TASSY—One approach to individualized test construction

by THOMAS L. BLASKOVICS and JAMES A. KUTSCH, JR.

West Virginia University
Morgantown, West Virginia

During the past ten years universities and the computing industry have seen the development of a new mode of teaching called Computer Aided—or Assisted—Instruction (C.A.I.). This new field, emerged as an attempt to meet and deal with the growing criticism and frustration of students, employers, legislators, and faculties, which stemmed from our inability to prepare students adequately.

Several very creative C.A.I. projects were directed toward providing a whole new system of instruction. However, to date, the success of C.A.I. has been limited, at best. PLANIT, PLATO, LYRIC, COPI, COURSEWRITER, and others have not been able to meet the needs of the teaching community. The problems reported by the major projects are only in part bounded by the technology of the computer.

At West Virginia University, we watched the development of these systems with great interest and concern, because we, like other universities, were faced with the same problems. We carefully examined several of the better-publicized systems with an eye toward implementing one of them to meet our instructional needs. As we analyzed the systems we discovered that C.A.I. systems:

1. Were too machine-dependent to allow a feasible implementation without scrapping our existing hardware (an IBM 360/75).
2. Were too expensive in terms of Core requirements.
3. Made (what appeared to be) unreasonable demands upon the instructor in terms of intimate knowledge of programming and/or computer technology.
4. Were “not yet available but would be soon” even though the projected date had slipped by several times.

We also found that the present C.A.I. systems were too monolithic. The decision to change a course or set of courses to a C.A.I. approach requires a “go-for-broke” commitment. We found that users did not like the non-incremental requirement of C.A.I. This was not too surprising to us when we considered that most of the end users who we were concerned about had little or no experience with computers.

Because of the many know factors in C.A.I. systems and the type of commitment required, many potential users we surveyed were reluctant to commit themselves and their resources to a C.A.I. effort.

As we were considering the major C.A.I. systems, we were also engaged in trying to determine what our students felt to be their own needs with regard to a college education. Along with study of student needs, we tried to discover the needs of the faculty with respect to their problems in teaching. The results of our study suggested that two problems existed. Students indicated that a major source of frustration (and possibly aggression) toward the University stemmed from a lack of feedback from the “establishment” regarding individual progress. A second, and only slightly less important, frustration was stated to be the lack of relevance of course material. Interestingly enough, the students felt that given the feedback, they would be able to deal with the problem of relevance themselves. Student evaluations of professors indicated that where amount of feedback was high, and good, relevance was not a problem. The faculty, by and large, agreed with the students feedback was a problem; however, they saw relevance as being more critical to the learning process. The faculty also indicated that they did not have an easy means to provide feedback.

In analyzing our findings, it appeared to us that one reason for the limited success of C.A.I. in other universities was possibly that it might have been the wrong solution for the problem facing the university at the present time.

One of the claims of C.A.I. is that it allows the instructor to individualize: to tailor the instructional
experience to the student through a series of incremental feedback statements. This seemed to imply that the instructor would be spending more time with the student. Our observation of ongoing C.A.I. systems indicated that, once the horrendous task of programming had been accomplished, the instructor retreated to his office or lab either to write new programs, to collapse, or to become more deeply involved in his own research again. The net effect of C.A.I. was to make the student more dependent for help upon the machine, or if he were available, upon a graduate assistant. In some cases, the instructor developed a bad case of “Blinking Light Syndrome” and spent his time diddling with the machine.

Since C.A.I. seemed fraught with problems, we decided to look more closely at the problems of feedback. We made an assumption (tentatively) that the instructor was able to teach the material. Analysis of the instructor’s time indicated that he spent a large portion of time developing, administering, and scoring exam questions, keeping track of what his teaching assistants were doing, worrying about the security of his files—and precious little time actually teaching. In most classes, the most feedback that students received were scores on one or two major exams and the final. In almost all cases we observed, the feedback was delayed until as much as two to three weeks following the respective tests. The students had virtually no opportunity to analyze their performance, or to learn where their deficiencies might be. It appeared that the feedback system we were currently using could not be seen, by any stretch of the imagination, as a learning experience. (In some cases it was viewed by the students simply as a means of satisfying the sado-masochistic tendencies of the faculty.)

In an analysis of one undergraduate course of 325 students, we found that the teaching assistants spent better than 10 percent of their time in purely mechanical activities, such as distributing, proctoring, and scoring tests. In addition, a course manager spent approximately 50 percent of her time doing clerical work necessary to keep the student’s grades up-to-date, etc. Initially, we felt that the time commitments were rather high; until we realized that the assistants and course manager had almost total responsibility for 2,275 hours of testing in one semester.1

Further consideration of the function of efficient feedback suggested utility for both students and instructor. For the former, it provides (1) a test score, (2) a diagnostic evaluation of material learned (and/or not learned), and (3) (hopefully) some prescription to remediate his problem.

In 1971, Baker presented the state of the art in Computer Based Instructional Management Systems (CBIMS). He suggested that the instructor is not only a teacher, but also the manager of a rather complex system of activities designed to help the student learn something.2

He indicated that “a major facet of this managerial task is composed of the mechanical tasks of scoring homework, test papers, and keeping track of what instructional materials a student has used.” Our discussions with faculty and teaching assistants tended to support the notions put forth by Baker.

Each of the systems Baker reviewed was designed to provide for the four major functions of any CBIM system, namely: test scoring, diagnosing, prescribing, and reporting. However, the actual operation of the systems seemed to be very awkward, and required that the student participate on some fixed schedule. Another difficulty that we observed was that the present CBIM systems seemed to double the work of the faculty member in that he had to develop essentially two sets of testing material—one set for the diagnostic function and another for the examination function.2

It appeared to us that if a system with interactive capabilities could be developed, it might resolve much of the awkwardness and restrictiveness we had observed. With regard to the second problem, having to maintain double sets of items, we asked the faculty why not let the students use the real thing for both diagnosis and evaluation. The rationale for this approach was that most professors have, over time, established large item pools from which they draw, in some more or less random manner, to make up any quiz. In addition to their own item pools, many instructors use items suggested from the instructor’s manual or handbook that accompanies the text being used that particular semester. Additional verification of this approach to test-design was accomplished by looking at the exam files in the fraternity and sorority houses at our campus.

Because of all of the above considerations, we decided to develop an automatic Testing And Scoring System (TASSY).

We felt that TASSY should have the following specifications:

1. it should be easy to use by both the student and the instructor;
2. it should allow for immediate feedback to the student;
3. it should allow the instructor, on demand, to review the progress of a student;
4. it should allow the student to individualize his request for proficiency;
5. it should have a high degree of security;
(6) it should meet at least the minimum needs of the registrar for recording grades;
(7) it should allow the student the option of taking an exam for diagnostic purposes or for grade purposes;
(8) it should maintain a record of each student's individual performance for instructor analysis of items.

TASSY'S PROGRAMMING STRUCTURE

TASSY takes the form of a main driving program with several small sub-routines. This structure was necessary because of the constraints of the Conversational Programming System (CPS) with West Virginia University's IBM 360-75. There is a limitation of four pages (each of four thousand bytes) placed on any CPS program. However, through the use of external procedures, a much greater effective program size can be attained (provided that not more than six thousand bytes are in the work space at any time).

The driving program consists of the “welcome” and “exit” lines as well as the calls for all the sub-routines.

When a student enters TASSY, he is first asked if he would like to see some operating instructions for the system. If he replies yes, an instruction sub-routine "HELP" is called. Next, a password routine, "PASS", is called. Here, it is determined whether he is permitted entry to the system. If the password is recognized as that of an instructor, the user has an option of seeing special operating instructions from "MORE HELP" (restricted to only instructor mode). The instructor then has the option of "UPDATE" or "DUMP" (described later).

If the password is recognized as that of a student, a call to sub-routine "GENER" is issued (also described later).

A third alternative is that the password is recognized as a master password, allowing access to control mode. From this mode, a system manager has the option of "UPDATE", "DUMP", "GENER", or "WHO". The system manager has access to any course.

When a user (student, instructor, or system manager) is finished, control is returned to the driving program where a "good-bye" line is printed. Then the system is ready for the next user.

ROUTINE "PASS"

The password routine has the main purpose of determining whether a given user is authorized to be in the system, and, if so, in what mode. A student password is given by the proctor to a student when he enters the testing center. This password varies systematically each hour of the day, and can be reset by the system manager each day or week as necessary.

An instructor's password consists of any combination of up to six letters, numbers, or special characters and is chosen by the instructor. If an instructor's password is recognized, a further check is made on the name entered by the user. After passing both checks, control is returned to the driving program, passing back a code to indicate that this user is authorized for instructor's mode. Also, the number of the course in which this instructor belongs is returned.

If the master password is found, a code is returned to the driving program to indicate that this user is the system manager and is authorized for anything.

All passwords, including the instructor's and the master password can be altered at any time by the system manager.

An added feature of the password check is the activity file (WHO). A record is written to this field when an instructor's password is found or when a user is not permitted entry in any mode, i.e., when he has entered an invalid password. This record contains the password used, the first and last name and ID number as entered by the user, and the date and time of his entry into the password procedure.

It was decided not to record valid student entries for two reasons. First, the number of entries would be great, and second, they are recorded as a part of "GENER".

ROUTINE "GENER"

This routine is called to generate, print, score, and record a student's examination. It is probably one of the most important components of TASSY.

Upon entering the routine the student is asked the course, section, and quiz number he desires (his name and ID number have been passed from the driving program).

From the course number, the appropriate question file is opened. Then, from a control record in the file, and from the entered quiz number, the type and number of the items to be given is determined. These items are generated from the file at random (except that there has to be the prescribed number of each type designated by the control record).

One rather interesting problem developed in choosing the algorithm for random generating of questions. The records are stored in the file in the order of entry. Along with each record is the attribute of the given question. From the control record, the desired attributes for a given quiz, and the number of each which should be generated, is obtained. In an early version of the system,
a random number between one and the number of questions in the file was generated by the built in function in CPS. The corresponding record was then read, and a check was made to determine if the record was of the correct attribute. If not, the number of this record was saved in a vector of 'used' items. If the item had the correct attribute it was printed as described below, and its number was also saved. As each new item number was generated, it was compared against the growing list of used items. This procedure prevents duplication of items on a given test. As the test proceeded from the attribute being used to the next as defined in the control record, the vector of used items was cleared for use by the next attribute.

The course number of the desired course (4 digits) demo
Which section? (two digits) 01
What quiz do you want? (1 through 9) 2
Enter yes to activate the verification option yes

You will now be given the requested test.

Good luck.

Question 1:
Elapsed time: .083 minutes
THE DIFFERENCE IN VAPOR PRESSURE BETWEEN SOIL AND ATMOSPHERE IS EQUILIBRATED BY PLANTS THROUGH:
1: ROOT HAIRS
2: OSMOSIS
3: TRANSPERSION
4: TRANSLOCATION
5: TURGOR

Your answer please:
3
Verify: 3?
Correct

Question 2:
Elapsed time: .566 minutes
GRANULAR ENDOPLASMIC RETICULUM CAN BE FOUND IN
1: ONLY PLANT CELLS
2: CELLS ACTIVE IN PROTEIN SYNTHESIS
3: CELLS WHICH ARE ABOUT TO DIVIDE
4: ALL OF THE ABOVE
5: NONE OF THE ABOVE

Your answer please:
4
Verify: 4?
Sorry, the correct response is 2

End of exam. Please wait while results are compiled
You have responded correctly to 1 out of 2 questions or 50.000 percent.

Breakdown of score
Attribute A 1 right out of a possible 1 or 100.000 percent
Attribute T 0 right out of a possible 1 or 0.000 percent

Mode?
stop
Thank you for your interest in our computerized testing service. Please come back again.
The following students have recently taken exams:

<table>
<thead>
<tr>
<th>NAME</th>
<th>ID</th>
<th>SECT</th>
<th>QUIZ</th>
<th>DATE</th>
<th>TYPE 1</th>
<th>TYPE 2</th>
<th>TYPE 3</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHANNES</td>
<td>JEAN</td>
<td>236694886</td>
<td>01</td>
<td>3</td>
<td>07/21/72</td>
<td>3</td>
<td>80.00</td>
<td>0</td>
</tr>
<tr>
<td>CHANNES</td>
<td>JEAN</td>
<td>236694886</td>
<td>01</td>
<td>2</td>
<td>07/21/72</td>
<td>2</td>
<td>90.00</td>
<td>0</td>
</tr>
<tr>
<td>JELLINKE</td>
<td>HOLLIS</td>
<td>237760553</td>
<td>02</td>
<td>5</td>
<td>07/21/72</td>
<td>5</td>
<td>70.00</td>
<td>0</td>
</tr>
<tr>
<td>JELLINKE</td>
<td>HOLLIS</td>
<td>237760553</td>
<td>02</td>
<td>5</td>
<td>07/21/72</td>
<td>5</td>
<td>90.00</td>
<td>0</td>
</tr>
<tr>
<td>MEADE</td>
<td>TONY</td>
<td>2347615045</td>
<td>02</td>
<td>4</td>
<td>07/21/72</td>
<td>4</td>
<td>65.00</td>
<td>0</td>
</tr>
<tr>
<td>MEADE</td>
<td>TONY</td>
<td>2347615045</td>
<td>02</td>
<td>4</td>
<td>07/21/72</td>
<td>4</td>
<td>70.00</td>
<td>0</td>
</tr>
<tr>
<td>VARGO</td>
<td>JERRY</td>
<td>17541554</td>
<td>01</td>
<td>4</td>
<td>07/21/72</td>
<td>4</td>
<td>75.00</td>
<td>0</td>
</tr>
<tr>
<td>MANN</td>
<td>KAY</td>
<td>234651515</td>
<td>02</td>
<td>4</td>
<td>07/21/72</td>
<td>4</td>
<td>80.00</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 2

As is apparent, if, during generation of items for a given attribute, an item of some other attribute were selected, it would be ignored and its record number would be stored so that this unusable record would not be selected again. It was thought that the machine time used in searching the “used” vector would be less than the I/O time required to keep selecting an unusable record (the records, in this case, must be read before usability can be determined).

What was not realized was that the item vector would grow as fast as it did. As an experiment, the algorithm was changed in a way that the records are read, the test of attribute is made, then the vector is scanned to see if this item has already been used in this test. In this way, many more records are read from the file, but, much fewer comparisons are made in the “used” vector.

The later of the two methods proved much more satisfactory. In the earlier method, when more than fifteen items of the same attribute were generated, real time between items ran 45 seconds, or more, while the time separation between items in the first part of the quiz (items one through eight) was minimal (on the order of two to six seconds).

In the latter method, the separation time was much more uniform from the beginning of a quiz through twenty items or more and was on the order of two to eight seconds. Needless to say, the latter method has been used since the day that the time differential was noted.

(It is thought that locating various attributes in different physical locations in the file may be a useful way of decreasing item generation time even more than what has been attained by the above change.)

As the items are generated, they are printed on the terminal one by one and a reply is requested. Upon entry of this reply, the student is told immediately whether he is correct. If the response is incorrect, the correct answer is given.

As the test is being given, a record is kept of each question, and, by question type, of the number correct.

Figure 3

Mitosis and meiosis are considered to be dynamic processes because:

a. The events are discontinous and discrete that occur randomly
b. The events are discontinous and discrete that occur in a systematic sequence
c. The events are continuous and discrete that occur in a systematic sequence
d. The events are continous and discrete that occur randomly
e. none of the above

Modes? list, insert, define, or stop

Figure 3
update
Course number?
demo
Section number? (two digits)
01

Mode? list, insert, define, or stop
insert

Item number? (four digits)
0012
Attribute? (one character)
9
Correct response? (one character)
3

Question and answers?
What is the correct date for the FALL Joint Computer Conference?
1. July 4
2. December 25
3. December 5
4. Feb. 31
5. None of the above

RECORD 0012 successfully entered

Mode? list, insert, define, or stop
list

Enter the number of the item to be listed
number=12
ITEM 0012 COURSE ID demo SECTION 01
ATTRIBUTE 9 CORRECT RESPONSE 3

What is the correct date for the FALL Joint Computer Conference?
1. July 4
2. December 25
3. December 5
4. Feb. 31
5. None of the above

Figure 4

Mode? list, insert, define, or stop
insert

Item number? (four digits)
0012
course

RECORD 0012 successfully entered

Mode? list, insert, define, or stop
list

Enter the number of the item to be listed
number=12
ITEM 0012 COURSE ID demo SECTION 01
ATTRIBUTE 9 CORRECT RESPONSE 3

What is the correct date of the FALL JOINT COMPUTER CONFERENCE?
1. July 4
2. Dec. 25
3. Dec. 5
4. Feb. 31
5. None of the above

Figure 5
Mode? list, insert, define, or stop

define

Definition of quiz parameters
There is a maximum of three attributes. These may be any letter or number (one character in length)

Please enter the number of the quiz to be defined
1
Enter the number of attribute used in this quiz. Max of 3
number=1
Enter attribute # 1
1 Quantity?
2 Enter attribute # 2
1 Quantity?
2 Enter attribute # 3
1 Quantity?
2 Quiz number 1 is now defined
Mode? list, insert, define, or stop
stop

and number attempted; i.e., the number of questions of that type which were on the quiz. At the end of the quiz, the student is given the totals of questions correct and attempted and the breakdown of this information by question or attribute type. (See Figure 1)

Before control is returned to the driving program, a student record is written onto a file, indicating the name, ID number, section number, quiz number, date, time, and subscores, and total score on the examination. (See Figure 2)

ROUTINE “DUMP”

The “DUMP” routine is used by the instructor to print the records in the student file. In one sense “DUMP” keeps the instructor’s grade book. The formatted file gives the instructor the student’s name, his student number, the date the quiz was taken, the number of the quiz, section number, and a percentage correct breakdown for each attribute and percentage correct for the total quiz.

The LIST function will list a requested item from the file. In the LIST is the correct response, question type (attribute), the question, and the distractors. (See Figure 3)

INSERT is the converse of LIST. It allows the instructor to replace or insert an item in the file.

There are two versions of INSERT. In the more commonly used version, the user is prompted before each entry. After each prompt the user enters the information requested. (See Figure 4)

It was found that this method is time consuming and
At present, the records are printed in chronological order. However, in later versions of TASSY the instructor will have the ability to have the records sorted for his convenience. (See Figure 2)

ROUTINE “UPDATE”

“UPDATE” is a routine for file maintenance of the question file. The user has the following options: LIST, INSERT, and DEFINE.

somewhat boring for the experienced user, especially when large numbers of questions are being entered. Accordingly, a ‘terse’ mode of INSERT was developed. When this mode is requested, no prompts are given. It assumes that the user knows the record structure and the entire record is entered at one time. The basic difference between the two methods is that the information (item number, attribute, correct response, course number, and section number) must be provided to the system in the correct order without prompts, when the ‘terse’ mode is used. (See Figure 5)

In both modes the user must learn only one special character. This is the ‘not’ (¬) sign which is used as a separator between the questions and its answers, as well as between the answers themselves. It should be noted

Figure 6

From the collection of the Computer History Museum (www.computerhistory.org)
Following persons were in the system

<table>
<thead>
<tr>
<th>CG305</th>
<th>YURA</th>
<th>DEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZL-*33</td>
<td>LAROCHE THOMAS</td>
<td></td>
</tr>
<tr>
<td>CG305</td>
<td>YURA</td>
<td>DEE</td>
</tr>
<tr>
<td>2-Z333</td>
<td>SKINNER BECKY</td>
<td></td>
</tr>
<tr>
<td>CG305</td>
<td>YURA</td>
<td>DEE</td>
</tr>
<tr>
<td>53391</td>
<td>YURA</td>
<td>DEE</td>
</tr>
<tr>
<td>53391</td>
<td>YURA</td>
<td>DEE</td>
</tr>
<tr>
<td>CG305</td>
<td>YURA</td>
<td>DEE</td>
</tr>
<tr>
<td>2-P*P3</td>
<td>ADKINS LINDA</td>
<td></td>
</tr>
<tr>
<td>2-P*P3</td>
<td>ADKINS LINDA</td>
<td></td>
</tr>
<tr>
<td>2++**3</td>
<td>McClain CHARLES</td>
<td></td>
</tr>
<tr>
<td>2++**3</td>
<td>McClain CHARLES</td>
<td></td>
</tr>
<tr>
<td>CG305</td>
<td>YURA</td>
<td>DEE</td>
</tr>
<tr>
<td>2++*53</td>
<td>GRINDLE BEVERLY</td>
<td></td>
</tr>
<tr>
<td>2325-1</td>
<td>CARLSON WILLIAM</td>
<td></td>
</tr>
<tr>
<td>2325-1</td>
<td>CARLSON WILLIAM</td>
<td></td>
</tr>
<tr>
<td>2325-1</td>
<td>CARLSON WILLIAM</td>
<td></td>
</tr>
<tr>
<td>2325-1</td>
<td>CARLSON WILLIAM</td>
<td></td>
</tr>
<tr>
<td>2325-1</td>
<td>MEADOWS SARA</td>
<td></td>
</tr>
<tr>
<td>B1011</td>
<td>NEPTUNE CHARLOTTE</td>
<td></td>
</tr>
<tr>
<td>CG305</td>
<td>YURA</td>
<td>DEE</td>
</tr>
<tr>
<td>2325-3</td>
<td>MURPHY SANDY</td>
<td></td>
</tr>
<tr>
<td>2325-3</td>
<td>MURPHY SANDY</td>
<td></td>
</tr>
<tr>
<td>2325-3</td>
<td>MURPHY SANDY</td>
<td></td>
</tr>
<tr>
<td>20**PP; JELLINEK HOLLIS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2++P33</td>
<td>BLIZZARD GINNY</td>
<td></td>
</tr>
<tr>
<td>2++P33</td>
<td>BLIZZARD GINNY</td>
<td></td>
</tr>
</tbody>
</table>

Table: Following persons were in the system

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>OK</th>
</tr>
</thead>
<tbody>
<tr>
<td>07/20/72</td>
<td>08:43:17</td>
<td>OK</td>
</tr>
<tr>
<td>07/20/72</td>
<td>09:10:29</td>
<td>OK</td>
</tr>
<tr>
<td>07/20/72</td>
<td>09:17:50</td>
<td>OK</td>
</tr>
<tr>
<td>07/20/72</td>
<td>11:05:06</td>
<td>OK</td>
</tr>
<tr>
<td>07/20/72</td>
<td>11:09:29</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>09:21:53</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>09:24:21</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>09:52:12</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>09:52:56</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>09:53:45</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>09:54:34</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>09:56:05</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>09:41:26</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>13:11:48</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>13:28:39</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>14:06:09</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>14:08:51</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>15:22:40</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>15:27:22</td>
<td>OK</td>
</tr>
<tr>
<td>07/21/72</td>
<td>08:51:36</td>
<td>OK</td>
</tr>
<tr>
<td>07/25/72</td>
<td>09:59:59</td>
<td>OK</td>
</tr>
<tr>
<td>07/25/72</td>
<td>11:08:44</td>
<td>OK</td>
</tr>
<tr>
<td>07/25/72</td>
<td>11:10:51</td>
<td>OK</td>
</tr>
<tr>
<td>07/25/72</td>
<td>11:13:13</td>
<td>OK</td>
</tr>
<tr>
<td>07/25/72</td>
<td>11:14:36</td>
<td>OK</td>
</tr>
<tr>
<td>07/25/72</td>
<td>11:59:38</td>
<td>OK</td>
</tr>
<tr>
<td>07/26/72</td>
<td>08:40:53</td>
<td>OK</td>
</tr>
<tr>
<td>07/26/72</td>
<td>08:56:29</td>
<td>OK</td>
</tr>
<tr>
<td>07/26/72</td>
<td>10:42:21</td>
<td>OK</td>
</tr>
<tr>
<td>07/26/72</td>
<td>10:43:40</td>
<td>OK</td>
</tr>
<tr>
<td>07/26/72</td>
<td>10:45:15</td>
<td>OK</td>
</tr>
<tr>
<td>07/26/72</td>
<td>10:46:22</td>
<td>OK</td>
</tr>
<tr>
<td>07/26/72</td>
<td>11:42:05</td>
<td>OK</td>
</tr>
<tr>
<td>07/26/72</td>
<td>14:09:06</td>
<td>OK</td>
</tr>
<tr>
<td>07/26/72</td>
<td>14:10:43</td>
<td>OK</td>
</tr>
</tbody>
</table>

Figure 7

that this is the only place in the entire system where a user has to learn a new symbol. All other commands to the system are in natural language and are very straightforward.

The DEFINE function is used to set up the control record with the quiz definitions. This includes the number of items to be on a given quiz and the breakdown for each type. This record is used by "GENER" when generating the quiz. (See Figure 6)

ROUTINE "WHO"

This routine, available only from control mode, is used to print the system activity file generated by "PASS." It shows who has entered the system and whether or not their password was accepted. The date and time are also available. (See Figure 7)

The value of this routine is to check on activity, especially if an instructor thinks his password is no longer secret.

MESSAGE ROUTINES

Message routines have been implemented. These allow communications from the instructors to the system manager through a file. It is thought that this feature may be valuable for reporting any difficulty to the system manager or for leaving suggestions for improved function.

CONCLUSIONS

Our interest in developing TASSY was to explore the problems and potentials of using the computer in the educational process. TASSY served that purpose in many ways. Our first concern was the problem of software development and record design. We originally designed the question record to be 500 characters long. We found that this is too short. Our next version of TASSY will have the ability to hold a question and associated distractors totalling 1,000 characters on each record.
At the time of this writing we are still not sure what optimal student record should look like. We estimate that the student record should have the ability to record a minimum of 75 items, the sub-scores from 10 attributes, and the total score, in addition to the necessary identification date mentioned in the system description.

A second problem we wanted to evaluate was the feasibility of operating under the auspices of a large central computer using telephone communication. Under the best circumstances, our experience has indicated that we not try it again.

Our experience was not unlike that of anyone else who has had to rely on the telephone system and someone else, i.e., the central system, to do the work for them. A third problem we encountered was that our system has had to rely on the telephone system and some else, i.e., the central system, to do the work for them. A third problem we encountered was that our system became fair game to students who would try to “break in” and look at the answers and the system. The computer center staff developed a special software “lock” for us that was, in effect, a self-destruct button. If any tampering was attempted, a system error was generated, duly logged, and the program disappeared. (See Figure 7) At times this security feature was inconvenient, but we felt the trade-off for security well worth it.

We have decided to develop TASSY to operate outside of the University’s central computer because of the cost of maintaining enough core and disc space on line. We estimate that the cost of a 16-terminal system would be almost double that of having our own minicomputer.

We are also concerned with the reactions of faculty, teaching assistants, and students. The students and teaching assistants liked TASSY very much. The students did not feel that the computer de-personalized them. In fact, most of the students felt that TASSY represented a meaningful step on the part of the faculty to meet their needs. The teaching assistants were overjoyed because the most boring 10 percent of their work assignment was removed. The faculty agreed with the idea, and liked the potential of the TASSY system.

We also hope to give the instructor his choice of scoring modes besides the traditional, rights, wrongs, etc. Finally, we hope, in the not too distant future, to be able to add some graphics capabilities to the system.

Because of the modularity of TASSY and its relatively simple-minded approach to testing and feedback we feel that it would be implemented readily by individual instructors at almost any level of college instruction. We hope that, as our experience with TASSY grows, we will be able to develop a product which will not only be easy to use, but will also be cost-effective enough so as to warrant serious consideration.

ACKNOWLEDGMENTS

The author would like to give thanks to Mrs. Rita Saltz for her invaluable editorial assistance and to the West Virginia University Computer for their patience.

REFERENCES

1 F B BAKER
Computer-based instructional management systems—A first look
Review of Education Research Feb 1971 Vol 14 No 1 pp 51-70

A C KELLEY
An experiment with TIPS—A computer aided instructional system for undergraduate education
The American Economic Review 1968 No 58 pp 446-457

From the collection of the Computer History Museum (www.computerhistory.org)