DOCUMENT RESUME

ED 290 627 SE 048 858

AUTHOR	Haase, Howard; And Others
TITLE	Design Features of Pedagogically-Sound Software in Mathematics.
INSTITUTION	Massachusetts Univ., Amherst. Dept. of Physics and Astronomy.
SPONS AGENCY	Digital Equipment Corp., Maynard, Mass.; EXXON Education Foundation, New York, N.Y.
PUB DATE	27 Apr 85
NOTE	16p.
PUB TYPE	Reports - Descriptive (141) -- Guides - Classroom Use - Guides (For Teachers) (052)
EDRS PRICE	MF01/PC01 Plus Postage.
DESCRIPTORS	*Algebra; *Computer Assisted Instruction; Computer Oriented Programs; *Courseware; Material Development; Mathematics Curriculum; Mathematics Education; *Mathematics Instruction; *Problem Solving; Programing; Secondary Education; *Secondary School Mathematics

ABSTRACT

Weaknesses in educational software currently available in the domain of mathematics are discussed. A technique that was used for the design and production of mathematics software aimed at improving problem-solving skills which combines sound pedagogy and innovative programming is presented. To illustrate the design portion of this technique, a "storyboard" for a sample problem from elementary algebra is presented. (Author/PK)

Design Features Of Pedagogically-Sound Software in Mathematics*

Howard Haase, Ronald Marion and Jose Mestre
Cognitive Development Group
Department of Physics and Astronomy
University of Massachusetts
Amherst, MA 01003

April 27, 1985

ABSTRACT

Weaknesses in educational software currently available in the domain of mathematics are discussed. We present a technique that we use for the design and production of mathematics software aimed at improving problem-solving skills which combines sound pedagogy and innovative programming. To illustrate the design portion of our technique, we present a "storyboard" for a sample problem from elemetary algebra.

* Work supported in part by grants from Exxon Education Foundation and Digital Equipment Corporation. The views contained herein are those of the authors and do not necessarily reflect the position, policy, or endorsement of these two organizations.

The increased availability of microcomputers in the late 1970s was viewed with high hopes by many in the educational community. Since then, the presence of computers in the classroom has met with mixed reviews. There are a number of reasons for this response. They include the ambivalent attitudes of teachers, students and school administrators toward computers, the high costs of computer hardware and software, and the poor quality of much of the educational software now available. In this paper the major shortcomings of currently available educational software will be discussed. In addition a design strategy for overcoming these weaknesses will be presented. Finally, a "storyboard" will be presented as an example of one stage of this design strategy.

The most significant shortcoming of available mathematics educational software is the paucity of material devoted to teaching problem solving skills. The National Assessment of Educational Progress (1983) survey found that most students did well in exercises that assessed routine math skills, but performed poorly on multi-step problems which assessed higher level cognitive skills. For example, both research and remedial mathematics instruction conducted by our group indicate that translating statements expressing mathematical relationships into algebraic notation is very difficult for most students. In light of these findings our primary goal has been to develop educational software which will both assess and improve problem solving skills.

Another serious problem with existing educational software is that it is unimaginative. This problem occurs because most software designers insist on using printed teaching aids, such as workbooks, as their model. Such an approach ignores the unique capabilities of the computer. Consequently, software developed in this way is of the drill-and-practice or electronic

workbook type in which little meaningful interaction is required on the part of the student. In this type of software student interaction is limited to an occasional yes/no response or to answering multiple choice questions. In other words the computer display mimics a printed page. With over 90% of available mathematics software being of this type, it is not surprising that many teachers and school administrators are ambivalent about computer-assisted instruction. In this period of tight budgets school administrators cannot be expected to invest in expensive "teaching machines" which are no better than traditional educational material. In future applications, the computer's analytic power and large memory capability must be used to create a more flexible learning environment which can respond to individual student needs.

Our primary goal in developing mathematics educational software is to produce a pedagogically sound learning environment which uses the computer's unique capabilities . One of these capabilities is the computer's ability to analyze and respond to free-form answers. For example, suppose we want to see if the user understands a certain functional relationship. To do so, we could have the computer ask the question, "What happens to Y as X increases?" One unexciting way to elicit a response is to ask the user to select from among the multiple choices: a) increases, b) decreases c) stays the same. Responses of this kind leave very little opportunity for the computer to analyze error patterns. A more interactive approach would be to let the user input a free-form response. This would give the user the freedom to choose from a number of possible answers, such as "increases", "gets bigger" or "shrinks", which the computer could later interpret. Given the analytical power of computers, recognizing a number of acceptable free-form answers is a perfectly reasonable feature to incoporate into the software. In the odd event that the student's response is not in the set of recognizable choices, a message would

be displayed on the screen telling the user that the computer cannot recognize his/her answer and that s/he should try to respond in a different way.

A more important reason for using free-form responses is that they allow us to look for error patterns which may be indicative of student misconceptions. This feature is important because in many cases these misconceptions interfere with learning. A number of research studies conducted by our group (Clement, 1982; Clement, Lochhead and Monk, 1981; Lochhead, 1980; Mestre, Gerace and Lochhead, 1982; Mestre and Lochhead, 1983; Rosnick, 1981; Rosnick and Clement, 1980) have revealed that a wide spectrum of people are not able to correctly translate statements relating two variables into algebraic equations. We have consistently found the same difficulties among students from five distinct cultural groups, among college undergraduates majoring in technical fields like engineering, and among both high school and college faculty members. These studies reveal two misconceptions which are very difficult to overcome. One consists of translating the mathematical statements into algebraic equations using a left-to-right word-order match with little regard to the mathematical content of the statement. The other consists of treating mathematical variables as if they were labels.

Our findings also indicate that students who hold a particular misconception do not reach the appropriate level of understanding when they are simply shown the correct answer or solution. This evidence suggests that misconceptions are deep-seated and require a sustained effort to be dislodged. However, once these misconceptions are dislodged students are able to accomodate the correct concepts. In response to these findings, we suggest the following guidelines when designing math problem-solving software. First, the software should anticipate student misconceptions. In order to accomplish this goal designers must be familiar with common error patterns that students

reveal when solving the problems of interest. This information can be obtained by consulting appropriate studies in educational research; if research on the types of problems of interest is not available, then a pilot research study can help identify areas where students experience difficulties. With a good understanding of those areas that cause difficulties, software can be designed that will detect common error patterns by analyzing the free-form responses of the student. Next, a conflict must be established in the student's mind between his/her answer and the correct solution to the problem. This can be accomplished by means of a series of carefully designed questions in the form of a computer-directed 'Socratic dialogue'. In this part of the program the student is led by a series of logical steps to the realization that his/her answer is inconsistent with some aspect of the problem. Finally, the student is guided through another interactive, computer-directed dialogue which helps him/her reach the appropriate level of understanding.

Today's software does not possess features which allow an instructor to analyze the cognitive processes used by students engaged in solving problems. One way to use computers as an interface between the student's thought processes and the teacher's ability to interact with these thought processes is to develop software which keeps a permanent record of student progress towards the solution of a problem. The instructor can then make use of this information when planning class lectures and tutorials. Including this type of feature in software would help increase the quality of personalized instruction within a classroom setting.

Educational mathematics software is often both designed and coded by the same person. We feel that this method should be modified since this person is rarely both an expert educator and an expert coder. Software which has been

innovatively programmed, complete with video-arcade graphics and sound, but which fails to use effective teaching methods may provide the student with a good time but will do little to educate. Similarly, software which is designed soundly from an educational standpoint may fail to motivate if it looks uninteresting. Obviously what is needed is a combination of sound pedagogy and innovative coding.

Our development approach divides tasks between people with different areas of expertise. The design is undertaken by experts in pedagogy, namely teachers and educational researchers, while the programming is carried out by expert coders. Separating these two tasks assures that each expert will focus on his or her area of expertise. We should quickly point out that we are not advocating that the design and coding tasks be divorced; rather, we advocate a collaborative effort in which the coders advise the designers on ways to maximally exploit the full power of the computer to accomplish the stated aims. For example, educators may not be familiar with the computer's ability to highlight or blink text, to delay before and after critical words or phrases, to produce pictures and graphics, or to divide the screen into a number of independent windows. The task of the educators in this collaboration is to produce a "storyboard", that is, the detailed flow-chart of the content of the module designating all the decisions, messages, and branch points that will occur at every step along the way. The storyboard is then turned over to an expert programmer who codes the storyboard in the appropriate computer language.

These two factors, an emphasis on improving higher-level problem-solving skills and a division of the design tasks between educators and coders should result in software that is both innovative and educationally sound. We now provide an example of a storyboard for a problem from elementary algebra.

REFERENCES

Clement, J. (1982). Algebra word problem solutions: Thought processes underlying a common misconception. Journal for Research in Mathematics Education, 13, 16-30.

Clement, J., Lochhead, J., & Monk, G. (1981). Translation difficulties in learning mathematics. American Mathematical Monthly, 4, 286-290.

Lochhead, J. (1980). Faculty interpretations of simple algebraic statements: the Professor's side of the equation. Journal of Mathematical Behavior, 3, 29-37.

Mestre, J.P., Gerace, W.J. and Lochhead, J. (1982). The interdependence of language and translational math skills among bilingual Hispanic engineering students. Journal of Research in Science Teaching, 19, 399-410.

Mestre, J.P. and Lochhead, J. (1983). The variable-reversal error among five cultural groups. In J. Bergeron and N. Herscovics, (Eds.), Proceedings of the Fifth Annual Meeting of the North American Chapter of the International Group for the Psychology of Mathematics Education, Montreal, Canada, Sept. 29-Oct 1, pages 180-188.

Rosnick, P. (1981). Some misconceptions concerning the concept of variable. The Mathematics Teacher, 74, 418-420.

Rosnick, P., & Clement, J. (1980). Learning without understanding: The effect of tutoring strategies on algebra misconceptions. Journal of Mathematical Behavior, 3, 3-27.

SAMPLE STORYBOARD

The following seven pages contain a sample storyboard for the following problem:

> The amount of money collected at an auction was $1000 more than the estimate made by the organizers. If the organizers estimated that E would be collected at the auction, write an equation which expresses the actual amount of money collected at the auction, M, in terms of E.

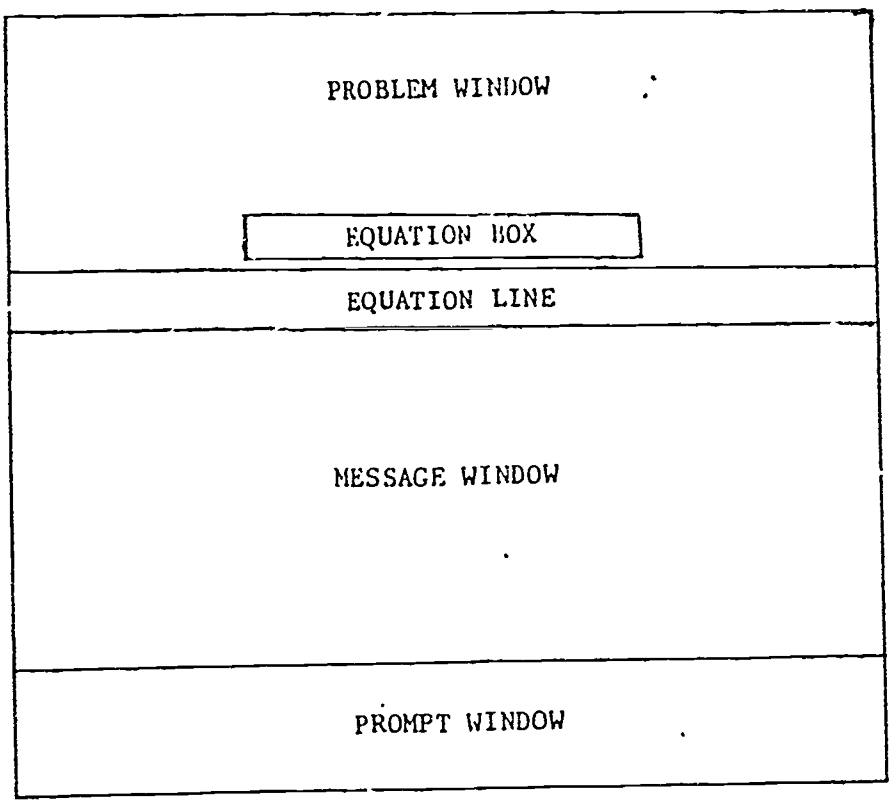
Certain details are not covered in the storyboard, such as the conventions that will be adopted by both the designers and the coders to ensure consistency. The notation used in the storyboard goes as follows:

1) Curly brackets are notes to the coder (i.e. {...}).

2) Square brackets denote inputs (i.e. [...]).

3) Tests of inputs are enclosed between slashes (i.e. /.../).

The screen design will make use of techniques such as text highlighting, flashing, and graphics. Most of the interaction will make use of the following screen design:



How these portions of the screen will be used is delineated in the storyboard.

1. (According to some pre-determined selection process, select a problem from the "problem library" and present it in the Problem Window)

(In Message Window, write)
"Type in your answer and press RETURN"

2. [Input S$]

(where by S$ we mean a string variable containing a response.
In this case, the response is an equation.
Analyze the input character by character as follows:

a) Accept and print in the Equation Line any of the following characters: m, M, e, E, +, -, =, 1, 0. Print E and M only in Caps. Accept M, m, E, e, and =, only once; if the student attempts to enter any of these a second time, don't print them, give a short beep, and write the following message in the Prompt Window: "This character is only allowed once."
Note that the set of allowed characters is problem-dependent. The set of allowed characters will reside in a data file corresponding to the particular problem.
However, any allowed characters which are letters will only be allowed once in any problem. Further, the equal sign, =, will only be allowed once in any problem.

b) If the student enters / or *, do not print, give short beep, and give the following message in Prompt Window: "The problem does not require multiplication or division."

c) If the student enters ".", ",", or "$" do not print, beep, and give the following message in the Prompt Window: "The equation can be written without this symbol." This will apply to all problems.

d) If student tries to enter any other key, beep, and give the following message in Prompt Window: "We don't recognize that symbol. Try again"

As soon as student enters equation, rewrite it in the Equation Box provided under the Problem Window)

3. /Is S$ an equation?/
(That is, does it have an "=" sign with something on both sides of it?)

(In Prompt Window:) 2 NO 1 (In Prompt Window:)

"An equation must have an equal sign."

"An equation must have an equal sign. Try again."

YES

4. /Is the constant consistent with the problem?/
(That is, has the student made up a constant that is not part of the problem. This information is problem-dependent. In this case, accept only 1000 and 0. If 0 appears in eqn. by itself, make sure it is isolated on one side of the equation. If any other number appears flash that number in the Equation Line)

NO-->-- (In Prompt. W.:) "The number flashing in your equation is incorrect. Try again" ->->->

YES

5. /Is the syntax of the equation legitimate?/
(For example, can't accept "A++-B=--C". Need to have only one operation between variables, etc.)

-<-<-<-(In Prompt W.)-<---<---<---<-----<--2 NO 1->-- (In Prompt W.)

"The form of your equation looks strange."

"The form of your equation looks strange. Try again" -->-->->->->->->->->->-->-

YES

(In Message W:) "You seem to be having some difficulty. Press:
T To try again
S To see the solution to the problem
Q To quit"

[INPUT] ---<---<---<---<----

6. /Is the equation the same as "M=E+1000" or "M=1000+E"?/

YES

(Give random good response in Prompt Window, and go to next problem)

/Is it "T"?/

<-<-<-<- YES NO

NO

7. / Is the equation the same as "E+1000=M" or "1000+E=M"?/

/Is it "Q"?/ YES-->--END

<-<-<-<-<-<-<-<- NO YES

NO

(Write in Message Window:)

/Is it "S"?/ NO -- BEEP --->--

"Your equation is correct! But usually we put the variable being asked for on the left hand side, like we are now showing above"

YES

(Wait about 8 seconds after this message appears. Then erase the equation from the Equation Box. Re-write it in the standard form, in this case "M=E+1000", and highlight it. Then, in Prompt Window, write the following message in highlighted form:)

(Branch to Problem Solution)

END OF THIS BRANCH

"Press Return to go to the next problem"

->->->->->->->->->->

v <-<-<-<-<- YES <-<-<-<-<-<- 8. /Is it some equation equivalent to M=E+1000?/ NO->->

(In Prompt Window:)
"Your equation is correct! it However, let's try to write in the way the problem asks"
(Wait 10 seconds and continue)

/Does the equation begin with M= , where M is variable being asked for in the problem?/

<-<-<-<- YES NO

(In Prompt W. write and wait 10 seconds:)
"Let's write your equation in the way that the problem asks and see if it's correct"

(In Message Window:)
"Look at the highlighted portion of the problem above. This means that M should be alone on the left hand side of the equation. try to rewrite your equation with M alone on the left hand side of the "=" sign, and press return."
(highlight portion: "write an eq ... in terms of E")

(In Message W. write)
"Look at the highlighted portion of the problem above. This means that M should be alone on the left hand side of the equal sign. Hit return to continue."
(Highlight portion: "write an eq ... in terms of E". Wait for student to type in return. If more than 20 seconds pass, flash the phrase "Hit return to continue" in the Message W.)

9. /input/ <-<-<-<-<-<-<-<-<-<-<-<-<-<-<-<-<-<-<-<-<-<-

(If no response in 5 seconds

(When student hits Return, write the student's equation in canonical

type in Prompt Window and flash it:)
"Please type in your answer and press return"

(Analyze input character by character.
Accept only the character which represents
the variable asked for, and an equal sign.
In this case, M= . If a "-" is hit fi
do not print and give the following
in the Prompt Window)
"You don't need a minus sign on the left hand
side of the equal sign. Try again"
(If any other key is hit, beep, and give the
following message in the Prompt Window)
"Read the problem and the hint above, and try again"

-(--(- YES /Student makes more than 3 mistakes?/

NO

(If the student types in M=, accept everything
after the equal sign according the instructions
in 2 and 4 above. When student finishes typing
equation and presses return, freeze the equation
in the Equation Line for comparison
with the one in the Equation Box.
Erase message in ... W.)

/Is the equation in the Equation Line
M= E+1000 or M=1000+E?/

YES

(Random good response in Prompt Window.
Go to next problem)

-(-(-(-(-(-2 NO

(In Prompt. W:)
"The equation you've just typed in is not the
same as your orignal equation. Try again."
(Erase everything in Message W, and type in it
the hints in the 8-YES branch above, i.e. "Look
at the highlighted...)

-)

form in the Equation Line. Erase Message
Window, and write in it:)
"So you see how we have put M alone on the
left hand side of the equal sign. We're
doing that because the highlighted portion
of the problem asks for M."
(Highlight portion: "write an eq ... in terms
of E".)

(In Prompt Window write the following
message and flash it if the student
does not repond in 15 seconds:)

"Press Return to continue"

(After student types return, continue)

|(-(-(-(-(-(-(-(-(-(-(-

16. /Is it "M=E-1000" or "M=-1000+E" or "M=(-1000)+E" ?/ ---)---)--- YES

(In Message Window, write)
"Unfortunately, your answer is not Correct. Suppose that the estimate, E, is $4000. According to your equation, how much money, M, was actually collected? Please give your answer as a number."

NO

[Input]

NO -----(----(- /Is the input a numerical value?/

(in Prompt Window)
"Please type a number for M."

YES

/ Is it 3000?/

(In Message Window) YES ---(-----(---
"You're right! If E is 4000 your equation predicts that M is 3000. But the problem states that more money was collected than was estimated. So M must be greater than E which is not what your equation predicts. Try again."

(- 2 (- NO -) 1 -)

(In Prompt Window. write:)
"No. Substituting E=4000 into your equation does not give ______. Try again."
(The "____" is the answer given by the student. Route student back up to input and erase his previous input)--)

(In Message Window)
"Your answer is still wrong. Substituting E=4000 into your equation gives:
M=4000-1000=3000.
However, the problem says that the money collected, M, was $1000 more (highlight "more") than the estimate, E. So you see why your equation can't be correct. Try to write a correct equation.
(If no response in 30 sec. write in Prompt W.:) "We're waiting for you to type in an equation.

17. /Is it "M=1000-E" or "M=(-E)+100^ or M=-E+100"/ --)-- NO --)-)-)-)-)-)-)-)--)- 18. /Is it "M=-E-1000" or "M=(-E)+(-1000) or "M=(-E) -1000" or "M=-1000-E" or "M=(-1000)-E" or "M=(-1000)-(-E)?/

YES

NO
(There should not be a NO branch here if all the tests were carried out appropriately)

YES

(In Message Window:)
"Your answer is not correct. Suppose that the estimate, E, is $4000. According to your equation, how much money, M, was actually collected? Please give your answer as a numerical value."

(In Message Window:)
"Your answer is not correct. Suppose that the estimate, E, is $4000. According to your equation, how much money, M, was actually collected? Please give your answer as a numerical value."

[INPUT]

[INPUT]

/Is it a numerical value?/ --NO -)- (In Prompt Window:)
"Please type in a number for M."

YES

/Is it a numerical value?/ NO-)- (In Prompt Window:)
"Please type in a number for M."

-- YES--)--

/Is it "-3000" or "-(3000)"?/ NO, 1 --)----

(In Prompt Window:)
"No. Substituting E=4000 into your equation does not give ____. Try again"
("___" is the student's answer)

--(-- 1, NO -(-- /Is it "-5000" or "-(5000)"?/

YES

NO 2

NO 2,

YES

(In Message Window:)
"Your answer is still wrong. Substituting E=4000 into your equation gives:
M=1000-4000=-3000.
This implies that the auction lost (highlight "lost") $3000 However, the problem states that the money collected, M, was $1000 more (highlight "more") than the estimate, E. So you see why your equation is wrong. Try to write a correct equation."

(In Message Window:)
"Your answer is still wrong. Substituting E-4000 into your equation gives:
M=-4000-1000=-5000.
This implies that the auction lost (highlight "lost"). However, the problem states that the money collected, M, was $1000 more (highlight "more") than the estimate, E. So you see why your equation is wrong. Try to write a correct equation."

7/7

(In Message Window:)

"You're right! If E=4000, your equation predicts that M is -3000, but this would mean that the auction lost (highlight "lost") $3000. However, the problem states that the money collected, M, was $1000 more (highlight "more") than the estimate, E. Think about this and try to type in the correct equation."

(In Message Window:)

"You're right! If E=4000, your equation predicts that M is -5000, which would mean that the auction lost (highlight "lost") $5000. However, the problem states that the money collected, M, was $1000 more (highlight "more") than the estimate, E. Think about this and try to type in the correct equation."