers and music* goes as follows:

- 5TH LEVEL:
  The frame structure style(np, [ ]) is converted to simple noun phrase.
  System mnemonic: simpleNP(none)
  Partial flag list: [ ]
  Corresponding element: *computers*
- 4TH LEVEL:
  The pp frame structure receives the output of the fifth level and converts the simple noun phrase analysis to a simple prepositional phrase analysis.
  System mnemonic: pp
  Partial flag list: [ ]
  Corresponding element: *about computers*
- 3RD LEVEL:
  - The first np frame of the conjNP2 structure receives the fourth-level output and converts it to a simple noun phrase containing a simple prepositional phrase.
    System mnemonic: simpleNP(pp)
    Partial flag list: [ ]
    Corresponding element: *books about computers*
  - The second np frame of the conjNP2 is converted to a simple noun phrase.
    System mnemonic: simpleNP(none)
    Partial flag list: [ ]
    Corresponding element: *music*
- 2ND LEVEL:
  The rule that checks for ambiguities and lack of parallelism in the organization of a compound structure detects a complexity in the information received from the third level, and the conjNP2 frame is converted to a complex two-way conjunction of noun phrases. A faulty parallelism flag (nonP) is added to the list of stylistic flags.
  System mnemonic: complexNP(np2)
  Partial flag list: [nonP]
  Corresponding element: *books about computers and music*
- TOP LEVEL:
  At this level, each functional entity is analyzed:[11]
  - The subject frame is analyzed and the proper-frame is converted to a simple noun phrase.[12]
    System mnemonic: subj(simpleNP(none))
    Partial flag list: [ ]
    Corresponding element: *John*
  - The verb phrase's active-frame receives complement information from the previous levels and converts the frame to a complex complement with no verb modification. The partial list of flags also carries on from the previous levels.
    System mnemonic: comp(complexNP(np2)) for the complement
    System mnemonic: noMod for the verb modification
    Partial flag list: [nonP]
    Corresponding element: *bought books about computers and music*

- SENTENCE LEVEL:
  This level completes the recursive process, produces a final list of flags, and builds the structural analysis from the output produced at each preceding level.
  STRUCTURAL ANALYSIS:
  clause(subj(simpleNP(none)),
  comp(complexNP(np2)), noMod)
  List of stylistic flags: [nonP]

The precise description of the higher-level constituents of the sentence contained in the structural analysis, along with the complexity analysis of its structure, is then passed on to the GSA module to serve as input for the coding of the goal-directed stylistic rules. Because it contains various mnemonics and abbreviations that would make little sense to a student, the structural analysis output used internally by the SSA is not printed directly in the output message. The analysis is "translated" with the help of the information module into an intelligible English reply. The final list of flags, the structural analysis, and the clarity diagnosis are then displayed to the student as shown in the output of Example 4.

```
-----------------------------------------------------------------
ANALYSIS OF:  John bought books about computers and music.
-----------------------------------------------------------------
  *  faulty parallelism
-----------------------------------------------------------------
 *** sentence is unclear ***

  >> complex structure in the predicate
-----------------------------------------------------------------
 <=> STRUCTURE: simple sentence

    clause -->
        subj: simple noun phrase (np)
        comp: complex >> conjunction of 2 noun phrases
-----------------------------------------------------------------
```

Example 4

4.2. The Goal-Directed Style Analyzer (GSA)

It is with the goal-directed style analyzer that STASEL exhibits its greatest novelty. The GSA module enables STASEL to judge sentences according to precise stylistic goals. The present implementation concentrates on the goal of structural clarity[13] and conciseness, but the system has been designed so that the inherent modularity of the goal-directed process will allow the analysis of other goals such as emphasis, concreteness, and formality to be easily implemented as future refinements of the system.

The clarity rules contained in the GSA module are based on formal principles of goal-directed stylistics that correlate patterns of syntactic structures and sentence constructions with specific writing goals. These rules are based on DiMarco's (1990) analysis of the syntactic correlates of stylistic goals and they enable the system to perform a clarity analysis that not only determines whether a given sentence is structurally clear, but also instructs the student in the reasons that led to the diagnosis.

The result of the GSA module's clarity analysis appears in the third segment of STASEL's output message, along with the high-level structural representation of the sentence's constituent (fourth segment). If a sentence is judged to be structurally clear, STASEL informs the student of the diagnosis and states the name[14] of the stylistic rule that was used in the analysis. If the sentence is somewhat clear, but contains a stylistic problem that, if corrected, would contribute to a clearer structure, STASEL responds with "NEUTRAL ANALYSIS" that explicitly states the stylistic problem encountered and provides remedial feedback. Otherwise, a message "*** sentence is unclear ***" is printed, followed by a list of reasons why the sentence lacks clarity. Since the structural analysis of the sentence is displayed below the clarity diagnosis, the student can correlate the reasons given by the system for the lack of clarity with precise sentence elements.

The notions of concord and discord

The stylistic principles used to judge the structural clarity of sentences in STASEL are derived from the theoretical model of goal-directed stylistics developed by DiMarco (1990) for the purpose of machine translation. This model is based upon the notions of concord and discord, for it is DiMarco's contention that "style is created by patterns of concord and discord giving an overall integrated arrangement" (p. 40). In STASEL, these notions have been expressed in highly practical terms by correlating them to explicit patterns and arrangements of syntactic elements within the sentence. (A syntactic element is any group of words within a sentence that can be defined as a grammatical or functional entity in the structural analysis of the SSA module. Noun phrases, prepositional phrases, infinitive clauses, main clauses, and modifying elements are all forms of syntactic elements.)

concord A syntactic element is concordant if it displays clarity and stability on its own, without ambiguity of attachment or excess structure (that is, elements that are identified as simple in the structural analysis of the SSA module). In practice, this implies that in a sentence, any grammatical entity that is not complex in structure and that clearly attaches to the rest of the sentence is concordant. For example, a simple noun phrase in the subject position is concordant. Other examples of entities that attach without ambiguity are the functional entities of a main clause (subject, complements, verb modifiers) and dependent clauses with a subject, such as conditional clauses and appositions. By extension, any functional entity of a sentence is concordant if all the grammatical entities with which it is associated are concordant. At the top level, a sentence is concordant if all its functional entities are concordant.

discord A syntactic element is discordant if it produces conflict, incongruity, or ambiguity because it is structurally complex and/or because its position in the sentence prevents a non-ambiguous attachment. In practice, any complex grammatical entity contributes to discord. Modifying elements that consist of a clause without a subject, a phrase displaying little structure (such as a simple adjectival or adverbial phrase), or an unusual construction (such as an inverted conditional phrase) are also discordant. For example, a prepositional phrase in the initial modifying position (that is, introducing the main clause) is discordant because it lacks an explicit subject and it may not be structurally clear which element it modifies, particularly if the following main clause contains many elements.

Discordant syntactic structures are by no means to be thought of as bad or wrong; they are commonly found in written texts, often bringing variety and pleasant relief from monotonous writing. If used carelessly, however, they may prove difficult to read and contribute to poor style.

Clarity rules

With the notions of concord and discord, and the complexity analysis provided by the SSA module (where grammatical and functional entities are termed simple or complex), it was possible to define a set of general rules, that correlate patterns of syntactic structures and sentence construction to the goal of clarity. STASEL defines eight such high-level rules[15] that correspond to the following general types of sentence constructions:

- **monoschematic** — simple sentence
  A monoschematic sentence consists of an independent main clause with no excess structure (i.e., a concordant main clause) and no dependent modifying elements.
  (1) monoschematic:-
      concordant main clause.
- **centroschematic** — complex sentence
  A centroschematic sentence consists of a concordant main clause accompanied by a concordant modifying element that either precedes, follows, or surrounds the main clause.
  (2) centroschematic:-
      concordant initial modifying element,
      monoschematic clause.
  (3) centroschematic:-
      monoschematic clause,
      concordant final modifying element.
  (4) centroschematic:-
      concordant initial modifying element,
      monoschematic clause,
      concordant final modifying element.
- **resolution** — complex sentence
  A sentence that produces a resolution is a special type of centroschematic sentence. Such sentences consist of a discordant initial modifying element followed by a highly concordant monoschematic clause that resolves the initial incongruity. The syntactic structures associated with a highly concordant monoschematic clause are more restricted than those of a simple monoschematic sentence, to ensure a clear resolution. Nonetheless, a resolution is not as clear as a centroschematic construct because of

the presence of a discordant element in the initial position (whereas the elements of a centroschematic sentence are all concordant).

(5) resolution:-
    discordant initial modifying element,
    highly concordant main clause.

- **positive interruption** — interrupted sentence
  A positive interruption consists of an interrupted monoschematic sentence in which the interrupting element is concordant.
  (6) positive-interruption:-
      concordant interrupting element,
      monoschematic clause.
  (7) positive-interruption:-
      discordant interrupting element,
      highly concordant main clause.
- **compound structure** — compound sentence
  A compound structure is a sentence that consists of the coordination of two balanced monoschematic elements (elements that display equivalent grammatical forms), and no modifying elements.
  (8) compound-structure:-
      monoschematic first clause,
      monoschematic second clause,
      balanced first and second clause.

Following DiMarco's model, we also defined a set of syntactic constructions that prevent clarity (i.e., that induce "obscurity"). We associated a lack of clarity in sentences with structural complexity, that is, sentences that contain too many dependent clauses or exhibit too much imitation (such as excessive noun modification and multiple conjunction of noun phrases). Interrupted sentences in which the interrupting element is discordant are also regarded as being unclear. Finally, it was established that the combination of coordination and clause dependency produces a discordant effect that prevents structural clarity. These "obscurity" principles were used as a basis for the design of STASEL to distinguish between simple (concordant) and complex (discordant) syntactic structures.

The high-level clarity rules corresponding to the sentence type are further broken down into specific principles of clarity (acceptable features) and obscurity (unacceptable features) and are applied in turn to the structural analysis of the input sentence. If the rules all fail, the GSA enters a set of backup rules that determines the cause of the failure. This mechanism is used to generate the remedial and instructional feedback that is printed in the GSA output segment along with the clarity diagnosis. Each set of backup rules contains a default rule that applies when all other backups have also failed, ensuring that a diagnosis is always produced.

An example: the monoschematic clarity rule
As an example, consider the monoschematic clarity rule for a simple sentence (that is, a sentence that has no dependent elements). In its details, it reads as follows: a sentence consisting of one independent clause that, in turn, consists of a subject, complements, and verb modifiers, is clear if all of the following are true:

- the subject is a simple noun phrase;
- the subject is not a three-way conjunction of noun phrases;
- complements, if present, are simple in structure;
- verb modifiers, if present, are simple in structure;
- there is no excessive[16] noun modification;
- there is no excessive qualification;
- there is no excess number of adverbs;
- there is no excess number of passive verbs.

If we apply this rule to the sentence

The museum acquired paintings and canvases that were recycled.

the system will detect that there is a complex element in the complement position as it is unclear whether both the *paintings* and the *canvases* or only the *canvases* were recycled (which corresponds to an ambiguous attachment of the two-way conjunction of noun phrases). Hence, the monoschematic rule fails on the third condition and the set of backup rules for simple sentences is invoked to diagnose the cause of the failure. The resulting diagnosis is printed in the final output message (Example 5) along with the structural analysis.

5. Concluding Remarks

STASEL is of interest because it performs "practical" goal-directed analysis of the input according to formal principles of stylistic clarity. Moreover,

```
------------------------------------------------------------------

ANALYSIS OF:  The museum acquired paintings and canvases that were
              recycled.

------------------------------------------------------------------

 *** sentence is unclear ***

  >> complex structure in the predicate

------------------------------------------------------------------

 <=> STRUCTURE: simple sentence

    clause -->
        subj: simple noun phrase (np)
        comp: complex >> conjunction of 2 noun phrases

------------------------------------------------------------------
```

Example 5

STASEL is based on a comprehensive parser/stylistic analyzer combination that recognizes an important range of syntactic structures and stylistic features of sentences. The subtlety and exhaustiveness of the clarity analysis performed by the system show that sophisticated goal-directed processing is not the abstract and informal concept that many believe it to be, but a worthwhile application that can be effectively computerized for the purpose of language instruction.

This research demonstrates that both syntactic and goal-directed principles of style can be practically and successfully implemented for the purpose of language instruction. This is achieved in STASEL by

- implementing formal principles of goal-directed stylistics;
- using a parsing approach that recognizes stylistic as well as syntactic features in the input sentence;
- generating explicit representations of the sentence's elements in the form of descriptive analyses that are specifically suited for stylistic processing;
- encoding the rules and conventions of English syntactic style in the analyzer modules;
- providing a communicative environment where the student can "learn by doing."

The descriptive analyses generated at various stages in the system, notably the stylistic parse tree and the SSA's structural analysis are particularly significant. They offer a novel and flexible means of treating various kinds of stylistic considerations and facilitate the refinement and extension of existing features. These analyses not only provide the preliminary processing that is fundamental to the detection of a wide (and expandable) set of stylistic problems, but also carry the basic information contained in the input sentence in a form that is tailored to the needs of subsequent stages. This technique contributes to a more controlled flow of information through the system, and allows for increased modularity and ease of expansion in the design.

Although STASEL covers a wide range of stylistic problems, the system was not intended to embody all the characteristics of a fully functional stylistic tutor, but rather to provide a structured core around which a pedagogically sound, highly communicative intelligent system can be built. To be practical in a classroom, the system's current implementation would require a more extensive stylistic and goal-directed coverage, a full dictionary (lexicon), an improved parser functionality to recognize very complex structures, some semantic processing capabilities, and a more comprehensive user interface. Moreover, it should undergo thorough testing on a corpus or in a real-life classroom. Despite these limitations, the design of STASEL displays important novelty in that its highly modular approach introduces a variety of new techniques in syntactic and goal-directed stylistic processing. Providing a high degree of expandability was a central concern in the design of STASEL and was dictated by the long-term objective of this research: that the STASEL prototype will form the basis for the development of intelligent tutoring systems for teaching writing.

The reader is invited to refer to Chapter 6 in Payette (1990) to obtain a detailed account of how the principles and techniques developed in STASEL may be used to implement the components of the ITS architecture for teaching writing that are not covered by the present prototype.

Acknowledgements
The research outlined in this article was conducted at the University of Toronto with financial support from the Natural Sciences and Engineering Research Council of Canada. Its theoretical foundation owes much to the work performed by Chrysanne DiMarco at the same university. We

would also like to extend our sincere thanks to Michael Stumm, professor in the Department of Electrical Engineering, for his guidance and constant support. Additional resources and computer facilities that have contributed to the writing of this article were provided by the IBM Research Laboratory in Zurich, Switzerland, and by Bell-Northern Research in Montreal.

A more detailed description of this research is available as technical report CSRI-247, Computer Systems Research Institute, University of Toronto.

Notes

* Currently at the Canadian Astronaut Program Office, Canadian Space Agency, P.O. Box 7014, Station V, Vanier, Ontario K1L 8E2, Canada. Requests for offprints should be directed to the second author.

[1] In its broadest sense, the term "style" refers to the totality of all the choices a writer can make regarding words and their arrangements in order to convey his or her thoughts and ideas. In this work, we use the term to mean the distinctive, formal, and standard manner of expression characteristic of effective writing and we define *stylistic rudiments* to mean the norms and conventions that are accepted, in textbooks on style, as conforming to standards of formal writing. This definition is then extended to include the study of the arrangements and the constructions within a sentence that support the realization of specific writing goals.

[2] Very few CALI programs have so far attempted to tackle the difficult task of integrating stylistic principles in the analysis of sentences let alone attempted to teach these principles for effective writing. Chapter 2 in Payette (1990) reviews some of these programs, outlining the features that discourage their use in a teaching environment and demonstrating where and how STASEL may provide a better alternative.

[3] In speculating about intelligent systems, one must realize that current implementations incorporate experimental AI techniques, and thus are unlikely to be seen in routine educational applications for some years to come.

[4] For a general description of ITS research in language instruction, see Bailin and Levin (1989), Duchastel (1988), Farghaly (1989), Neuwirth (1989), Payette (1990).

[5] Such a "divide and conquer" teaching method is advocated, among others, by Williams:

> *To understand why anyone writes badly, we have to be able to look at a sentence and understand how it works, how the ideas have been distributed through its different parts, and then decide how to write it better.* (Williams, 1981, p. 12)

[6] PROLOG was chosen over other programming languages because of its strong affinity with natural language processing. Its descriptive nature favours the implementation of a knowledge base in which objects (words) and the relationship between objects (grammar or style rules) can be formally defined. More practically, PROLOG features built-in pattern-matching routines that are particularly useful when searching for strings of words within a sentence, a powerful list processing capability that allows a sentence to be stored as an ordered list of items (the words) that is easily manipulated and referenced, and a higher-level syntax that simplifies the coding of grammar rules. PROLOG also displays intrinsic modularity, which is suitable not only for building prototype systems quickly, but also for adding successive stages to an existing program.

[7] See Catt (1988) for an example of an intelligent grammar corrector specifically designed for language instruction.

[8] The performance inhibitors of the system lie almost entirely in the backtracking nature of the parsing process. As discussed thoroughly in section 3.3 of Payette (1990), STASEL's parser has been designed to circumvent some of the inherent problems associated with the use of a backtracking parsing mechanism, yet there are still significant fluctuations in response time according to the nature and structure of the input sentence. STASEL's parser would benefit from having additional disambiguation mechanisms (following Hirst [1987]) to reduce the parsing time of ambiguous structures, but this represents a research topic of its own and diverges from the main interest of this work.

[9] See Payette (1990), section 3.1.2, for more detailed information on STASEL's parsing processes.

[10] The set of principles and conventions implemented in the prototype system is by no means exhaustive or meant as an absolute norm. Anyone who disagrees with our choice of rules could easily substitute or add their own.

[11] Frame statements for proper names and pronouns both produce the same analysis as a simple noun phrase that has no post-modification (that is, has no attached prepositional phrase or modifying clause). The system mnemonic for all three cases is simpleNP(none), where none stands for no post-modification.

[12] If other entities besides the subject and the verb phrase are present, for instance, in a compound sentence structure or in a sentence that contains a modifying element, the top-level analysis is performed on each entity in turn.

[13] Clarity is understood to mean clarity in structure rather than clarity in meaning, as, in the absence of semantic capabilities, STASEL cannot comment on the meaningfulness of a sentence. The system looks solely at the surface structure and bases its analysis on the syntactic elements it recognizes. Consequently, a sentence such as

> *I swim in the office.*

will be judged by STASEL as being stylistically correct and structurally clear as indeed it is, even if it is nonsense.

[14] The names given to STASEL's stylistic rules were partly derived from DiMarco's vocabulary of abstract stylistic elements (DiMarco, 1990). Each corresponds to a general type of sentence construction. Refer to Payette (1990), Section 5.1.2, for more details.

[15] For conciseness, the clarity rules presented here only show a high-level description of the principles involved and are stated in pseudo-PROLOG form where the symbol ":-" means *if* and a comma in the body of the rule is read as *and.* The

details of each rule may be found in chapter 5 of Payette (1990).

[16] The terms *excessive qualification* or *excess number of adverbs* are not prescribed by the system and may be set by instructors according to their own teaching methods.

References

Bailin, Alan. "Artifical Intelligence and Computer-Assisted Language Instruction: A Perspective." *CALICO*, 5, 3(1988), 22—45.

Bailin, Alan and Lori Levin. "Introduction: Intelligent Computer-Assisted Language Instruction." *Computers and the Humanities*, 23, 1(1989), 3—11. Special Issue on Intelligent Computer-Assisted Language Instruction.

Burston, Jack. "Towards Better Tutorial CALL: A Matter of Intelligent Control." *CALICO*, 6, 4(1988), 75—89.

Catt, Mark Edward. "Intelligent Diagnosis of Ungrammaticality in Computer-Assisted Language Instruction." Master's thesis, Department of Computer Science, University of Toronto, October 1988. Published as technical report CSRI-218, Computer Systems Research Institute.

Chapelle, Carol. "Using Intelligent Computer-Assisted Language Learning." *Computers and the Humanities*, 23, 1(1989), 59—70. Special Issue on Intelligent Computer-Assisted Language Instruction.

DiMarco, Chrysanne. *Computational Stylistics for Natural Language Translation.* PhD thesis, Department of Computer Science, Univerity of Toronto, May 1990. Published as technical report CSRI-239, Computer Systems Research Institute.

Duchastel, P. C. "Models for AI in Education and Training." In *Artificial Intelligence Tools in Education.* Ed. P. Ercoli and R. Lewis. North-Holland, 1988, pp. 17—28. Proceedings of the IFIP TC3 Working Conference on Artificial Intelligence Tools in Education, May 1987.

Farghaly, Ali. "A Model for Intelligent Computer Assisted Language Instruction (MICALI)." *Computers and the Humanities*, 23, 3(1989), 235—50.

Ferney, Derrik. "Small Programs that 'Know' What They Teach." In *Computer Assisted Language Learning.* Ed. Keith Cameron. Ablex Publishing, 1989, pp. 14—27.

Hamburger, Henry. "Evaluation of L2 Systems: Learners and Theory." *Computer Assisted Language Learning*, 1, 1(1990), 19—27.

Hirst, Graeme. *Semantic Interpretation and the Resolution of Ambiguity.* Cambridge University Press, 1987.

Kane, Thomas S. *The Oxford Guide to Writing: A Rhetoric and Handbook for College Students.* Oxford University Press, 1983.

Last, R. W. *Artificial Intelligence Techniques in Language Learning.* Ellis Horwood Series in Computers and their Applications. Halsted Press, 1989.

Macdonald, N. H., G. P. Frase, and S. A. Keenan. "The WRITER'S WORKBENCH: Computer Aids for Text Analysis." *IEEE Transactions on Communication*, 30 1(1982), 105—109.

Mulford, George W. "Semantic Processing for Communicative Exercises in Foreign Language Learning." *Computers and the Humanities*, 23, 1(1989), 31—44. Special Issue on Intelligent Computer-Assisted Language Instruction.

Neuwirth, Christine M. "Intelligent Tutoring Systems: Exploring Issues in Learning and Teaching Writing." *Computers and the Humanities*, 23, 1(1989), 45—57. Special Issue on Intelligent Computer-Assisted Language Instruction.

Payette, Julie. "Intelligent Computer-Assisted Language Instruction in Syntactic Style." Master's thesis, Department of Electrical Engineering, University of Toronto, October 1990. Published as technical report CSRI-247, Computer Systems Research Institute.

Pereira, Fernando C. N. and Stuart M. Shieber. *Prolog and Natural Language Analysis.* Center for the Study of Language and Information, Stanford University, 1987. CSLI lecture notes 10.

Richardson, Stephen D. and Lisa C. Braden-Harder. "The Experience of Developing a Large-Scale Natural Language Text Processing System: CRITIQUE." In *Proceedings of the Second Conference on Applied Natural Language Processing*, Austin, TX, 1988, pp. 195—202.

Underwood, John. "On the Edge: Intelligent CALL in the 1990's." *Computers and the Humanities*, 23, 1(1989), 71—84. Special Issue on Intelligent Computer-Assisted Language Instruction.

Williams, Joseph M. *Style: Ten Lessons in Clarity and Grace.* Foresman and Company, 1981.