MicroCAT™ News

April 1988

Assessment Systems Corporation • 2233 University Ave., Ste. 440 • St. Paul, MN 55114 • 612-647-9220

MicroCAT Version 3.0 The Best Gets Better

We are currently completing work on Version 3.0 of the MicroCAT Testing System and plan to release it this summer. It will be sent free of charge to all users who have current maintenance agreements and to those who purchased the MicroCAT system after February 1987. An upgrade for other MicroCAT licensees will be available at a reduced price.

The major changes fall into four categories: graphics support, language features, conventional-testing features, and bank utilities. All of these changes are compatible with previous versions of the MicroCAT system so that all of the items and tests you have developed should work fine with the new version of the system.

Graphics Support

IBM has dramatically improved its graphics capabilities on the PS/2 series of computers. We have enhanced the MicroCAT system to take advantage of some of these improvements. Specifically, Version 3.0 will support both the 256-color, 320 by 200 pixel resolution mode and the 2-color, 640 by 480 pixel resolution mode of the MCGA adapter. This adaptor is standard on IBM's Model 25 and 30 and is a subset of the VGA adaptor available on the more expensive models. The 256-color mode will allow you to present graphics with extended colors and shading that are not possible now. The high-resolution mode will allow you to present much sharper monochrome graphics images than are possible now. In addition, Version 3.0 will support EGA and Hercules monochrome graphics.

We are also adding an imaging capability to the new version. This means that you will be able to digitize previously drawn images using an image scanner and import those images directly into your items using the MicroCAT Item Banking Program. You will also be able to import photographs and include them in your items.

Finally, Version 3.0 will expand the area in which your graphics items may fit. The current version provides a scrollable item type that allows the examinee to scroll through a textual item that is too big to fit on a single screen. Version 3.0 will expand that capability to graphics items. You will be able to create graphics items that are up to 1.7 times the length of the screen in several of the graphics modes. The examinee will be able scroll the item to examine any part of it on the screen.

Language Features

We have added two new features to the MCATL language, the authoring language in which MicroCAT tests are specified. The first allows you to create adaptive reading comprehension tests using the SEARCH command. The new feature, called CLUSTER, allows you to define clusters of items (e.g., those associated with a passage) and to branch from cluster to cluster

(Continued on page 4)

An Adaptive Testing Program For a Clinical Clerkship

Douglas U. Smith, Ph.D., and George F. Reed, M.D.
SUNY - Health Science Center at Syracuse Syracuse, NY

The Department of Otolaryngology at the SUNY - Health Science Center at Syracuse is currently engaged in a study designed to assess the feasibility of using adaptive testing in a third-year clinical clerkship. This project was undertaken for two reasons. First, the assessment of a student's cognitive skills has been based solely on a subjective evaluation made by the attending physician in charge of the two-week clerkship. Furthermore, since different faculty are involved in teaching and evaluating students during each of these periods, consistent and objective assessments of the students' knowledge base were difficult, if not impossible, to obtain. Second, requiring students to take a comprehensive,

(Continued on page 2)

Inside this Issue

An Adaptive Testing Program *(Continued from page 1)*

objective-based examination was not considered practical since the amount of time which would have to be set aside for testing would be excessive in relation to the total number of contact hours in the course. A lengthy examination would also have the undesirable side effect of limiting students' opportunities for additional clinical exposure. These factors led to consideration of an adaptive testing program which, if properly designed, would have the potential to permit an accurate, comprehensive assessment of the student's knowledge base in a considerably smaller amount of testing time.

The first stage of the project began with a review of 128 candidate items by departmental faculty. All items were classified into content and process categories resulting in a table of specifications for the item pool. Once classified, items were independently reviewed by another faculty member with training and experience in item writing. This review led to some items being set aside for revision or removed from the candidate set. Altogether, 100 items were retained in the pool. The percentage of items assigned to each of the content and process categories was then reviewed by the faculty and judged to be appropriate for the objectives of the course.

Next, our attention focused on developing a testing protocol that would permit the calibration of the entire item pool, while keeping the test administration time to about one hour. Estimating that students would require, on the average, between 50 and 60 seconds to answer a question, and allowing some time for students to review their test, it was clear that not all 100 items could be administered to each student. Therefore, we chose to develop two parallel test forms each having 60 items. Forty items were unique to each form, and 20 items were common to both forms. These common items will later be used to link the item calibrations from the separate forms onto a common scale.

During actual test construction, the 20 common items were selected first. Using item statistics obtained from pretesting the items on a smaller but similar group of students, the common items to be used for linking items were chosen from among the more highly discriminating items found within categories of item difficulty ranging from 0.20 to 0.80. From the remaining set of 80 items, 40 items were randomly selected and assigned to Form A; the remainder to Form B. Following test construction, a review of the content/process specifications for both forms found them to be quite similar to one another and to the total pool.

An additional concern we had was that of maintaining the security of the test questions. Since only a small number of students would be tested during any two week period, the collection of minimally sufficient response data for item calibration would require approximately six to eight months. During that time it is conceivable that students might attempt to pass information about the test items on to others. Although having two forms of the test would minimize this problem somewhat, additional security measures were deemed necessary. As a result, we also chose to randomize the order in which items were presented to students.

Using MicroCAT's Development Subsystem, items were entered into a bank of questions, then carefully reviewed for errors. Next, test specifications for both forms were developed. These specifications called for conventional, fixed-length tests consisting of 60 items to be presented at random (without replacement) from items comprising the particular test form. In addition, several evaluation items, designed to gather specific student feedback regarding both the test and the testing format were appended to the end of each test. These items required both student-selected and student-generated (i.e. open-ended) responses.

The second phase of the project, now underway, involves administering the tests to approximately 300 third and fourth year medical students (100-150 students per form). Several days prior to testing, each student is provided with a handout describing the testing format, microcomputer operating procedures (including a keyboard diagram), and directions on how to enter, correct, and review responses. During the afternoon on the last day of the clerkship, tests are administered using one of two IBM PC-compatible microcomputers located in the department's medical library. Both microcomputers are equipped with a color monitor, dual floppy disk drives, and 512K of memory. The MicroCAT Examination Subsystem and the necessary test files for Form A are installed on one microcomputer, while the files for Form B are installed on the other. Pertinent scores, complete item response data (including response latencies), total testing time, and evaluation responses are recorded for later use.

Thus far, the information we have obtained has been used to monitor both ongoing testing as well as students' reactions to the computer-based testing format. The Conventional Item and Test Analysis Program (ITEMAN) and several specialized, locally developed BASIC programs have assisted in this process. Preliminary findings to date have been encouraging. First, total testing time (including review) has averaged just under 60 minutes—near what was considered to be our practical limit in terms of testing time. Second, although early item analysis results have identified a few misbehaving items, the test means, variances, and average response latencies for the

(Continued on page 5)

MicroCAT™ News

is published by Assessment Systems Corporation, 2233 University Avenue, Suite 440, St. Paul, MN 55114, USA, 612-647-9220.

President — C. David Vale
Editor — J.S. Prestwood

Tutorial: Defining New Alphabets

You may remember a brief description in our last newsletter of undocumented features in the MicroCAT system. Two of these are useful in entering text in foreign languages or other special alphabets. We will describe how these features can be used in more detail in this tutorial.

The standard alphanumeric character you see on the screen of an IBM personal computer is defined by a matrix of pixels (or "picture elements"). The matrix is usually eight pixels wide and eight pixels tall, although in some high-resolution display modes, more pixels may be used. In the MicroCAT system, however, all characters are either defined directly by this matrix or by doubling the height or width of the matrix.

The standard 128 characters of the ASCII character set contain most of the common symbols (letters, numbers, and punctuation) you use in creating item text. This leaves a second set of 128 (actually 124 in the MicroCAT system) for special characters. This set can be used to create a special alphabet for foreign-language items or other uses. You are probably already aware of the font generating capability (MAKEFONT) provided with the MicroCAT system for creating special characters. In this tutorial, we will briefly review that function and then discuss how it can be used to conveniently create items using a special alphabet.

Creating Special Characters with MAKEFONT

The MAKEFONT command is described in Chapter 4 of the MicroCAT User's Manual. As described, you start the Font Generation Program by entering the command MAKEFONT, which may be followed by a name for the special character file. For this tutorial, let's say we are going to create a special italic version of the standard U.S. alphabet. We can begin by entering

```
MAKEFONT ITALIC
```

in response to the Development Subsystem command menu.

This command tells the font generator that you want to create an set of characters (i.e., a font) and store it in a file called ITALIC (the actual file name will be ITALIC.CST). If a font called ITALIC already exists, the font generator will load that file so that the characters in it can be modified or new ones can be added.

Once you press the return key, a screen like that shown on page 52 of the manual will appear. You can then follow the instructions on the screen to choose character codes and define their corresponding characters by turning individual pixels on and off in the matrix shown on the screen. This process is described in Chapter 4 of the manual, however, and we won't discuss it again here.

Creating Special Alphabets with MAKEFONT

A feature of the item banker that we will discuss later allows you to shift the keyboard into a special-character mode so that the special characters can be entered with a single keystroke. Since our characters are simply variations of the standard characters, it would be very convenient if our italic "A" could be obtained by pressing the standard character "A"

There is a trick to choosing the number that corresponds to each of the special characters. All of the standard characters have a number assigned to them. This number is the ASCII code (a list of the ASCII codes are included in most books on microcomputer programming). The number entered when the banker is in special-character mode is exactly 128 greater than that of the corresponding standard character.

For example, the capital letter "A" is represented by the number 65. If we choose the number 193 (65+128) for our capital italic *"A"*, then we can type a shift-A while in special-character mode to enter a capital italic *"A"* in our item text.

The letters "A" through "Z" are represented by the numbers 65 though 90, the letters "a" through "z" by 97 through 122, and the numbers "0" through "9" by 48 through 57. Just as we chose the number 193 for the capital italic *"A"*, we can choose 194 (66+128) for the capital italic *"B"*, or 226 (98+128) for the lower-case italic *"b"*, etc.

Entering Special Characters with BANK

The usual special character font is called SPECCHAR. It is loaded into the item banker by default. If you want to use a different font, you must explicitly load it. You do this with the Action Menu in the item banker (see Figure 3-2 in the User's Manual).

After you have loaded the font, you can use it in creating items. The standard characters (the first 128) will still be available, too. To enter text, you first have to define a text box. If you only plan to enter a few special characters, you may do it with the numeric keypad by holding down the ALT key and pressing the three numbers corresponding to the character's code.

If, however, you have a lot of special characters to enter, you will probably want to switch to special-character mode. Do this by holding the ALT key and pressing the "H" key (for High character set). Now, every key you press will give you the special character with a numeric code 128 higher than that of the key you pressed. To get back to the standard character set, hold the ALT key and press "H" again. Closing the text box will also return the keyboard to the standard mode.

A related facility is useful if you are entering items from a foreign alphabet that is read from right to left. Holding the ALT key and pressing "D" (for Direction) before you enter the first character in a text box or in a B/W Text item will cause the direction to change from left-to-right to right-to-left. This feature, of course, is not necessary for our application of italic characters. The right-to-left text mode will remain set until you reset it in another text box or exit from the item in which you set it.

Creating Tests That Use the Special Characters

As was true of the item banker, the test compiler (COMPILE) assumes that your special characters are contained in the file SPECCHAR, unless you tell it otherwise. The statement in the MCATL language that allows you to do this is the LOADFONT statement. To

(Continued on page 6)

MicroCAT Version 3.0 *(Continued from page 1)*

rather than from item to item. Clusters are selected on the basis of the average psychometric information they provide. Clusters may also prove suitable for balancing the content of an adaptive test if each cluster contains items that cover all of the content facits included in the test.

The second language feature is an AUTOEXEC section. You may already be familiar with the AUTOKEEP statement that writes to the data file each time the examinee responds to an item. The AUTOEXEC section is an expansion of this idea. At the beginning of your test specification, you will be able to set aside a section of MCATL statements that you want to be executed each time the examinee responds. Then, during the test, this entire set of statements will be executed every time an item is administered. This will allow you to do things like score individual items and then write their right/wrong codes in the data file.

Conventional Testing Utilities

As many of you know, the MicroCAT system was originally developed for administering on-line tests. The test printing capabilities came later when the Conventional Testing Subsystem was released with Version 2.0. We are adding new capabilities to this subsystem for Version 3.0.

First, we are expanding the Test Printing Program. The new version will be able to print the new high-resolution items you create with the new item banker. We are also improving the format of the output to allow you to produce more professional-looking tests.

Second, we are improving the Test Building Program. The current version only allows you to choose items at random from within specified item banks. With the new version, you will be able to select items on the basis of their parameters and other characteristics as well.

Bank Utilities

Five new bank utility functions will be included as part of Version 3.0. They are RENAME, COPY, RECOVER, PACK, and REBUILD.

Those of you who have tried to rename a bank using the DOS Rename command will appreciate the RENAME utility. MicroCAT item banks contain the name of the bank within the file. Renaming just the bank file with the DOS command does not change the internal name. In fact, trying to rename the file with the DOS command can confuse the MicroCAT programs and result in the loss of some items. The RENAME utility, however, will let you rename banks properly.

The COPY utility will allow you to make a new copy of a bank and to simultaneously rename all of the items in it.

If you have ever erased an item from the bank and then wished that you hadn't, you will appreciate the RECOVER function. The RECOVER function will allow you to get it back if you have not renamed, packed, copied, or rebuilt the bank with one of the other new utilities.

When you erase an item from a bank, you do not actually remove it but you erase the pointer to that item in the bank's directory. Thus the item continues to occupy space in the bank file even though you no longer have a use for it. The PACK utility will remove all of the dead space from a bank and may result in considerable space savings for some older banks.

Finally, the REBUILD utility will help recover items from a damaged bank. We're not sure how banks become damaged, but we've had user's do it once or twice. The REBUILD function will search through the remnants of the bank, find all of the items that are still intact, and rebuild the item directory. You will probably never need this function but maybe you'll sleep better knowing that you have it!

We hope that you will find these new features of benefit to you in your work with the MicroCAT Testing System. Beta-test versions of the new version should be available in June to interested users.

Extended Runtime License Available

The original licensing plan for the MicroCAT Testing System called for one Examination Subsystem license for each testing station you set up. Although we kept the price of this license quite low ($50), some of our users who have set up testing laboratories have found it inconvenient to keep track of exactly how many licenses they need to remain within the law. Similarly, some users planning large implementations of the system have found the cost prohibitive for setting up several hundred stations around the country.

We have therefore decided to make an extended runtime license available to test developers. This license will allow users to administer their tests at any number of testing stations or sites for a single fee. The cost of the extended runtime license is $900. Now, any user planning to set up 18 or more testing stations can make a one-time investment. The runtime license is, of course, limited to the Examination Subsystem and only allows distribution outside the original licensee's organization for administering tests developed by the original licensee.

Many of our current users have already purchased multiple Examination Subsystem licenses. Those who have purchased 18 or more will automatically be granted the extended runtime license for no additional charge. Those who have purchased fewer than 18 will be credited for the licenses already purchased if they want to buy an extended runtime license.

Ask The Experts

Q. Space-bar response selection is great but how can I get rid of the "ANSWER" prompt that appears at the bottom of the screen?

A. The SETMESSAGE statement in the MCATL language will do the job. If you just want to eliminate the prompt, use the statement

```
SET MESSAGE MSG-ANSWER ("")
```

A better approach, however, might be to substitute a more appropriate prompt (e.g., "Space Bar to select—Return to enter"). To substitute a prompt, insert it between the double-quotation marks within the parentheses in the statement above. For more details on the SETMESSAGE statement, see Pages 74-75 in the MicroCAT manual.

Q. How can I write two lines to the output file with the AUTOKEEP statement?

A. You can't with the current version of the system. Version 3.0, however, will let you do this by including two KEEP statements in an "AUTOEXEC" section. Any statement(s) included in the AUTOEXEC section will be executed whenever a response is recorded.

Q. When creating an item with text and graphics, I have found it convenient to draw the graphics first and then define a single text box that covers the entire screen. That way I can position and enter all my text at once. Will this cause me any trouble?

A. It shouldn't as long as you're careful. In the earliest versions of the MicroCAT system, there was no "Textover" feature in the item banker. When text was written to the screen, a box behind each character was set to the background color. Thus, any blank characters in a text box wiped out whatever figures they covered. As long as the Textover characteristic in the banker is set to "Y" (it is always set to "Y" unless you change it), you should have no trouble with your approach.

Q. Often when I'm creating an item, I have to delete a part of it I drew several commands previously. I know I can step back through the item and delete just that part, but sometimes, I delete the wrong thing. Is there an Undo command in the item banker?

A. Not under that name, but there is a safe way to delete portions of an item. Start by stepping back through the item as you usually do but continue to step back past the part you want to delete. When what you want to delete disappears from the screen, step forward one command and watch it reappear. At that point you can issue the delete command and be sure that the correct command will be deleted.

Q. The segmenting capability of the item banker is very useful when I want to repeat portions of an item in several places on the screen, but when I change colors within the segment, the drawing comes out a different color every time I execute the segment. What's wrong?

A. Colors are chosen by the Color command, and each Color command steps to the next color until the first color is reached again. When you change the color within a segment, be careful to change it back to the original color before you close the segment. Then, you can execute the segment several times and the colors you expect will appear.

Say, for example, the color is currently green and you want to change it to yellow within the segment. Issue two Color commands within the segment to change the color to yellow at the appropriate time. Then, after you have finished drawing the segment, issue two more Color commands to get the color back to green before you close the segment. Then when you execute the segment several times, the colors will be the same as when you defined them.

Also, if you change colors between the time you define the segment and the time you execute it, or between two Execute commands, you must still remember to reset the foregound color to green before you proceed. Otherwise, the segment will be drawn in colors other than those you used when you defined it.

An Adaptive Testing Program

(Continued from page 2)

items on both forms have been quite similar.

Student feedback, too, is helping us identify unforeseen problems and has provided us with a glimpse of their perceptions with respect to the computer-based testing format. In general, student acceptance has been favorable. Aside from its novelty, students have cited the ease by which responses can be entered and changed, better pacing, and the ability to pay greater attention to individual questions as advantages to the computer-based approach. Eye strain, inability to write notes next to a question, and not being able to skip back and forth between items during the test have been mentioned as drawbacks.

Implementing an adaptive testing program within a clinical clerkship is hampered by a number of factors including small class size (approximately 150 students per year) and the nature and organization of the clinical curriculum. For example, since it is possible to test only six to eight students every two weeks, we expect that it will take at least a year before we can obtain even a minimally sufficient number of test administrations for item calibration. In order to expedite this process, we anticipate that students rotating through a similar clerkship at a nearby medical school will be able to access our item bank through a remote microcomputer equipped with a modem and telecommunications software. Currently, both the hardware and software necessary for making this possible are being tested locally.

By the end of this year, when sufficient data have been collected and following some further data analyses, item calibration will be attempted using methods based on item response theory (IRT). Soon after this has been completed, we will begin implementation and evaluate the efficacy of adaptive testing within this clinical clerkship setting.

Management Subsystem To Be Dropped

The Management Subsystem supports testing stations integrated into local area networks. We developed the Management Subsystem several years ago when networking systems were quite rudimentary and did not provide the facilities necessary to support a testing center (such as a convenient way to collect all data on a single machine).

Since that time, local area networking hardware and software has become more powerful and more varied. At the same time, the Management Subsystem has become less necessary and more difficult to support.

With the release of Version 3.0 of the MicroCAT Testing System, we will drop the Management Subsystem from our official product line. The new version of the system will still be compatible with the Management Subsystem for those who are using it, and we will make the subsystem available to new users who specifically request it. We will not, however, continue to advertise or further refine it.

Tutorial *(Continued from page 3)*

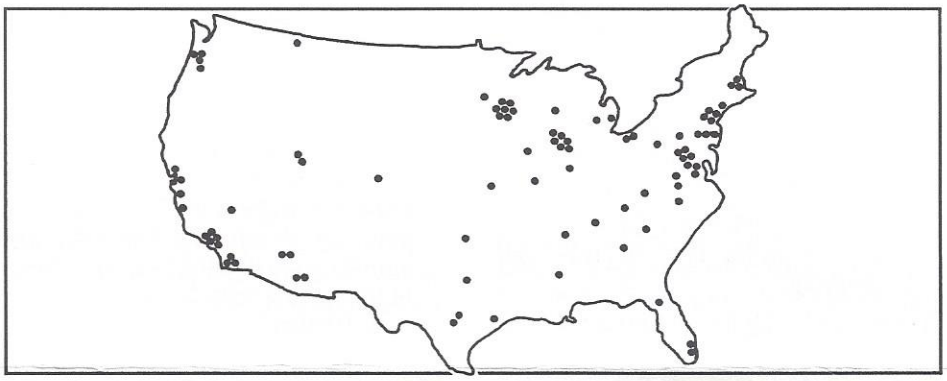
use the new italic characters, the statement

```
LOADFONT ITALIC
```

should appear in the test specification before any items that need the special characters are administered. The special characters in ITALIC will be used in all items until another LOADFONT statement is encountered.

Where Are They Now?

There are currently over 80 MicroCAT Testing Systems installed around the world and many additional users of the individual item analysis programs ITEMAN, RASCAL, and ASCAL. Users can be found in such diverse locations as England, Belgium, India, Australia, and both Chinas! As you might expect, most of the installations are in the United States. The map below shows the locations of MicroCAT users in the lower 48 states. Did we forget anyone?



SAINT PAUL
APR 22 '88
MINN.
U.S. POSTAGE
0.25
P.B. METER
1471106

MicroCAT™ News

Assessment Systems Corporation
2233 University Avenue, Suite 440
St. Paul, MN 55114

FIRST CLASS

Rick Blasenborg
32 Hickory Lane
Amherst, MA 01002