Behavior Research Methods, Instruments, & Computers
1992, 24 (4), 535-540

Computer dialogue system (CDS): A system for capturing computer-mediated dialogue

JUN LI, JAI H. SEU, and MARTHA EVENS
Illinois Institute of Technology, Chicago, Illinois

and

JOEL A. MICHAEL and ALLEN A. ROVICK
Rush Medical College, Chicago, Illinois

In the process of studying human tutoring sessions as a basis for building an intelligent tutoring system, we developed a computer dialogue system (CDS) that allows two PC users to communicate with each other over a telephone line by typing at a computer keyboard. CDS records the content of dialogue on a disk in a specified, well-formatted way. It also makes available a way to mimic some of the characteristics of face-to-face dialogue such as repair. It was developed in the Smartcom III (Hayes Communication package) environment. Thus, it is fast, portable, and easy to use. In addition, to help us study the recorded dialogues, we have also written a numbering program to label each turn and each sentence within each turn. Although CDS was originally designed for the study of tutorial dialogues between students and teachers, it can be used to conduct and record dialogues of any kind.

The purpose of the paper is to describe a computer-based dialogue system that we developed to study tutoring in a setting in which the tutor and student cannot rely on visual or intonational cues. This setting closely resembles the dialogues that might occur between a student and a computer tutor, our area of interest. We also illustrate how we use the output of the system.

Computer Dialogue System (CDS)

We are in the process of developing an intelligent tutoring system (ITS; see Wenger, 1987) that will help first-year medical students understand the behavior of the baroreceptor reflex (the portion of the cardiovascular system that maintains our blood pressure; Kim, Evens, Michael, & Rovick, 1989). To build an efficient natural language interface for this application and to incorporate effective tutoring rules, it seemed appropriate to study actual human tutoring sessions. Our initial studies involved audiotaping the tutorial dialogue generated by a tutor (J.A.M. or A.A.R.) and a student. Transcripts in machine-readable form were then processed to provide information about the characteristics of the language used in this specific domain (i.e., the cardiovascular sublanguage; Seu, Evens, Michael, & Rovick, 1991). Debriefing sessions with the transcripts yielded preliminary tutoring rules (Woo, Evens, Michael, & Rovick, 1991).

However, studies have shown that the bandwidth of the communications channel available for a dialogue greatly affects the characteristics of the sublanguage (Thompson, 1980). The dialogue between a person and a computer has a relatively narrow bandwidth, since the only way information is currently conveyed from the student to the tutor (computer) is through text entered at the keyboard. On the other hand, in addition to the *words* used in a dialogue between people, information conveyed by intonation, facial expressions, and gestures is used. It has been shown (Fox, 1990; Grosz, 1978; Thompson, 1980) that the same individual, working in the same domain, will use a different sublanguage when interacting with a computer than when talking to another person. To obtain the most useful data for our sublanguage analysis, we felt that an environment that allows users to interact by using a keyboard would provide the best model for the man-machine interaction that we are attempting to construct.

The computer dialogue system (CDS) has been developed to allow two PC users to communicate with each other, over a telephone line or a direct cable connection, by typing at the computer keyboards. During the course of communication, the two users take turns typing. Each character typed by one user is displayed on both screens almost simultaneously. To improve readability, CDS uses windows to separate one user's input from the other. It also provides the users with the ability to mimic some verbal characteristics of natural dialogues, such as interruptions of one participant by the other, without losing the message. Furthermore, it has a timing facility that dis-

This work was supported by the Cognitive Science Program, Office of Naval Research under Grant N00014-89-J-1952 (Grant Authority Identification Number NR4422554) to the Illinois Institute of Technology. The content does not reflect the position or policy of the government, and no official endorsement should be inferred. The first three authors are in the Computer Science Department at the Illinois Institute of Technology. Address correspondence concerning this work, or requests for the software described here, to Martha Evens, Computer Science Department, Illinois Institute of Technology, Chicago, IL 60616 (e-mail: csevens@minna.iit.edu).

plays elapsed times and latency times on the screen and records them in the final saved record. Finally, the dialogue and information on the date and participants are well formatted and saved on the hard disk of one of the computers.

CDS is easy to use. Besides ordinary printable character keystrokes, it uses only a limited number of control keys (keys used to request an interruption, or a change in turn, etc.). A help window that describes the function of all important keys is visible on the screen all the time. CDS utilizes sound and the background color of the window to help the user identify the appropriate action to take next. When communication begins, a short summary screen that explains how to use the system is displayed.

The CDS program is portable and reliable (as long as no memory-resident program interferes[1]). CDS runs on two IBM (or compatible) XT (or above) computers equipped with hard drives, Hayes compatible modems (baud rate $\geq$ 1,200) installed on COM1, color (EGA or VGA) monitors, and Hayes Smartcom III (SCOM3) installed on both hard disks. The program can be easily adapted, however, to use monochrome monitors or to use a different port.

For a review of other tools available for interface development, we refer the interested reader to the detailed and comprehensive survey by Hartson and Hix (1989). More recently, Hix and Schulman (1991) have described a methodology for tool evaluation.

The CDS Development Environment

Telecommunication programs are very device-dependent and can take a long time to develop. To simplify our task, we decided to use SCOPE (the Simple Communication Program Environment), a Hayes command language (embedded in the Smartcom III communication package) developed especially for automating data communication tasks (Hayes Microcomputer Product, Inc., 1989). This programming language allows for control of the communication functions, thus making it more suitable than a general purpose language (e.g., C) for this application, even though it is less flexible for data processing. One example of its rigidity is the way it handles all nonprintable keystrokes such as <Backspace>—SCOPE completely ignores them. Therefore, CDS must mimic the effect of the <Backspace> key by erasing what is on the current window and redisplaying the original input string without the erased character. Nevertheless, SCOPE does relieve the programmer of the need to deal with the many tedious details that must be managed in order to communicate between two computers. The CDS system must, therefore, run in the SCOM3 environment. SCOM3 allows a communication session to be initiated from the DOS prompt, which means that the users of CDS do not need to know anything about SCOM3 itself.

In SCOM3, a communication session is controlled by two setting categories (Hayes Microcomputer Product, Inc., 1989): activity and connection. The activity setting affects communications tasks such as the screen display or the file transfer method. It may include a piece of program (written in SCOPE) that defines what is to occur during each session. The connection setting affects how the communication is made—for example, what telephone number to dial and at what speed to transmit information.

Since CDS communicates between two PCs, there are two sets of these settings—one set for each PC. The settings are identical except for the scripts and the initial transmission mode (either receiving mode or dialing mode). These settings can be easily imported and exported; this makes installation or modification an easy task.

In our context, one of the PC users is the "teacher," and the other is a "student." The dialogue record is saved on the hard disk of the teacher PC. The script used to control the teacher's PC is called "ctttside"; script for the student's PC is called "ctstside." These two scripts are essentially mirrors of each other. That is, when "ctttside" is processing incoming keyboard data, "ctstside" must be processing data from the receiving buffer, and vice versa. During the whole period of communication, the scripts are in a loop that alternates between processing keyboard input and processing data received from the other PC. The synchronization between these processes is achieved by message passing.

The CDS Prototype

In the initial version of CDS, there was no program control over the process of conducting a dialogue. After invoking CDS on both PCs, the two computers were connected, and whatever was typed on one of the keyboards appeared on both screens simultaneously. However, if both users typed at the same time (not a rare occurrence), the resulting display was a mixture of the two messages (which was usually meaningless). For example, if one user typed "mix" and the other typed "together" at the same time, the screen display might be something like "mtogie txher" instead of "mix together."

To ensure more meaningful interaction, a simple protocol for each user was developed to force the users to take turns typing:

1. Each user was asked to type a specified string to signal the end of a turn (xxx was chosen for one user, zzz for the other).

2. If one user wants to interrupt the other (who has the turn), another specified string (yyy) is typed. When the other user sees this string, he/she must give up the turn.

This protocol can be made to work well. The first eight tutoring sessions that we studied were obtained in this way. We were able to achieve 60-90 turns in a 1-h tutoring session (Seu, Chang, et al., 1991).

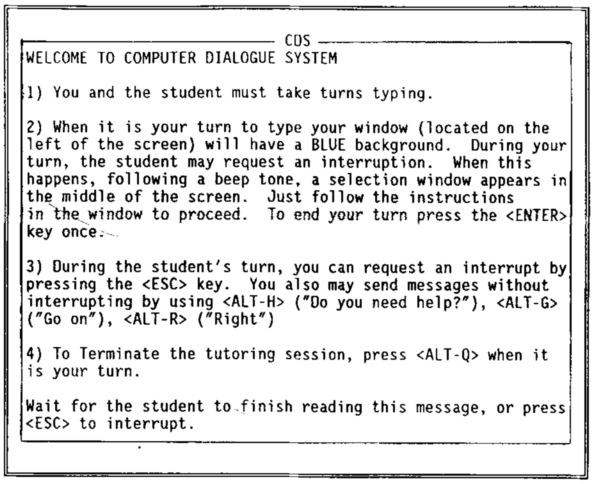
However, this protocol has three major drawbacks:

1. The screen provides no assistance in determining whose turn it is at any given time, and since all input is continuously displayed on both screens, it is hard to locate one particular entry.
2. If one user wants to interrupt the other, the other has no choice but to give up his/her turn. This is not a feature of natural dialogues, and it frequently results in loss of one's train of thought.
3. The record of the dialogue was not well formatted. Even though we could tell who generated any given input by counting "end signals" and "interrupt signals," it was difficult and confusing.

We set out to solve these problems in a new version of CDS.

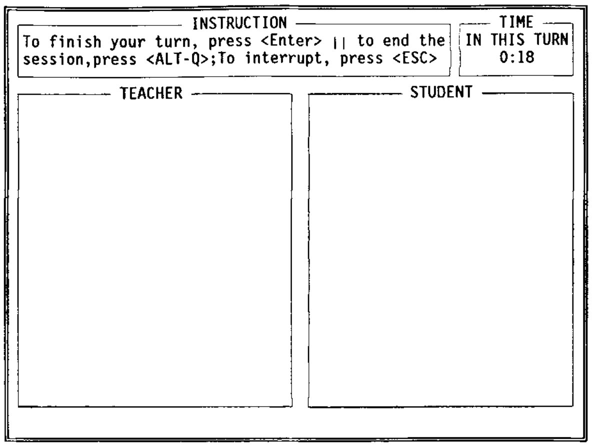
The Current Computer Dialogue System

The current program is able to handle labeling, formatting, and concurrent typing. On initiation, the first thing the program does is to set up a window (Figure 1) to provide operating instructions on the use of all important keys.



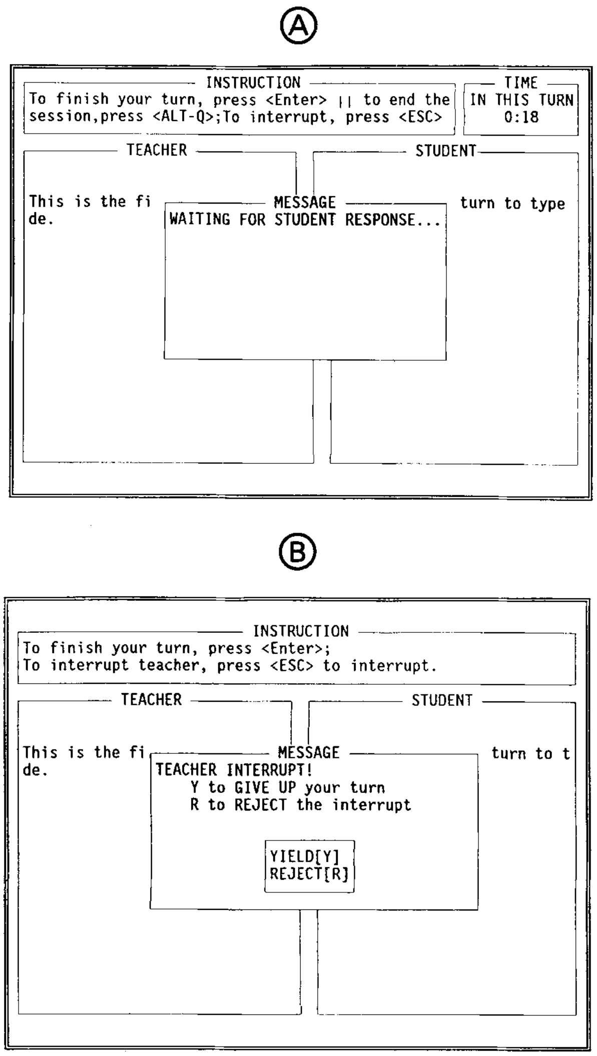
```
CDS
WELCOME TO COMPUTER DIALOGUE SYSTEM

1) You and the student must take turns typing.

2) When it is your turn to type your window (located on the
left of the screen) will have a BLUE background.  During your
turn, the student may request an interruption.  When this
happens, following a beep tone, a selection window appears in
the middle of the screen.  Just follow the instructions
in the window to proceed.  To end your turn press the <ENTER>
key once.

3) During the student's turn, you can request an interrupt by
pressing the <ESC> key.  You also may send messages without
interrupting by using <ALT-H> ("Do you need help?"), <ALT-G>
("Go on"), <ALT-R> ("Right")

4) To Terminate the tutoring session, press <ALT-Q> when it
is your turn.

Wait for the student to finish reading this message, or press
<ESC> to interrupt.
```

Figure 1. CDS introductory window.



```
INSTRUCTION                                          TIME
To finish your turn, press <Enter> || to end the  IN THIS TURN
session,press <ALT-Q>;To interrupt, press <ESC>       0:18

TEACHER                        STUDENT
```

Figure 2. CDS screen layout (teacher side).



```
Ⓐ
INSTRUCTION                                          TIME
To finish your turn, press <Enter> || to end the  IN THIS TURN
session,press <ALT-Q>;To interrupt, press <ESC>       0:18

TEACHER                        STUDENT
This is the fi      MESSAGE              turn to type
de.         WAITING FOR STUDENT RESPONSE...
```

```
Ⓑ
INSTRUCTION
To finish your turn, press <Enter>;
To interrupt teacher, press <ESC> to interrupt.

TEACHER                        STUDENT
This is the fi      MESSAGE              turn to t
de.         TEACHER INTERRUPT!
              Y to GIVE UP your turn
              R to REJECT the interrupt

                    YIELD[Y]
                    REJECT[R]
```

Figure 3. (A) Wait for response to the interrupt request; (B) choices to reject or yield to an interrupt request.

To improve the readability of the dialogue on the screen, two windows are used (Figure 2) to separate one user's entries from those of the other user. To maintain the continuity of the dialogue, the user's last entry is also displayed during the current entry. When color monitors are available, users can determine whether to type or to request an interrupt by changes in the background color of the user's window from blue to black or vice versa.

The interrupt facility has also been improved. If User A wants to interrupt User B, he/she can do so by simply pressing the <ESC> key and waiting for User B's response (Figure 3A). User B can then yield to or reject this interrupt request (Figure 3B). If User B decides to yield, he/she gives up the turn, and User A can type; otherwise, User B continues.

The stored transcript is formatted with each turn labeled with either TU (the teacher uses the "tutor" side of the software) or ST (the student uses the "student" side of the software). Interrupts are labeled with TI when the tutor interrupts the student and SI when the student interrupts the tutor. At the beginning of the dialogue session, both participants are asked for their last names; that information, as well as the date, is also recorded in the transcript.

Since CDS was designed for conducting tutoring sessions, several other features have been added to this system. CDS records (in the hard disk file) the time that it takes for the student to type the first character in his/her turn (the response latency) and the total time taken in that turn. The time is also displayed in a "clock" on the teacher's screen (see Figure 2). Both expert tutors (J.A.M. and A.A.R.) found it important to know the duration of the delay in the student response before initiating an interrupt to see if the student needed help. A third feature that was added is a noninterrupt message-sending capability: The teacher can send messages to the student without having to actually interrupt the student's turn. The following messages are currently predefined in the program:

- Do you need help?
- Right
- Go on

These were added because they provide instant feedback to the student (which Fox, 1990, argues is very important to keep the student working well) without artificially increasing the time required for a turn.

A new feature is under development that will allow the tutor to specify a text file for the student to read during the CDS session. This would be very useful in our experiment because it would allow the students to be given either new problems or some formal explanation regarding certain situations.

Reformatting and Numbering the Computer Records Output by CDS

Before we could study the language and content of the captured dialogue in detail, it was essential to first tag each sentence or fragment with an identification number. A good numbering scheme must tell us which part of the dialogue is involved (i.e., who typed this sentence at what time in the session) and must also take as little space as possible since we are handling a large amount of data. The current numbering scheme has the following format:

session# - who - turn# - sentence#

For example, the number K24-ST-057-01 indicates that the sentence comes from Keyboard Session 24, that the student is typing, that this is Turn 57, and that it is the first sentence in that turn. The second component of this number may be TU (tutor), TI (tutor interrupting the student), ST (student), or SI (student interrupting the tutor). The turn number shows how many times the turn has changed during the current session. The sentence number indicates the order of the sentences or sentence fragments during the current turn. Figure 4 shows sample output that is produced by the automated numbering program.

```
4/10/90 Teacher: Rovick  Student: Rejman

CT: Time when turn starts (in form of minutes:seconds).
ET: Elapsed time in following turn.
IT: Initial response time (between turn change and first
    keystroke.
TU: Teacher
ST: Student

CT=0:5  ET=0:0  IT=0:0
ST:-> REJMAN
......
CT=11:23 ET=0:18 IT=0:12
ST:-> HR I
TU:-> Good.  What is affected next?

CT=11:56 ET=0:9 IT=0:5
ST:-> co i
CT:-> Very good.  Then what?

CT=12:41 ET=0:18 IT=0:13
ST:-> map i
CT:-> Super.  Where do you want to go next?

CT=14:33 ET=0:25 IT=0:11
ST:-> tpr i
TU:-> Remember, we're dealing with the DR period.  That's
      before there are any reflex responses.  TPR is a
      neurally controlled variable.  Try again!

CT=17:2 ET=0:16 IT=0:8
ST:-> TPR o
TU:-> Right.  What others neurally controlled variable is
      there and how is it affected?

CT=17:53 ET=0:9 IT=0:2
ST:-> cc o
TU:-> Great.  We're on a roll.  Lets come back to an
      earlier prediction of yours.  You said (correctly)
      CO I.  What would that affect?

......
```

Figure 4. Sample output of a dialogue conducted through CDS.

The numbering program is written in C. It begins by detecting changes in turn and identifying the speaker/writer as a tutor or a student. Then, within the turn, the program determines sentence endings. This is a weak point—abbreviations terminated by a period are misidentified as sentence endings. Consequently, obtaining correct output requires a manual pass for postediting. It would be possible to improve sentence identification by adding a list of standard abbreviations to the system.

Using CDS in the Design of an Intelligent Tutoring System

CDS has been used to record 28 keyboard-to-keyboard tutoring sessions as a first step in the design and development of an intelligent tutoring system. We are carrying out an extensive analysis of both the language and content of the transcripts.

Both CDS and the numbering program are used to study the sublanguage used by the tutor and the student (Grishman & Kittredge, 1986). The first step is to extract the vocabulary that they used. A word list construction program makes sorted lists of words and phrases. These lists tell us how many words were used, how many were used in each session, and what kinds of words were used. Next we divide the lists into separate lists of words used by tutors and students. We want to find out when and how the student's use of words follows the tutor's use; the numbering system is particularly useful here. We eventually will be able to trace the process by which students learn the proper physiology terminology from their tutors. We are also studying the use of phatics and connectors in these sessions. The list of words and phrases is also being used as the lexicon for the language understander in the tutoring system.

The language analysis includes the study of the vocabulary, syntax, and discourse for use in the understanding of ill-formed input (Lee, Evens, Michael, & Rovick, 1990; Seu, Evens, et al., 1991) and the generation of system responses (Zhang, Evens, Michael, & Rovik, 1990). The analysis of content involves the attempt to infer tutoring rules and student modeling rules from the transcripts (Woo et al., 1991). Most recently, we have carried a comparison of the language used in both face-to-face and keyboard-to-keyboard tutoring sessions (Seu, Chang, et al., 1991). Before the development of CDS, the input-understanding and text-generation modules used samples of written examinations and transcripts of face-to-face tutoring sessions to guide their research. When we obtained the first eight keyboard-to-keyboard sessions and began to analyze them, we discovered some striking new phenomena:

1. We had expected to find terse responses, but we were surprised by the number of student responses that consisted of a single adverb such as "up" or "down."

2. Terseness led to some peculiar abbreviations and novel syntax, for example, "cc d" for "cardiac contractility decreased."

3. Just as "d" stands for "decrease(d)," upper or lower case "i" stands for "increase(d)." This use of "I" is much more frequent than the use of the first person pronoun.

4. Novel abbreviations are common. Capitalization seems random.

5. Enthusiastic one-word confirmations of the correct answers were much more frequent than they were in the face-to-face sessions, for example, "Absolutely," "correct," "excellent," "great," and "super." These came from deliberate attempts by the tutors to convey signals that they express nonverbally in face-to-face sessions.

6. The students ask fewer questions than they do in face-to-face sessions; when they do request help, they are much more likely to phrase it as a declarative sentence, such as: "I don't understand cardiac output" or "I am confused about cc." As more data became available, a much more comprehensive comparative study of face-to-face and keyboard-to-keyboard sessions has been made possible. The first results have been described in Seu, Chang, et al., (1991). Here, we summarize only the major findings.

7. As we expected, more words were exchanged in the face-to-face sessions than in the keyboard-to-keyboard sessions. The average number of words per turn dropped from an average of 13.1 in the face-to-face sessions to 10.1 in the keyboard-to-keyboard sessions. The students were more affected by the change in modality than were the tutors. The student contributes only 25.3% of the words in a keyboard dialogue as opposed to 37.2% in face-to-face sessions.

8. The average sentence length dropped from 6.5 to 5 words, but the drop in sentence complexity was even more marked. In the keyboard-to-keyboard sessions, the frequency of relative clauses and the frequency of sentential complements were both cut in half.

9. We expected that the range of vocabulary items would also drop. We were wrong. Both tutor and student used larger vocabularies in typing than in speaking.

10. There are a number of differences in discourse strategy:

(a) The tutor asked more questions in the keyboard sessions.
(b) The tutors sought verbal means to express approval and enthusiasm.
(c) The tutors uttered many more imperative sentences in keyboard sessions. This may have been a result of a feeling of loss of control that the tutors experienced in the keyboard sessions.

Several recent papers have described the results of studies of the sessions collected by the CDS program with data identified by the numbering program. Lee, Evens, Michael, and Rovick's (1991) spelling-correction program is based directly on these data, as is Hume and Evens's (1992). Hume's (1992) paper is based on a study of student modeling in the live tutoring sessions. Sanders, Hume, Evens, Rovick, and Michael's (1992) study looks at student initiatives in the live sessions and suggests strategies for handling these initiatives in CIRCSIM-TUTOR.

Summary

We have used CDS to carry out 28 hour-long keyboard-to-keyboard tutoring sessions. The numbering program has been useful in the analysis of the language and the content of the sessions. The results of this analysis have been used at every step in the design and development of an intelligent tutoring system for cardiovascular physiology. The language analysis has been essential to the design of language understanding and generation modules. The tutoring rules are used extensively by our planning component. The student modeling rules that we have inferred are essential to the function of the student modeler.

Although we have only used these programs to capture tutoring sessions, we believe that they can be of use to anyone who needs to capture computer-mediated dialogues. We have found them fast, reliable, and easy to use.

Both of these programs run under PC-DOS. To obtain a copy of either of them, send net mail to Martha Evens (her current address is csevens@minna.iit.edu) or send a PC-DOS diskette to Joel Michael.

REFERENCES

Fox, B. (1990). *Final report to the office of naval research*. Boulder, CO: University of Colorado, Department of Psychology.

Grishman, R., & Kittredge, R. (1986). *Analyzing language in restricted domains: Sublanguage description and processing*. Hillsdale, NJ:Erlbaum.

Grosz, B. (1978). Discourse analysis. In D. Walker (Ed.), *Understand-*

ing spoken language (pp. 138-176). New York: Elsevier-North Holland.

Hartson, H. R., & Hix, S. (1989). Human-computer interface development: Concepts and systems. *ACM Computer Surveys*, **21**, 5-92.

Hayes Microcomputer Product, Inc. (1989). *Smartcom III reference manual*. Atlanta: Author.

Hix, D., & Schulman, R. S. (1991). Human-computer interface development tools: A methodology for their evaluation. *Communications of the ACM*, **34**(3), 74-87.

Hume, G. (1992, June). A dynamic student model in an intelligent cardiovascular tutoring system. *Proceedings of the Fifth Computer-Based Medical Systems Symposium*, Durham, NC. Los Alamitos, CA: IEEE Computer Society Press.

Hume, G., & Evens, M. (1992). Student modeling and the classification of errors in an intelligent cardiovascular tutoring system. *Proceedings of the Fourth MAICSS Conference* (pp. 52-56), Starved Rock, IL. Carbondale, IL: Southern Illinois University, Computer Science Department.

Kim, N., Evens, M., Michael, J. A., & Rovick, A. A. (1989). CIRCSIM-TUTOR: An intelligent tutoring system for circulatory physiology. In H. Mauer (Ed.), *Proceedings of the International Conference on Computer Assisted Learning* (pp. 254-266). Berlin: Springer-Verlag.

Lee, Y. H., Evens, M., Michael, J. A., & Rovick, A. A. (1990). IFIHS: Ill-formed input handling system. *Proceedings of Second Midwest Artificial Intelligence and Cognitive Science Society Conference* (pp. 93-97). Carbondale, IL: Southern Illinois University, Computer Science Department.

Lee, Y. H., Evens, M., Michael, J. A., & Rovick, A. A. (1991). Spelling correction for an intelligent tutoring system. *Proceedings of the First Great Lakes Computer Science Conference* (pp. 77-83). New York: Springer.

Sanders, G., Hume, G., Evens, M., Rovick, A., & Michael, J. (in press). An analysis of how students take the initiative in keyboard-to-keyboard tutorial dialogues in a fixed domain. *Proceedings of the Cognitive Science Conference* (pp. 1086-1091). Hillsdale, NJ: Erlbaum.

Seu, J. H., Chang, R-C., Li, J., Evens, M., Michael, J. A., & Rovick, A. A. (1991). Language differences in face-to-face and keyboard-to-keyboard tutoring sessions. *Proceedings of the Cognitive Science Conference* (pp. 576-580). Hillsdale, NJ: Erlbaum.

Seu, J. H., Evens, M., Michael, J. A., & Rovick, A. A. (1991). Understanding ill-formed input to an intelligent tutoring system in an LFG framework. *Proceedings of the Third Midwest Artificial Intelligence and Cognitive Science Society Conference* (pp. 36-40). Carbondale, IL: Southern Illinois University, Computer Science Department.

Thompson, B. H. (1980). Linguistic analysis of natural language communication with computers. *Proceedings of the International Conference on Computational Linguistics*, Tokyo.

Wenger, E. (1987). *Artificial intelligence and tutoring systems*. Los Altos, CA: Morgan Kaufmann.

Woo, C. W., Evens, M., Michael, J. A., & Rovick, A. A. (1991). Dynamic planning in an intelligent cardiovascular tutoring system. *Proceedings of the Fourth Annual IEEE Symposium on Computer-Based Medical Systems* (pp. 226-233). Los Alamitos, CA: IEEE Computer Society Press.

Zhang, Y., Evens, M., Michael, J. A., & Rovick, A. A. (1990). Extending a knowledge base to support explanations. *Proceedings of the Third IEEE Conference on Computer-Based Medical Systems* (pp. 259-266). Los Alamitos, CA: IEEE Computer Society Press.

NOTE

1. It is our experience that if any other program is run prior to the CDS session, there is a possibility that the CDS program will "freeze." These programs include some DOS commands such as "type." Otherwise, the CDS program runs well.

(Manuscript received June 20, 1991;
revision accepted for publication July 10, 1992.)